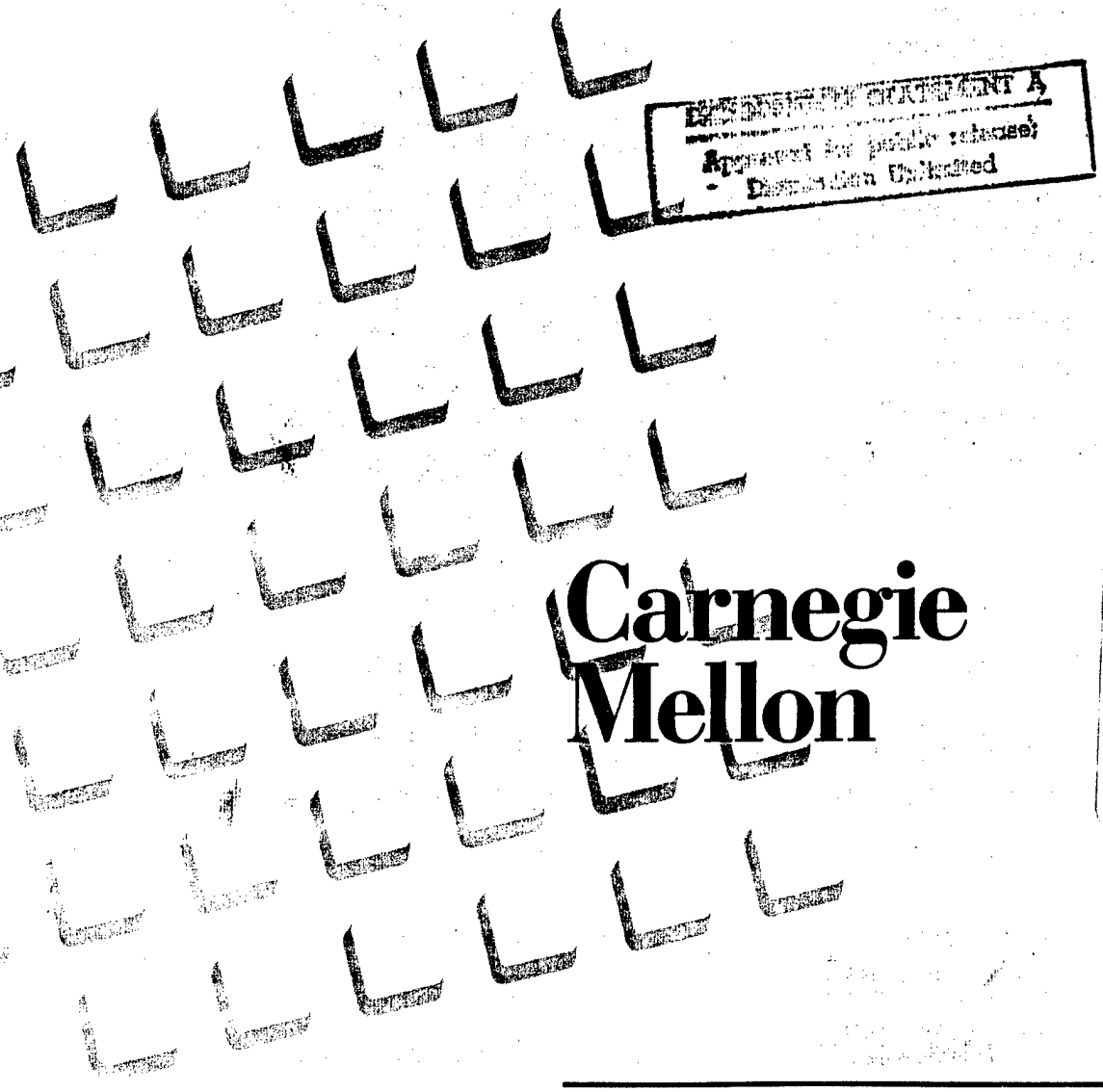
Computer Science

Translucent Cache Management for Mobile Computing

Maria R. Ebling
March 1998
CMU-CS-98-116

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited



Carnegie Mellon

19980903 123

Translucent Cache Management for Mobile Computing

Maria R. Ebling
March 1998
CMU-CS-98-116

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Thesis Committee:
Mahadev Satyanarayanan, Chair
Bonnie E. John
James H. Morris
Douglas B. Terry, Xerox PARC

Copyright © 1998 Maria R. Ebling

This research was sponsored by the Defense Advanced Research Projects Agency (DARPA), Air Force Materiel Command, USAF under agreement numbers F19628-96-C-0061 and F19628-93-C-0193, the Xerox Corporation, and the Intel Corporation. Some of the equipment used in the process of conducting this research was acquired through the National Science Foundation Equipment Grant #9022511.

The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Intel, Xerox, DARPA, or the U.S. Government.

Keywords: Disconnected operation, distributed file systems, high availability, mobile computing, caching, transparency, optimistic replication, hoarding, user interface, Coda.

School of Computer Science

DOCTORAL THESIS

in the field of

COMPUTER SCIENCE

Translucent Cache Management for Mobile Computing

MARIA EBLING

Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

ACCEPTED:

THESIS COMMITTEE CHAIR	**DATE** 3/23/98
DEPARTMENT HEAD	**DATE** 5/14/98

APPROVED:

DEAN	**DATE** May 14, 1998

To Chris

Abstract

Recently, distributed files systems have aggressively exploited caching to provide high availability to their clients. Such systems allow users to access and modify cached data even when clients become disconnected (e.g. a laptop removed from its docking station) or weakly connected (e.g. a laptop connected via a modem) to servers. In doing so, these file systems violate one of the key properties of caching: transparency.

When disconnected or weakly connected, these systems cannot always maintain the illusion of connectivity. For instance, the system may not be able to service a cache miss or may only be able to do so with a lengthy delay. When the illusion of connectivity breaks down, the usability of the system is undermined. Early designers of these systems either did not recognize or failed to appreciate the extent to which users would be affected. After extensive experience with the Coda file system, however, we have witnessed the arduous learning curve experienced by new Coda users. We have also observed occasional, but serious, confusion caused by modal behaviors introduced to cope with the range of network connectivity encountered by mobile clients.

This dissertation describes the design, implementation, and evaluation of the CodaConsole, a graphical interface that makes caching *translucent* to users of such systems. To make caching translucent, the interface exposes critical aspects of caching to support availability, while hiding noncritical details to preserve usability. For example, the system informs users about the availability of their tasks, but does not bother them with minutia of cache consistency. The interface also offers opportunities for users to control where network resources are spent.

To evaluate the CodaConsole, I performed a usability test. This study showed that users were able to learn how to use the interface in about one hour. They were able to operate the interface with little difficulty. They were also able to understand problems indicated by the interface and then determine what actions might resolve those problems. Perhaps the most surprising result, however, was the fact that novice Coda users performed almost as well as expert Coda users. This interface successfully makes caching, in the presence of disconnected or weakly connected operation, translucent.

Acknowledgments

Graduate school has been a long and arduous journey. It has been a time of learning and of personal growth. People make Carnegie Mellon a special place and I would like to thank those who helped me during my journey here. In writing these acknowledgements, I have undoubtedly forgotten people who deserve to be mentioned. If I forgot to mention you by name, I apologize!

I would like to begin by thanking my advisor, Satya. He's a brilliant researcher who has provided me with valuable technical guidance and made time for me in his busy schedule. More than anything else, Satya taught me to ask questions and to let those questions be my guide. Other members of my committee also have my gratitude. Jim Morris made me consider users of non-Unix systems; he also had many insightful comments about the bigger picture. Bonnie John opened the doors to HCI and provided support throughout. Doug Terry, my host when I was a summer intern, introduced me to industrial research. In the process, he provided a light at the end of a very long, very dark tunnel.

The past and present members of the Coda group—Peter Braam, Jay Kistler, Puneet Kumar, Yui-Wah Lee, Qi Lu, Lily Mummert, Brian Noble, Henry Pierce, Josh Raiff, and David Steere—deserve special thanks. Without many of their contributions, my thesis would never have been. The Coda users throughout the years also deserve special recognition for the frustration inherent in depending on a research system. I would like to personally thank Stuart Clamen, David Eckhardt, Juan Leon, Henry Rowley, and Amy Moormann Zaremski for the comments and insights they gave me about the problems new and experienced users of Coda faced.

My fellow graduate students lent a unique kind of support during their own journeys. I extend my thanks to Vince Cate, Herb Derby, David Maltz, Francesmary Modugno, Lily, Brian, Sue Older, John Pane, Sasha Wood, and especially to Amy, Jennifer Kay, and Matt Zekauskas. I greatly appreciate the neighborly advice, friendship, and encouragement you all provided. Friends beyond the confines of Wean Hall—including Rea Freeland, Roseanne Hickey, Mark Zaremski, Carrie O'Donnell, David Groat, and Mike and Marcy Langer—reminded me about life on the outside.

A very special thanks goes to Amy, Jennie, Rea, David, and Chris Okasaki for reading earlier versions of my thesis. They all provided me with valuable comments and greatly improved the final document. Sharon Burks deserves more thanks than anyone could ever give her. She cuts through red tape and provides invaluable advice.

I would like to thank the administrators of Carnegie Mellon's User Studies Lab for the use of their facilities, particularly Sandy Esch and Tom Pope. Sandy deserves special

recognition for her commitment to the lab. Without her expertise and practical advice, performing a user study would be a much more difficult process. The participants in the user study, whom I cannot thank by name but who know who they are, also deserve special recognition. Each of them volunteered the better part of a day to help me with my research. I greatly appreciate their generosity.

A number of programs also deserve special recognition. These include tck/tk and the tix widget library, Kurzweil Voice, and Microsoft Word. The graphical interface that I built as part of my thesis research was implemented with tcl/tk using the tix widget library. Though I believe that my interface gradually outgrew these programs, I found them to be incredibly useful tools. I would like to thank Brent Welch for his advice in using canvases, back before we had his wonderful reference book. I would also like to thank Ioi Lam for building the tix widget library. This thesis was written with the assistance of the Kurzweil Voice program. Voice recognition has finally matured into toddlerhood. At times, it amazes you with the wonder of the things it can do; yet, at other times, it frustrates you with its stubborn defiance of your authority. As a whole, Voice has given me more freedom than I have had in years and it has enabled me to work much longer hours at the computer. For that, I am extremely grateful, but I eagerly await the day when voice recognition systems mature beyond the "terrible twos." Microsoft Word, on the other hand, deserves a different kind of recognition. Its ability to corrupt documents never failed to amaze me. In fact, this *commercial product* holds the dubious distinction of corrupting more of my data in six months' time than the *experimental* Coda file system corrupted in approximately seven years. Some day, I hope that its designers have the joy of writing a large document using their own product. May they live interesting lives.

My parents taught me to persevere in the face of obstacles, but they also taught me a love of knowledge. My mother typed in the whole of Appendix C and parts of Appendix B on her "vacation" and also proofread a draft of the thesis in her "spare" time. Without the influence of my father, I would never have come as far as I have. He kept me in the mathematics pipeline even as most of my friends dropped out, and I thank him. They provided me with a strong foundation on which to build, but let me find my own path. I hope I can do as well for my son!

Colin taught me to leave school at school and to enjoy the simple things in life. Motherhood has been a new challenge. It is infinitely rewarding, but also requires infinite amounts of patience, especially now that we have hit the two-somethings. You are a wonderful little boy and I look forward to seeing the man that you will one day become!

I do not quite know how to thank Chris, my husband and best friend. He stood with me when I hit the lowest of the lows, and suggested that I wait one month before quiting (a trick we learned in our childbirth class with "month" substituted for "hour" and "quit" substituted for "painkillers"). Without his support and encouragement these past few years, I can honestly say I would not have completed this degree. Thank you!

Maria Ebling
Pittsburgh, Pennsylvania

Table of Contents

1 INTRODUCTION 1

1.1 THE THESIS 2
1.2 BACKGROUND AND HISTORICAL PERSPECTIVE 2
1.3 LACK OF TRANSPARENCY 3
1.4 PERSPECTIVE OF USERS 5
1.5 ROADMAP OF DISSERTATION 7

2 CONTEXT AND BACKGROUND 9

2.1 INFLUENCE OF AFS 9
2.1.1 Client/Server Model 10
2.1.2 Volumes 10
2.1.3 Client Caching 11
2.1.4 User Authentication 11
2.2 EXTENSIONS FOR HIGH AVAILABILITY 11
2.2.1 Server Replication 12
2.2.2 Disconnected Operation 12
2.2.3 Weakly Connected Operation 14
2.3 USER INTERFACES 15
2.3.1 The Repair Tool 16
2.3.2 The Hoard Program 16
2.3.3 The Spy Tool 16
2.4 SUMMARY 18

3 DESIGN RATIONALE 19

3.1 WINDOWS OF TRANSLUCENCE 19
3.1.1 Presenting Error Conditions 19
3.1.2 Rationalizing Behaviors 20
3.1.3 Controlling Resources 21
3.2 BURDENS OF TRANSLUCENCE 24
3.2.1 Failure of Availability 24
3.2.2 Excessive Time Investment 24
3.2.3 Overloading the User's Cognitive Capacity 25
3.3 PRINCIPLES OF TRANSLUCENCE 25
3.4 OVERVIEW OF TRANSLUCENT CACHING 27
3.4.1 Feedback 27
3.4.2 Control over Critical Resources 28

4 INTERACTING WITH THE USER 31

4.1 THE INDICATOR LIGHTS INTERFACE 32
4.2 CONTROL PANEL 34
4.2.1 Event Configuration 34
4.2.2 Urgency Colors 36
4.2.3 Physical Connectivity 36

4.2.4 Logical Connectivity *37*
4.2.5 Behavior *37*
4.3 TOKENS 37
4.4 SPACE 39
4.5 NETWORK 39
4.6 ADVICE 42
4.6.1 Requesting Advice from the User *43*
4.6.2 Offering Advice to the User *48*
4.7 HOARD WALK 49
4.8 REINTEGRATION 51
4.9 REPAIR 52
4.10 TASK 53

5 IMPLEMENTATION 57

5.1 ARCHITECTURE OF SYSTEM 57
5.2 DETAILS OF IMPLEMENTATION 58
5.2.1 User Interface *59*
5.2.2 Advice Monitor *60*
5.2.3 Venus Modifications *62*
5.3 USE OF TRANSLUCENT CACHING API 69
5.4 SUMMARY 74

6 REDUCING THE BURDEN ON USERS 75

6.1 HOARDING PROGRAMS AUTOMATICALLY 75
6.1.1 Design Requirements *76*
6.1.2 Instrumentation *76*
6.2 ADVISING ABOUT HOARD CONTENTS 79
6.2.1 Heuristics *80*
6.2.2 Instrumentation *81*
6.3 MODELING USER PATIENCE 83
6.3.1 Proposed Model *83*
6.3.2 Exploring the Parameters *85*
6.4 SUMMARY 94

7 EVALUATING THE INTERFACE 95

7.1 HCI EVALUATION TECHNIQUES 95
7.1.1 Purpose *96*
7.1.2 Selecting Participants *96*
7.1.3 Selecting Tasks *97*
7.2 USABILITY TEST: GOALS 97
7.3 USABILITY TEST: METHOD 98
7.3.1 Participants *98*
7.3.2 Treatment *99*
7.3.3 Measurements *100*
7.4 USABILITY TEST: RESULTS 101
7.4.1 Learnability *101*
7.4.2 Understandability *102*
7.4.3 Operability *102*
7.5 USABILITY TEST: ANALYSIS 104
7.5.1 Tutorial *108*
7.5.2 Exercises *118*
7.5.3 Evaluation Questionnaire *127*
7.5.4 Verbal Protocol *128*
7.6 USABILITY TEST: FINDINGS 132

7.6.1 Issues with Approach *132*
7.6.2 Bugs *134*
7.6.3 Usability Problems *134*
7.6.4 Omissions *135*
7.6.5 Confusing Behaviors *135*
7.6.6 Enhancements *137*
7.7 REFLECTIONS 137
7.7.1 Recruiting Users *138*
7.7.2 Pilot Testing *138*
7.7.3 Benefit of Transcripts *140*
7.7.4 Optimal Number of Participants *142*
7.8 SUMMARY 147

8 RELATED WORK 149

8.1 HOARDING 149
8.1.1 Little Work *151*
8.1.2 FACE *151*
8.1.3 Briefcase *152*
8.1.4 Transparent Analytical Spying *152*
8.1.5 Seer *153*
8.1.6 Laputa *154*
8.2 DASHBOARD METAPHOR 154
8.3 OPEN IMPLEMENTATION 155
8.4 SUMMARY 157

9 CONCLUSION 159

9.1 CONTRIBUTIONS 159
9.2 FUTURE WORK 162
9.2.1 Validating Aspects of the Interface *162*
9.2.2 Improving the Interface *164*
9.2.3 Extending the Translucence *166*
9.3 FINAL REMARKS 170

A TRANSLUCENT CACHING API 171

A.1 CALLS SERVICED BY VENUS 171
A.1.1 NewAdviceService *171*
A.1.2 ConnectionAlive *172*
A.1.3 GetServerInformation *172*
A.1.4 RegisterInterest *173*
A.1.5 GetCacheStatistics *174*
A.1.6 OutputUsageStatistics *175*
A.1.7 SetParameters *175*
A.1.8 ResultOfASR *176*
A.1.9 Imminent Death *176*
A.2 CALLS SERVICED BY ADVICE MONITOR 177
A.2.1 LostConnection *177*
A.2.2 TokensAcquired *177*
A.2.3 TokensExpired *178*
A.2.4 ActivityPendingTokens *178*
A.2.5 SpaceInformation *179*
A.2.6 ServerAccessible *179*
A.2.7 ServerInaccessible *179*
A.2.8 ServerConnectionWeak *180*
A.2.9 ServerConnectionStrong *180*

A.2.10 NetworkQualityEstimate *181*
A.2.11 Reconnection *181*
A.2.12 ReadDisconnectedCacheMissEvent *182*
A.2.13 WeaklyConnectedCacheMissEvent *182*
A.2.14 DisconnectedCacheMissEvent *183*
A.2.15 HoardWalkAdviceRequest *184*
A.2.16 HoardWalkBegin *184*
A.2.17 HoardWalkStatus *184*
A.2.18 HoardWalkEnd *185*
A.2.19 HoardWalkPeriodicOn *185*
A.2.20 HoardWalkPeriodicOff *186*
A.2.21 ObjectInConflict *186*
A.2.22 ObjectConsistent *186*
A.2.23 ReintegrationPendingTokens *187*
A.2.24 ReintegrationEnabled *187*
A.2.25 ReintegrationActive *188*
A.2.26 ReintegrationCompleted *188*
A.2.27 TaskAvailability *188*
A.2.28 TaskUnavailability *189*
A.2.29 ProgramAccessLogAvailable *189*
A.2.30 ReplacementLogAvailable *190*
A.2.31 InvokeASR *190*

B MATERIALS 191

B.1 PROCEDURES 192
B.2 CONSENT FORM 196
B.3 BACKGROUND QUESTIONNAIRE 198
B.4 TUTORIAL 204
B.5 EXERCISES 226
B.6 EVALUATION QUESTIONNAIRE 232
B.7 ANSWER KEY TO EXERCISES 237
B.8 PAR TO USABILITY STUDY EXERCISES 241

C EXPERIMENTAL DATA 253

C.1 DEMOGRAPHIC DATA 254
C.2 QUANTITATIVE DATA 258
C.3 QUALITATIVE DATA 265

D TRANSCRIPTS 275

D.1 C1 276
D.1.1 Tutorial *276*
D.1.2 Exercises & Debriefing *279*
D.2 E1 276
D.2.1 Tutorial *287*
D.2.2 Exercises *290*
D.2.3 Debriefing *292*
D.3 E2 287
D.3.1 Tutorial *294*
D.3.2 Exercises *296*
D.3.3 Debriefing *297*
D.4 E3 294
D.4.1 Tutorial *298*
D.4.2 Exercises *302*
D.4.3 Debriefing *305*

D.5 N1 298
D.5.1 Tutorial *307*
D.5.2 Exercises *310*
D.5.3 Debriefing *312*
D.6 N2 307
D.6.1 Tutorial *312*
D.6.2 Exercises *316*
D.6.3 Debriefing *319*
D.7 N4 312
D.7.1 Tutorial *319*
D.7.2 Exercises *321*
D.7.3 Debriefing *322*
D.8 P1 319
D.9 P2 322
D.9.1 Tutorial *323*
D.9.2 Exercises *323*
D.9.3 Debriefing *324*
D.10 P3 323
D.10.1 Tutorial *325*
D.10.2 Exercises *329*
D.10.3 Debriefing *330*
D.11 P4 325
D.11.1 Tutorial *331*
D.11.2 Exercises *334*
D.11.3 Debriefing *336*
D.12 P5 331
D.12.1 Tutorial *337*
D.12.2 Exercises *338*
D.12.3 Debriefing *340*
D.13 S1 337
D.13.1 Tutorial *341*
D.13.2 Exercises *343*
D.13.3 Debriefing *345*
D.14 T1 341
D.14.1 Tutorial *346*
D.14.2 Exercises *351*
D.14.3 Debriefing *353*

E ANALYSIS 355

E.1 IDENTIFYING TUTORIAL DIFFICULTIES 355
E.1.1 Keystroke-Level Model *355*
E.1.2 Screen Difficulty Analysis *364*
E.1.3 List of Tutorial Results *368*
E.1.4 Summary of Tutorial Results *369*
E.2 IDENTIFYING EXERCISE DIFFICULTIES 370
E.2.1 Metrics *370*
E.2.2 List of Exercise Results *372*
E.2.3 Summary of Exercise Results *373*
E.3 IDENTIFYING QUALITATIVE OPINIONS 374
E.3.1 List of Qualitative Results *374*
E.3.2 Summary of Qualitative Results *376*
E.4 IDENTIFYING SPONTANEOUS OPINIONS 377
E.4.1 List of Spontaneous Comments *377*
E.4.2 Summary of Spontaneous Comments *386*

E.5 SUMMARY OF RESULTS 393
E.5.1 General *394*
E.5.2 Indicator Lights *399*
E.5.3 Control Panel *400*
E.5.4 Tokens *402*
E.5.5 Space *402*
E.5.6 Network *402*
E.5.7 Advice *403*
E.5.8 Hoard Walk *403*
E.5.9 Task *404*
E.5.10 Study Materials *409*

F USABILITY FINDINGS 413

F.1 LEVEL 1 414
F.1.1 Global *414*
F.1.2 Local *414*
F.2 LEVEL 2 418
F.2.1 Global *418*
F.2.2 Local *419*
F.3 LEVEL 3 421
F.3.1 Global *422*
F.3.2 Local *426*
F.4 LEVEL 4 432
F.4.1 Global *432*
F.4.2 Local *433*

BIBLIOGRAPHY 441

List of Figures

FIGURE 2.1: VENUS STATES AND TRANSITIONS 15
FIGURE 2.2: HOARD PROGRAM COMMAND SET 17
FIGURE 2.3: SAMPLE HOARD PROFILE 17
FIGURE 2.4: SAMPLE SPY OUTPUT 18

FIGURE 4.1: INDICATOR LIGHTS 33
FIGURE 4.2: THE TABS OF THE CONTROL PANEL 35
FIGURE 4.3: THE WINDOWS OF THE TOKENS INDICATOR 38
FIGURE 4.4: SPACE INFORMATION WINDOW 40
FIGURE 4.5: NETWORK INFORMATION WINDOW 41
FIGURE 4.6: ADVICE INFORMATION WINDOW 42
FIGURE 4.7: DISCONNECTED CACHE MISS QUESTIONNAIRE 44
FIGURE 4.8: WEAK-MISS INTERACTION 45
FIGURE 4.9: READ DISCONNECTED CACHE MISS QUESTIONNAIRE 46
FIGURE 4.10: HOARD WALK ADVICE 47
FIGURE 4.11: ADVICE REGARDING HOARD CONTENT 50
FIGURE 4.12: HOARD WALK INDICATOR WINDOWS 50
FIGURE 4.13: REINTEGRATION INFORMATION 51
FIGURE 4.14: REPAIR INFORMATION 52
FIGURE 4.15: THE WINDOWS ASSOCIATED WITH THE TASK INDICATOR 54

FIGURE 5.1: ARCHITECTURE 58
FIGURE 5.2: STATE MACHINE OF THE TOKENS INDICATOR 59
FIGURE 5.3: STATE MACHINE OF THE HOARD WALK INDICATOR 61
FIGURE 5.4: STATE MACHINE OF THE TASK INDICATOR 61
FIGURE 5.5: RELEVANT VENUS INTERNALS 63
FIGURE 5.6: STATE MAINTAINED BY THE ADVICECONN CLASS 65
FIGURE 5.7: CALLS OF THE API SERVICED BY THE ADVICE MONITOR 66
FIGURE 5.8: CALLS OF THE API SERVICED BY VENUS 67
FIGURE 5.9: LOCATION OF HOOKS SUPPORTING TRANSLUCENT CACHING 68
FIGURE 5.10: INITIALIZATION 70
FIGURE 5.11: NOTIFICATION OF TOKEN EXPIRATION 72
FIGURE 5.12: TRAFFIC DURING A HOARD WALK 73

FIGURE 6.1: REFERENCE LOG FRAGMENT 77
FIGURE 6.2: USER PATIENCE MODEL 84
FIGURE 6.3: DISTRIBUTION OF TASK PRIORITIES 87
FIGURE 6.4: ASSUMPTIONS MADE IN BUILDING KLM MODELS 88
FIGURE 6.5: OPERATORS OF THE KEYSTROKE-LEVEL MODEL 88
FIGURE 6.6: MEASURED SYSTEM RESPONSE TIMES 89
FIGURE 6.7: METHOD #1—DESKTOP WITHOUT POPUP QUERY 90
FIGURE 6.8: METHOD #2—LAPTOP WITHOUT POPUP QUERY 91
FIGURE 6.9: METHOD #3—DESKTOP WITH POPUP QUERY 91
FIGURE 6.10: METHOD #4—LAPTOP WITH POPUP QUERY 91

FIGURE 6.11: SENSITIVITY TO CHANGES IN β AND γ 93

FIGURE 7.1: COMPUTER SCIENTISTS ARE SPECIAL PEOPLE 97
FIGURE 7.2: CORRECTNESS RATINGS 103
FIGURE 7.3: EFFICIENCY RATINGS 105
FIGURE 7.4: SUMMARY OF FINDINGS 107
FIGURE 7.5: RANGE OF READING RATES 110
FIGURE 7.6: SCREEN DIFFICULTY 111
FIGURE 7.7: SCREEN 27 111
FIGURE 7.8: FINDING 17 112
FIGURE 7.9: SCREEN 59 114
FIGURE 7.10: SCREEN 22 114
FIGURE 7.11: SCREEN 23 114
FIGURE 7.12: SCREEN 34 116
FIGURE 7.13: SCREEN 49 116
FIGURE 7.14: COMPARISON OF EXPERT AND NOVICE CONTRIBUTIONS 117
FIGURE 7.15: SCREEN 20 119
FIGURE 7.16: SCREEN 21 119
FIGURE 7.17: SCREEN 60 119
FIGURE 7.18: NUMBER OF INCORRECT RESPONSES 120
FIGURE 7.19: SCREEN 71 120
FIGURE 7.20: SCREEN 72 121
FIGURE 7.21: SCREEN 76 122
FIGURE 7.22: NUMBER OF INEFFICIENT RESPONSES 123
FIGURE 7.23: SCREEN 73 125
FIGURE 7.24: SCREEN 77 125
FIGURE 7.25: SCREEN 78 125
FIGURE 7.26: SCREEN 79 126
FIGURE 7.27: RELATIVE DWELL TIME 127
FIGURE 7.28: EXCERPT FROM E1 TRANSCRIPTS 129
FIGURE 7.29: EXCERPT FROM TABLE SUMMARIZING COMMENTS 131
FIGURE 7.30: COMMENT FREQUENCY 131
FIGURE 7.31: CLASSIFICATION OF FINDINGS 133
FIGURE 7.32: FINDINGS IDENTIFIED BY CLASSIFICATION AND SEGMENT 140
FIGURE 7.33: FINDING ORIGINATIONS 141
FIGURE 7.34: FREQUENCY OF REPORTING 143
FIGURE 7.35: NUMBER USERS REQUIRED TO COVER MOST SEVERE FINDINGS 144
FIGURE 7.36: PROPORTION OF FINDINGS FOUND AND EXPECTED 145

FIGURE 8.1: TAXONOMY OF CACHING FOR AVAILABILITY MECHANISMS 150

FIGURE 9.1: REPAIRING OBJECTS IN CONFLICT 168
FIGURE 9.2: REINTEGRATION INTERFACE PAPER DESIGN 169

FIGURE C.1: DEMOGRAPHICS 254
FIGURE C.2: PROFICIENCY WITH VARIOUS OPERATING SYSTEMS 254
FIGURE C.3: USE OF VARIOUS WINDOWING SYSTEMS 255
FIGURE C.4: EXPERIENCE WITH UNIX-BASED OPERATING SYSTEMS 255
FIGURE C.5: WINDOW MANAGER PREFERENCES 255
FIGURE C.6: EDITOR PROFICIENCIES 256
FIGURE C.7: DOCUMENT PROCESSING PROFICIENCIES 256
FIGURE C.8: DISTRIBUTED FILE SYSTEM EXPERIENCE 257
FIGURE C.9: RELATED EXPERIENCE WITH BRIEFCASE 257

FIGURE C.10: TIMING DATA COLLECTED DURING THE TUTORIAL 259
FIGURE C.11: TIMING DATA COLLECTED DURING THE EXERCISES 260
FIGURE C.12: TIME UNTIL INVESTIGATION BEGINS 261
FIGURE C.13: CORRECTNESS RATINGS 262
FIGURE C.14: EFFICIENCY RATINGS 263
FIGURE C.15: RESULTS FOR THE TIMETODISPLAY METRIC 264

FIGURE E.1: GOMS ANALYSIS OF TUTORIAL SCREENS 362
FIGURE E.2: COUNT OF GOMS OPERATORS PER TUTORIAL SCREEN 363
FIGURE E.3: OPERATORS OF THE KEYSTROKE-LEVEL MODEL 364
FIGURE E.4: RANGE OF READING RATES 365
FIGURE E.5: SCREEN DIFFICULTY ANALYSIS 366
FIGURE E.6: DIFFICULT TUTORIAL SCREENS BY USER 369
FIGURE E.7: EXERCISE ANALYSIS 371
FIGURE E.8: EXERCISE FINDINGS BY USER 373
FIGURE E.9: QUALITATIVE FINDINGS BY USER 376

1 Introduction

Programmers have traditionally exploited caching to improve performance. The power of this technique has been its ability to substantially improve performance in a way that is completely transparent to higher levels of the system. Coda was the first system to exploit caching not only to improve performance but also to improve *availability*. The term availability refers to providing continuous access to data.

Exploiting caching for this purpose fundamentally violates transparency. For example, the system can no longer guarantee its ability to service a cache miss in a reasonable amount of time. Instead, the system must either halt until such time as the miss can be serviced, or return an error indicating that the miss cannot currently be serviced. Neither of these options maintains transparency to higher levels of the system or to the user, and both options reduce the usability of the resulting system.

The goal of this thesis is to explore ways in which caching can be made *translucent* to higher levels of the system by exposing critical aspects of caching to support availability, while hiding noncritical aspects of caching to preserve usability. For example, when the cost of using the network is high, the system might expose cache misses to the user, allowing users control over network usage. In addition, if the system had high-level knowledge of the tasks that users must perform, it could inform them of the availability of those tasks.

Because Coda is a Unix-based file system, the strategies I use must be compatible with Unix idiom and usage custom. One key feature of Unix is the ability for programs to run without user attention [19, 42]. Balancing the desire to retain this feature against the desire to provide better support for attentive users proved to be a major challenge for my research, with sometimes far-reaching consequences. For example, the use of pop-up dialog boxes that block waiting for the user's answer was simply unacceptable. Such out-of-band communication assumes that a human is paying attention to the system and precludes programs that run unattended. My philosophy throughout this research was to maintain the Unix heritage to the greatest extent possible, while adding enough out-of-band communication to improve the usability of the system. When the user is available, the functionality and/or performance of the system is improved. When the user is unavailable or when a program is running unattended, the system must preserve transparency by acting on the best available information.

1.1 The Thesis

Previous research on systems that cache for availability focused almost exclusively on proofs of concept. The outstanding questions were whether it *could* be done. Users of these systems were primarily the developers plus a few of their friends. These users proved not only that availability could be improved, but also that this class of users could make effective use of it. Now, our focus must shift to making these systems accessible to less motivated and less experienced users.

The lack of transparency intrinsic to systems that cache for availability reduces usability in ways that were either unforeseen or underestimated by early designers. To improve the usability of future systems that exploit caching for availability, we must make caching translucent to users. This observation leads directly to the thesis statement:

> Mechanisms that selectively expose internal details of the cache state and operating environment can improve the usability of systems that exploit caching for availability. Further, it is possible to measure the effectiveness of these mechanisms.

Because systems that transparently exploit caching for availability are rare, I examine these mechanisms within the context of the Coda file system. The ideas behind these mechanisms, however, are applicable beyond Coda. As caching for availability becomes a well-understood tool, it is likely to be applied more widely. The Web, whose current consistency model can, at best, be described as primitive and whose current performance is frequently abysmal, may well endeavor to provide high availability in the years to come. Personal Digital Assistants may also strive to provide high availability, as users become less willing to manually manage the increasingly large "cache" of data stored on these systems. This thesis provides a critical opportunity to learn about mechanisms for improving the usability of caching for availability, allowing us to pave the road for future application.

1.2 Background and Historical Perspective

The full effect of caching for availability on the user was not well understood in early versions of Coda. Once Coda had matured sufficiently and had experienced substantial use by many different users with a variety of backgrounds, we realized that we were seeing usability problems with the various tools *as well as* with the violations of the transparency. In this section, I briefly describe the history of Coda, focusing on the influence each advance had on the usability of the system.

In 1991, Kistler introduced *disconnected operation* in the Coda file system [22, 23]. Disconnected operation provided autonomy to clients of a distributed file system by

allowing them to continue using cached data during temporary network or server failures. In summary, Coda exploited its file cache to provide high availability as well as to improve performance. The usability of the system seemed greatly increased because of the new functionality.

Kistler observed that the voluntary disconnection of a laptop computer from a network was equivalent to a total network failure between client and server; thus, disconnected operation also supported portable clients of a distributed file system. To make this mode of operation useful (particularly for disconnection lengths typical of portable clients), the system allowed users to provide hints to the cache manager about important data. These hints helped the system manage the cache in preparation for potential disconnection, helping to ensure that the most useful data is available to the user.

Although disconnected operation provides important functionality, it is not a panacea. Disconnected clients suffer from several serious limitations, each of which reduces the usability of the resulting system:

- Cache misses may impede progress
- Local updates are not visible remotely, and vice versa
- Updates are vulnerable to theft, loss, and damage of the portable computer
- Update conflicts are more likely
- Exhaustion of cache space is a concern

Further, users of the system must provide extensive hints to make effective use of disconnected operation. The system requires that these hints be specified at the level of files and directories. Users quickly learn that application programs access numerous files—files these users never realized existed. New users of the system encounter a steep learning curve before they are able to make effective use of it.

To address a number of these limitations, Mummert extended Coda to exploit *weak* networks [33, 34]. A weak network is one that exhibits high latency, low bandwidth, intermittence, or high cost. When operating weakly connected, Coda services cache misses either in the foreground or in the background depending on the importance of the missing file and the expected service time. It trickles updates to the servers in the background, making updates visible to other clients and lessening both the likelihood of future update conflicts and the cache space required on the client. It also maintains cache coherence by updating the client's cache as other users change files on the servers.

1.3 Lack of Transparency

Weak connectivity ameliorates many of the limitations experienced by disconnected clients, but, like disconnected operation, it is not a panacea. It is fundamentally limited by the available network bandwidth. Further, it cannot obviate the need for disconnected operation because weakly connected clients may well find themselves disconnected—either involuntarily during a network outage, or voluntarily because of concerns regarding

cost, privacy, or the perception that the network is temporarily unusable. It does not address the need for detailed hoard hints and it brings to light several previously unrecognized problems, such as excessive delays, the need for both disconnected and weakly connected operation, and inadequate feedback.

Excessive Delays. Servicing a cache miss of even a moderately large file over a slow network can cause excessive delays. Consider the canonical example of a user accessing a 1 MB file while weakly connected. At 9600 b/s, such a demand fetch would require the user to wait almost fifteen minutes! At 28.8 Kb/s, it would still take just under five minutes. Neither of these delays is tolerable for interactive systems [35]. Although modem speeds are increasing, color picture and movie "illustrations" in documents can increase file sizes substantially beyond the canonical 1 MB. And if the weak network connection is intermittent, the expected delays rise proportionally. The system is fundamentally limited to the available network bandwidth.

Need for Disconnected Operation. Although portable clients may have the ability to operate weakly connected at all times, there remain situations in which the user may choose to operate disconnected. For example, the user may wish to extend battery life by not squandering this limited resource on network transmissions; the user may wish to reduce network charges when costs are high; or, the user may want to avoid network transmissions when the network quality degrades substantially. Thus, even weakly connected clients benefit from the ability to operate totally disconnected from the network.

Inadequate Feedback. Disconnected and weakly connected operation introduce modal behaviors[1], which have long been known to cause usability problems [53]. These modal behaviors are critical to supporting programs that operate unattended. Unfortunately, they introduce a mismatch between the user's expectations and the reality of the current operating conditions. This mismatch contributes substantially to the usability problems we have identified in Coda. For example, until recently, our local network was sufficiently overloaded that network bandwidth regularly fell below the threshold for strong connectivity, causing Coda clients to behave as if they were weakly connected. In this mode of operation, the system delays propagation of updates for a brief period of time to allow future updates to make propagation unnecessary. This delay has confused a number of users because such users were not expecting weakly connected behavior while connected via the building's Ethernet. The standard technique for helping users cope with modal behaviors (when they cannot be avoided) is to introduce feedback to help users recognize the current mode of operation [32].

Disconnected and weakly connected operation increase the range of behaviors exhibited by Coda. These strange and seemingly "abnormal" behaviors occasionally cause substantial confusion and frustration for even experienced users. However, the

[1] A system is said to exhibit *modal behaviors* when the same action (e.g., keystroke or mouse click) can cause different things to happen under different circumstances. For example, when using vi, typing the letter 'i' does one of (at least) two things. If the program is already in "insert mode", then the letter 'i' is inserted into the text of the document. If not, then the program is put into "insert mode" and the letter does not appear in the document.

transparent transitions that cause this confusion among our users are *critical* to supporting programs that have not been extended to adapt to changing conditions. One view of my thesis then is to explain the system to users and to allow them control over its behaviors, while not sacrificing the transparency that is so important to programs.

1.4 Perspective of Users

For the benefit of those readers unfamiliar with the lineage of Coda and for whom my brief description of the problem was perhaps insufficient, let me describe the use of Coda and its predecessors from the user's perspective. To make this description concrete, let me tell a story about two fictitious employees, Peer and Yoni, and their equally fictitious boss, Alex. The topic of the story is their love-hate relationship with a distributed file system.

> Peer, Yoni, and Alex all have desktop machines. Their company has decided to move to using a distributed file system to ease the maintenance problems faced by their facilities staff. Their data is now stored on central file servers and they access it via their local area network. The file servers are maintained and monitored by the company's facilities staff. Their desktop machines interact with these server machines to give them access to their data, caching the data locally to provide improved performance.
>
> Once the initial problems are worked out, the entire company becomes enamored with the new file system. Users like it because they can easily share data with co-workers. In fact, novice users, unwilling to deal with ftp, no longer use sneaker-net[2] to share their data. They simply send a pathname that is accessible from any company computer. The facilities staff likes the system because it greatly reduces the effort necessary to maintain the company computers. They are no longer required to reinstall new versions of software on every single machine. They simply install the new version in the distributed system and let the automatic scripts deposit it on each client machine. Backups have also become much simpler since they need only deal with a small number of server machines and need not backup the client machines at all.
>
> Soon, however, Peer and Yoni learn a few painful lessons about the downside of distributed systems. The first lesson occurred when their network went down the night before a critical deadline. The second occurred when their file server crashed in the middle of an important presentation. The cause was the same in both situations: even though the computer sitting right in front of them was fully operational and had the data cached, they could not access that data. You see, to preserve consistency in the distributed file system, the local client must maintain an active connection to the file server. During periods of server or network failures (when an active connection is not possible), the client cannot allow access to data even if that data is in its local cache because it cannot know whether or not that data is valid. Shortly after those two disasters, Peer and Yoni convince their boss, Alex, to purchase the high availability add-on that will allow them to continue to access their cached data even during periods of server and network failures.
>
> With the new extension, they find that they can access their data even when the server crashes or the network goes down. They are, once again, quite pleased with their system. In fact, it is working so well that Alex decides to invest in laptop computers,

[2] Sneaker-net is defined as the network achieved by writing data to a floppy or tape on one machine, walking it over to another machine, and reading the data on that second machine.

giving them access to their data during their frequent business trips. They soon learn that operating *disconnected* has advantages, but that it is not without pitfalls. Peer's first trip with the laptop was a near disaster. He knew that he needed to provide hints to the cache manager so that it would know what data to keep cached and he carefully did so, but he forgot to authenticate before disconnecting so he did not have permission to write to any of his data during his trip. He was able to read the data, however, so he copied it onto the local disk so that he could work.

Yoni's first trip with his laptop was equally disastrous. He forgot to tell the system about a file critical to starting up his window manager. Without his window manager, he could not present his slides on-line. Luckily, because he was not completely comfortable with the new technology yet, he had printed backup slides and was able to use them for the presentation. After hearing about their experiences, Alex was very careful to test his hints before leaving to make sure that everything worked as expected. His presentation went smoothly and he was very pleased, until he tried to implement some of the suggestions made during his presentation that happened to affect files he had not anticipated needing. He found that the system provided no support for fetching missing files over the internal modem; the process was, thus, tedious and time consuming. He made a note to look into the newly released add-on supporting *weak connectivity*.

Upon returning to the office, he asked Peer to evaluate the weak connectivity support and get back to him. Shortly thereafter, Peer recommended that they purchase it. Alex began using weak connectivity immediately and found it extremely useful. On his next trip, he dialed-in from his hotel room to update the servers with his changes and download any files that Peer and Yoni had modified. Once again, all three of them were in love with the new system and found it greatly increased their ability to share data on the road. They experienced only a few minor annoyances, in that they were never quite sure when the updates had finished transmitting, but nothing major.

Shortly thereafter, Yoni's wife, Leah, accepted a job offer in a different city. Rather than lose such an experienced and valued employee, Alex decided that it would be more productive to allow Yoni to work from a rented office in the new location, accessing his data via an Internet Service Provider (ISP) and taking full advantage of weak connectivity. Yoni typically remained weakly connected to the home office via the Internet. He found that, although performance was usually acceptable, he occasionally had to disconnect because network characteristics degraded beyond usability.

Once Yoni started interacting closely with Peer on a new project, however, he became frustrated with the system. He found that, because the network performance between his ISP and the home company was variable, he was never quite sure whether he was operating weakly connected or strongly connected. He was frustrated because sometimes his updates did not appear on the servers when he thought they should have. In fact, one day, he and Peer wasted a large of amount of time tracking down what appeared to be bugs, but in fact, turned out to be version skew resulting from these consistency problems. They finally remembered that, when the system is operating weakly connected, updates are not propagated to the servers immediately; instead, the system ages them for a few minutes and then ships them over. They have since learned to check the state of their connectivity whenever such confusion arises. After hearing about these problems, Alex suggested that Yoni look into a recently released interface add-on that addresses these problems.

This story summarizes the development of AFS and Coda. It gives a broad overview of how users might use systems like AFS and Coda as well as the problems those users might

encounter. The story ends where my thesis begins, with the promise of an interface to help users of a highly available distributed file system that supports weak connectivity.

1.5 Roadmap of Dissertation

The Coda file system is the research vehicle used to test and to validate the ideas in this thesis. Chapter 2 provides background on the current Coda system. It focuses on disconnected and weakly connected operation, discussing those aspects of Coda relevant to this thesis.

Chapter 3 discusses the design rationale for the translucent caching interface. It begins with a discussion of what aspects of caching need to be exposed to the user. It then continues with a discussion of the potential burdens that translucent caching places on users. It concludes with design principles for limiting those burdens.

Chapter 4 describes the detailed design of the CodaConsole interface, which was inspired by the dashboard of an automobile and features a set of nine indicator lights. The indicators of the interface differ from those of a dashboard because the interface allows the user to expand the indicators to obtain more detailed information. This chapter presents each indicator, describing what it indicates as well as what additional information is available.

Chapter 5 describes the implementation of the interface. It presents an overview of the system architecture. It continues with a discussion of the details of the implementation, including a summary of the Translucent Caching API. It then presents three examples of the use of the Translucent Caching API: an initialization sequence, a simple event notification, and a complex series of event notifications. It concludes with a brief reflection on the implementation.

Chapter 6 examines three techniques used by the interface to reduce the burden translucent caching places on users. The first technique allows users to specify the name of a program and then automatically tracks the references made by that program to identify the files on which that program depends. The second allows the system to offer the user advice regarding her hint specifications. The third allows the system to model the user's patience for waiting for network activity.

Chapter 7 presents the results of a usability test evaluating the interface. The usability test has two primary goals: first, to determine how well users, particularly novice Coda users, are able to learn, operate, and understand the interface; and second, to uncover usability problems in the interface, primarily as a way to focus future work on improving the interface.

Chapter 8 describes work related to this thesis, including work on automatically predicting the files and directories users will use during a disconnected session and work related to exposing the internal workings of other systems to their users.

Chapter 9 concludes the thesis with a discussion of the contributions it makes and suggestions for future research directions.

2 Context and Background

This work was performed within the context of the Coda file system, a distributed file system, whose primary goal is to provide continued access to data during network partitions or server failures. Secondary goals, largely inherited from its predecessor—the Andrew file system (AFS)—include scalability, security, performance, and ease of system administration.

The designers of AFS used a client/server model, partitioning the system into two mutually exclusive sets of machines. The first set, called *servers*, consists of a relatively small number of trusted machines administered by a central authority and dedicated to servicing requests from the second set. The second set, called *clients*, consists of a much larger group of untrusted machines administered and operated by independent groups or individual users. Client and server machines communicate over a local area network via a remote procedure call (RPC) mechanism [5].

Both AFS and Coda make similar usage assumptions. The workload is assumed to be that of an office or engineering environment, including applications such as electronic mail, word processing, program development, and data analysis. Specifically excluded are applications requiring highly concurrent, fine-grained access to data, such as database applications. Further, each client is assumed to be a powerful computer with a moderate amount of disk space. AFS also assumes that each client communicates with servers over a high-bandwidth, low-latency network connection. Coda, however, assumes only that clients will have occasional access to a high-bandwidth, low-latency network connection. In addition, Coda assumes that a single user controls each client.

In the next section, I discuss how these design goals and usage assumptions influenced the basic design of AFS, and therefore, of Coda. The following section continues from the point where Coda diverges from AFS, with an overview of Coda's mechanisms for providing high availability. Finally, I summarize Coda's user interfaces. The discussions in these sections are limited to only those details necessary to understand the work described in this thesis. For further details, please refer to previous AFS and Coda publications [13, 46].

2.1 Influence of AFS

The dominant factor in the design of AFS was the desire to support up to approximately 10,000 clients. In the early 1980s, this was a very ambitious goal—at least an order of magnitude beyond the largest existing distributed file system. Secondary factors in the

design of AFS include security, performance, and ease of administration. In the sections that follow, I examine how each of these goals influenced the design of AFS, and therefore that of Coda. Given the ambitious goal of creating a distributed file system expected to service an entire university, it is not surprising that scalability became the dominant concern in its design. As I review the design, issues of scalability will arise time and again.

2.1.1 Client/Server Model

The client/server model addresses two important issues: system administration and security. First, the relatively small number of server machines makes the scale of the system appear, from the point of view of system administrators, to be significantly smaller than the total number of machines in the system. Second, the integrity of the system as a whole relies primarily on the integrity of the server machines. These machines are secured behind locked doors, are accessible only to system administrators, run only trusted programs, and are closely monitored. The integrity of client machines, however, is not assumed. If a client is subverted, the potential for damage is localized and cannot extend to arbitrary portions of the namespace. Unauthorized release of data is limited to the data cached on the subverted machine and to data accessible to any authenticated user logged onto that machine. Similarly, unauthorized modification of data is limited to that data accessible to an authenticated user on that machine. Authentication tokens automatically expire 25 hours after they were originally acquired, so the timeframe for unauthorized modification is limited.

2.1.2 Volumes

AFS divides the file system hierarchy into manageable pieces, called *volumes*. A volume is a subtree of the file system containing related files and directories, such as a user's home directory. A volume corresponds roughly to a mountable Unix file system. Like disk partitions, volumes are mounted within the file system hierarchy at what is called a *mount point*. Unlike partitions, a volume mounted by one client is visible to other clients. In this way, volumes are glued together to form the complete file system hierarchy. The client recognizes mount points during name resolution and crosses them transparently.

Each file system object, which can be a file, directory, or symbolic link, is identified by a unique *file identifier* (*fid*), a 96-bit quantity consisting of a 32-bit *volume identifier* and a 64-bit *vnode number*. To locate a file system object, the client begins by looking up its volume identifier in a database to determine the servers for the object. It then contacts those servers to fetch the file.

The volume abstraction is crucial for supporting name and location transparency. Since mount points are stored globally in the file system rather than locally at each client (as in NFS), every user has the same view of the file system hierarchy. Furthermore, since file names never contain location information, system administrators are free to move volumes

to different servers or disk partitions to balance the load among the servers. Thus, both users and system administrators benefit from this important abstraction.

2.1.3 Client Caching

Clients cache data from servers in AFS. Thus, when a user accesses an uncached file (i.e., the user encounters a cache miss), the client cache manager, called *Venus*, fetches that file, in its entirety, from the servers. A copy of the file is then stored on the client's local disk. Should the user access the file again in the near future, the request can be serviced from the local disk and the user need not wait for the file to be fetched from the servers again.

To maintain cache coherence, the client obtains a *callback promise* from the server for every object in its cache. If another client updates a file for which a server has promised a callback, the server must issue a *callback break* before completing the update. Thus, a client can confidently allow access to objects for which it holds a callback promise as long as it remains in contact with the server making the promise.

The design of this caching scheme has important implications for the overall goals of AFS. Caching improves the performance of the client since there are substantial amounts of locality of reference in the anticipated workloads. Caching also improves scalability since it reduces the amount of server load imposed by each client, thus allowing more clients to be supported by each server.

2.1.4 User Authentication

Every AFS user has an AFS identity and password. To access data stored in AFS, the user authenticates to an AFS authentication server, providing a username and password [44]. If the user provides the correct password, the server issues *tokens* to Venus on the user's behalf. Venus then stores these tokens, allowing it to establish secure connections to the file servers. Tokens are valid for a period of 25 hours, after which the user must reauthenticate. Users may view their tokens with a special program.

2.2 Extensions for High Availability

AFS had a strong influence in the design of Coda. In fact, the previous section could have been written to describe Coda. Coda now diverges from AFS, however, as it strives to provide high data availability.

High availability becomes the dominant goal. Scalability, though still important, assumes a secondary role to high availability. Certain assumptions also change slightly. Clients are now assumed to have only the *ability* to connect to a high-bandwidth, low-latency network at least occasionally. At other times, clients may be disconnected from the

network or connected via a low-bandwidth network. Clients are further assumed to be controlled by a primary user; in fact, certain activities are limited to that user.

Coda provides high availability through the use of three complementary mechanisms. The first mechanism, *server replication*, provides resiliency in the face of server failures and network partitions. The second mechanism, *disconnected operation*, maintains the illusion of connectivity when a client is completely isolated from the servers. The third mechanism, *weakly connected operation*, alleviates some of the shortcomings of disconnected operation when only a low-bandwidth network connection is available.

2.2.1 Server Replication

Coda uses *optimistic replication* at the volume level to provide resiliency to server failures and network partitions [28, 46]. With optimistic replication, updates to file objects are allowed provided that at least one replica is visible to the client performing the update. Thus, conflicting updates (caused by write sharing) can and do occur. Coda detects such updates, automatically resolves most types of conflicts, and provides tools to help users manually resolve the remaining ones. Fortunately, in an office and engineering environment, write sharing is a relatively infrequent occurrence. This thesis depends minimally on Coda's replication mechanism so I will not delve into further details on this topic; instead, interested readers can refer to Kumar's thesis [28].

2.2.2 Disconnected Operation

Disconnected operation provides resiliency to failures that completely isolate a client from all Coda servers. Because of this resiliency, Coda supports mobile clients [22]. When operating disconnected, clients act as pseudo-servers and allow the user to work entirely out of the client cache. The trick to making a disconnected session successful, then, is to make sure that all necessary file objects are cached. Coda uses two techniques to prepare for voluntary as well as involuntary disconnection. These two techniques are *prioritized cache management* and *user-assisted hoarding*.

In the version of Coda described by Kistler [22, 23], each volume cached by Venus is in one of two stable states (*connected* or *disconnected*) or in a transitory state (*reintegrating*). While connected, operations are serviced as in AFS with minor differences caused by the details of replication. While disconnected, operations are serviced as long as the object is cached locally. All mutations to a volume are logged in the volume's *client modify log* (CML). Upon reconnection, the entries in the CML are replayed to the servers, in a process called *reintegration*. If reintegration is successful, the volume transitions to the connected state and nothing further need be done. Otherwise, the problematic operations are excised from the log, marked for user attention and the volume then transitions to the connected state. Obviously, this description has glossed over many important details regarding the system's activities in each of these states. For

the most part, these details are irrelevant for the purposes of this thesis. One exception, however, is what happens in preparation for disconnected operation.

The key to maintaining the illusion of connectivity is ensuring that all file system objects the user will access while disconnected are cached prior to disconnecting from the network. This is a tricky, but critical task. Coda uses prioritized cache management together with user-assisted hoarding to enable the user to specify what objects are likely to be accessed. Prioritized cache management allows Venus to combine explicit information (optionally provided by the user) with implicit information (gathered by Venus) to manage the content of its file cache.

The user's role in this system is to assign priorities to file system objects[3]. Venus stores these user-assigned priorities, called *hoard priorities*, in the Hoard Database (HDB). User-assigned priorities range from 0 to 1000, with higher values representing greater importance. Venus determines an object's *overall priority* using a linear combination of the object's hoard priority and the object's *reference priority* [22]. The reference priority, implicitly set by Venus, measures how recently the object has been accessed. Symbolically,

$$p(x) = \alpha \cdot h(x) + (1 - \alpha) \cdot r(x)$$

where $p(x)$ is the object's overall priority, $h(x)$ is the object's hoard priority, $r(x)$ is the object's reference priority, and α is a constant called the *horizon parameter*. When space is required to fulfill a demand request, Venus evicts the object with the lowest overall priority.

The horizon parameter, which ranges from zero to one, balances the hoard priority and the reference priority. Values closer to zero favor recently referenced objects and force the cache to rely upon locality of reference to provide data during failures. Values closer to one favor objects explicitly mentioned by the user as being important during disconnected operation, and force the cache to rely upon user-supplied information to provide data during failures. Currently, the horizon parameter is set to 0.75.

Venus' goal is to maintain *cache equilibrium*, by ensuring that the cache contains only the highest priority objects at all times. Maintaining a perfect equilibrium, however, would be impractical because it would cause unacceptable overhead and could potentially suspend user activity for long periods of time. Instead, Coda takes a periodic approach to maintaining cache equilibrium. Every 10 minutes, Venus performs a *hoard walk*. Users may also initiate hoard walks, and typically do so immediately before voluntarily disconnecting from the network.

During a hoard walk, Venus equilibrates the contents of the cache, ensuring that the highest priority objects are available. Hoard walks occur in two phases. During the first phase (called the *status walk*), Venus ensures that the highest priority objects have a valid

[3] The user specifies this information to better prepare for disconnected and weakly connected operation. The information is optional. If provided, it need not be complete. In the absence of information, the object is managed using only the implicit information.

status block cached. During the second phase (called the *data walk*), Venus fetches the data for those same objects.

When all the pieces fit together, users are blissfully ignorant of whether they are connected or disconnected. Unfortunately, the illusion crumbles the moment the user attempts to access an object that was not anticipated. The request results in an ETIMEDOUT error message, and the user becomes painfully aware of the fragile nature of disconnected operation.

2.2.3 Weakly Connected Operation

Mummert extended Coda to allow clients to operate over *weak* network connections [33, 34]. She defined a weak network connection to be one with high latency, low bandwidth, or high costs. Thus, a client operating over a weak network is said to be operating *weakly connected*, whereas one operating over a network not characterized by high latency, low bandwidth, or high cost is said to be operating *strongly connected*. Many aspects of the system had to be rethought to support weakly connected clients.

Of these, the aspect most relevant to my work is reintegration. Originally, reintegration was a transitory state used to transition from the disconnected state to the connected one. The system performed this transition by replaying the mutations stored in the CML. The problem is that it replayed the entire CML. If the CML was large, this process could dominate the weak network for a substantial period of time, blocking all other network activity. Although this design worked well for clients reconnecting to a strong network, its performance for clients connected via a weak network was abysmal. To correct this problem, Mummert introduced the idea of reintegrating the CML in pieces proportional to network bandwidth. She called this process *trickle reintegration*.

The implementation of trickle reintegration required her to transform the transitory reintegrating state into a stable state, called the weakly connected state. The new volume state diagram is shown in Figure 2.1. A volume in the weakly connected state services read requests as if it were connected, but logs mutating operations to the CML as if it were disconnected. Mutations remain there until they have *aged* a reasonable amount of time. Aging each mutation allows the system to continue to take advantage of the proven benefits of *log optimizations*. Log optimizations reduce the size of the CML substantially, thus conserving disk space and reducing the amount of data that must be transmitted to and processed by the file servers [23]. Once a set of operations has aged sufficiently, it is considered ripe for reintegration. The reintegrating process locks a subset, or *chunk*, of these ripe operations to prevent ongoing mutations from optimizing them out from under the reintegrating process. The size of this set depends upon the current network bandwidth and is determined by calculating how many updates can be transferred within a time limit imposed by Venus. If the reintegration succeeds, the operations in the set are removed from the log. If it fails, the operations are unlocked and any pending optimizations are processed. Between reintegrating these chunks, the reintegration process defers to high priority network traffic. In this way, updates are "trickled" back to the servers.

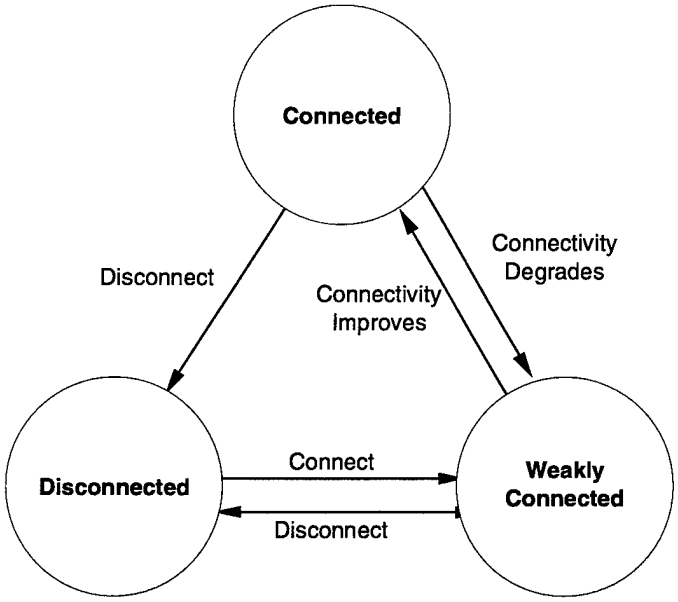


Figure 2.1: Venus States and Transitions
This figure illustrates Venus volume states and transitions as described by Mummert [33].

A second aspect that required modification and is relevant to this work is the way in which Venus validates its cache after a disconnection. Disconnected operation encourages users to increase the size of their cache (limited only by the amount of free space on the disk) and fill that cache with hoarded data. Upon reconnecting to the network, Venus validates that entire cache, one object at a time. Over an Ethernet, this process was not noticeably painful, but over a modem, it could take up to 20 minutes! To address this problem, Mummert introduced *volume callbacks*. Volume callbacks increase the granularity at which clients track server state. Each volume now has a version stamp that the client can store and compare with the current version stamp on the servers. If those stamps are equal, the client knows that all objects from that volume are valid. If the stamps are unequal, the client must resort to validating individual objects within a volume. These large granularity callbacks allow Venus to validate a large subset of the name space with a single RPC almost 97% of the time, substantially reducing the costs associated with validating the entire contents of even a large cache following a network partition [33].

2.3 User Interfaces

To judge the contributions of this thesis, it is important to understand the interfaces available to users of Coda. Coda users are provided with a tool, `repair`, to help them manually repair files that have experienced conflicting updates. Users of disconnected operation are provided with two additional programs. The first, `hoard`, enables the user to specify the hoard priorities of file system objects. The second, `spy`, assists the user in identifying objects to which she should assign priorities.

2.3.1 The Repair Tool

During a network partition, an object may be updated in ways that cannot be automatically resolved by the system. In such situations, the user must manually repair conflicts using a special tool, called `repair`. This program offers a command-line interface that diagnoses the problem, generates suggested actions to repair the object's replicas, and, after confirmation, issues the commands to fix the object in question.

2.3.2 The Hoard Program

The special tool with which users provide hoard information to Venus is called the `hoard` program. It provides a simple front end to the HDB interface [22]. This interface allows entries to be added to or deleted from the HDB. In addition, it allows the contents of the HDB to be listed, cleared, and walked. The command set for the `hoard` program directly reflects this interface and is shown in Figure 2.2.

Figure 2.3 illustrates some typical input to the `hoard` program. These examples are taken from my own hoard profiles. Each line of input consists of a command, a pathname, a hoard priority, and possibly meta-information. In each of the above examples, the command was `add` (abbreviated by the 'a'). The number, which follows the pathname, is the hoard priority I have assigned this object. Optional meta-information, such as "`:c`" and "`:d+`", can be added to input lines whose pathnames specify a directory. 'c' indicates that the current children of this directory should be hoarded, while 'd' indicates that the current descendants of this directory should be hoarded. Adding a '+' indicates that *future* children or descendants should also be hoarded. In essence, the hoard program provides an assembly language for hoarding.

2.3.3 The Spy Tool

In addition to the `hoard` program, Coda provides a tool, called `spy`, which enables users to observe all Coda file references. The tool works with the cooperation of Venus. It connects to Venus over a socket and Venus informs it of every file open event that arrives from the kernel. These open events are printed to standard output. The tool makes no attempt to separate open events based upon the process or user generating them.

This tool has proven particularly helpful in tracking down file references made by other programs. One example of a program that requires this sort of tool is `latex`, a document formatting utility. When processing a document, `latex` accesses other executable programs as well as many different fonts and library files. Some of these file references are relatively easy to track down because they are explicitly referenced in the document source file. Others, however, are used as auxiliary files and are not specifically mentioned in the document source file. These are the files that require a `spy`-like tool. Figure 2.4 shows the output produced while spying on the LaTex program.

Command	Function
add filename [priority] [:{c|d}[+]]	Add `filename` to the database with optionally specified `priority` and meta-expansion attributes ('c' and 'd' indicate children and descendant expansion respectively, with '+' indicating the dynamic rather than static variant).
clear [uid]	Clear all database entries or those for specified `uid`.
delete filename	Delete database entry with specified `filename`.
list filename [uid]	List all database entries or those for specified `uid` to specified output file.
walk	Perform an immediate hoard walk.

Figure 2.2: Hoard Program Command Set

This table describes the parameters associated with each hoard command. This table is provided courtesy of Jay Kistler [22].

```
# Hoarding individual programs
a /usr/local/tex/bin/latex 50
a /usr/local/tex/bin/xdvi 50
a /usr/local/tex/bin/bibtex 40

# Hoarding project files
a /coda/project/synrgen 20:c+
a /coda/project/synrgen/cfg 20:c+
a /coda/project/synrgen/doc 20:c
a /coda/project/synrgen/lib 20:c+
a /coda/project/synrgen/include 20:c+
a /coda/project/synrgen/man 20:c+
a /coda/project/synrgen/src 20:c+

# Hoarding private files
a /coda/usr/mre/thesis 100:d+
```

Figure 2.3: Sample Hoard Profile

These lines were taken from three of my personal hoard profiles, `tex.hdb`, `synrgen.hdb`, and `personal.hdb`. In the first set, I hoard individual programs associated with formatting my documents. The second set hoards specific directories inside the SynRGen project directory. In each case, the directory as well as its current and future children are hoarded. The third set hoards my thesis directory and all of its current and future descendants.

```
<7f0000f1, ./i386_mach/omega/.COMMON/bin/latex>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/bin/pstex>, 2660, 50
<7f0000f1, ./i386_mach/omega/bin/virtex>, 2660, 50
<7f0000f1, ./i386_mach/omega/bin/xdvi>, 2660, 50
<7f0000f1, ./i386_mach/omega/bin/bibtex>, 2660, 40
<7f0000f1, ./i386_mach/omega/lib/formats/psplain.fmt>, 2660, 50
<7f0000f1, ./i386_mach/omega/lib/formats/lplain.fmt>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/macros/article.sty>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/macros/art10.sty>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/macros/psfonts.sty>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/macros/times.sty>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/macros/plain.bst>, 2660, 40
<7f0000f1, ./i386_mach/omega/.COMMON/lib/fonts/times.tfm>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/fonts/timesb.tfm>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/fonts/timesit.tfm>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/fonts/courier.tfm>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/pk/lw/common/pk300>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/pk/lw/common/pk300/cmbsy10.300pk>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/pk/lw/common/pk300/circlew10.300pk>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/pk/lw/common/pk300/cmr10.746pk>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/pk/lw/common/pk300/lasy6.300pk>, 2660, 50
<7f0000f1, ./i386_mach/omega/.COMMON/lib/pk/lw/common/pk300/cmmib10.329pk>, 2660, 50
                                        .
                                        .
                                        .
```

Figure 2.4: Sample Spy Output

The figure shows output typical of the spy program. For each Coda object referenced, spy shows the volume in which that object resides (e.g., 7f0000f1) as well as a volume relative pathname (e.g., ./i386_mach/omega/.COMMON/bin/latex). In addition, it shows the id of the user who has hoarded the object and the hoard priority it has been assigned.

2.4 Summary

The Coda file system has been greatly influenced by its AFS heritage. Many of the underlying assumptions and much of the basic design are the same. Coda's primary goal, however, differs. Rather than focusing on issues of scale, Coda focuses on high availability. It offers continued access to data during periods of complete isolation from servers as well as during periods of weak connectivity with servers. Coda assumes that users are willing to help the system prepare for these periods of poor connectivity and that users are willing to accept the responsibility for fixing any conflicting updates that occur. It provides basic tools to the user for performing these tasks.

3 Design Rationale

Before designing an interface that makes caching translucent, I must address the question of which aspects should be exposed to the user. Issues to consider include:

- What errors must be brought to the user's attention?
- What system behaviors cause confusion and why?
- What aspects of the system do users need and want control over?

I begin this chapter by discussing these questions and presenting aspects of caching that should be exposed to the user.

In sketching out a preliminary design of an interface to make caching translucent, I quickly realized that doing so could actually endanger the usability of the resulting system in significant ways. Although users may be less confused about the inner workings of the file system, they may spend so much time paying attention to those activities that they are unable to use the file system effectively. Thus, I continue this chapter with a discussion of the burdens imposed by translucent caching and then present five design principles intended to minimize those burdens. An overview of the activities and events that could be exposed to the user completes the chapter.

3.1 Windows of Translucence

In this section, I discuss those aspects of caching I considered exposing to users. I examine how to present errors to the user, how to reduce confusion caused by modal behaviors, and what aspects of the system users need to control.

3.1.1 Presenting Error Conditions

Contemporary computer systems, as a whole, fail to present errors to users in useful ways. What user cannot tell a story about a horrible error message they have encountered? In the process of writing this thesis, I have certainly run into more than my fair share. Although one might expect these sorts of errors from a Unix-based system, the error messages to which I am referring all came from supposedly user-friendly systems like Windows95 and Office97.

PowerPoint helpfully informed me that it could not print my document because either it had run out of memory or the printer was not on the network. Thankfully, I was disconnected from the network and had accidentally clicked the print button so it was an easy error for me to diagnose. However, in another situation, it could have been far more confusing. In all likelihood, the printing software received an error and handed that error off to the user rather than investigating the problem itself so that it could present the error with some potential solutions to the user.

Similarly, Microsoft Backup does not bother to determine whether the cause of its inability to write to the selected backup device was that (a) the device was write protected, (b) the network restricted access rights, (c) the device was full, or (d) the device name was not valid. Once again, the interface dropped the ball by not investigating the cause of the failure more thoroughly.

Experience has shown that, like these systems, neither AFS nor Coda present errors to users well. One user complained about AFS' habit of silently expiring tokens, making little effort to inform the user and allowing her to figure it out based upon seemingly unrelated symptoms of the problem. One very experienced Coda user reported a confusing behavior he had experienced recently. At the bottom of his bug report, he writes: "Eventually, by dumb luck, I realized that I had forgotten to clog [authenticate]. Boom, everything's fine, resolve works, I'm back in business. I think a better error message here would be nice" [post to cmu.cs.proj.coda.bugs on 17 July 97]. Clearly, the lack of tokens was not evident to either of these users. In this particular instance, an error reported to the requesting process may have been the right thing to do. In other situations, however, there may be no user process to which an error can be returned. Although both of these examples relate to token expiration, other less frequent errors could cause similar confusion. A successful interface for translucent caching will present error reports with appropriate amounts of detail to its users.

3.1.2 Rationalizing Behaviors

Coda users also experience confusion caused by fundamentally modal behaviors. Coda attempts to hide vagaries of network performance by quietly changing its behavior. When the system detects that network bandwidth is low, it automatically switches to weakly connected operation. When it detects that network connectivity has disappeared, it automatically switches to disconnected operation. By attempting to continue to operate even while network performance is substandard, Coda increases the availability of data and thereby increases usability. But because these transitions are hidden from the user, Coda decreases usability. Even Coda developers have expressed confusion over these silent transitions!

The zephyr logs[4] captured an excellent example of this confusion. A Coda developer and expert user wrote: "Help, why would venus [sic] claim a timeout on a directory when all

[4] Zephyr is an on-line chat system used by many members of the department to communicate. The Coda project uses a particular zephyr instance that all developers read at least occasionally.

of the servers are up? Further, you can ls -ld the directory, but not ls -l or ls..." [Coda zephyr instance, March 1996]. After chatting with him, it became clear that what had happened was that Venus had silently transitioned into weakly connected operation.[5] Because the directory in question could not be fetched within a certain time limit, Venus returned an ETIMEDOUT message. Because the user had not noticed the transition to weakly connected operation, he was baffled by the seemingly strange behavior he observed.

A second example, also captured in the zephyr logs, occurred in August 1997. Another Coda developer wrote to report strange behavior. Venus had inexplicably timed out during a compile into his Coda object area and had done so, albeit unpredictably, in the past. This time he had captured a Venus log file and wanted me to look at it. While we were trying to sort out this first strange behavior, we discovered that the log file, which he had saved into Coda, had zero length. These behaviors suggested that his Venus was operating weakly connected. Indeed, upon further examination, he determined that Venus had, unbeknownst to him, transitioned into weakly connected operation.

This incident, and similar ones not captured by zephyr, results directly from transparent transitions between different modes of operation. Modal behaviors were known to cause usability problems as early as 1973 [53] and the best solution, obviously, is to avoid them. However, because Coda provides these modal behaviors as a *feature* to support legacy programs that know nothing about adapting to changing network conditions, it cannot avoid them. Instead, Coda must take steps to reduce confusion caused by the behaviors. Research suggests that exposing the modes and their transitions to the user will help. Providing such feedback in other interfaces has reduced mode errors by as much as 70% [32]. Our interface for translucent caching should strive to make users aware of the current mode of operation and notify users when a transition between modes occurs.

3.1.3 Controlling Resources

The three most precious resources to the mobile client are cache space, network bandwidth, and battery power. Since issues of power resources are only indirectly related to file systems, I do not discuss them further. Instead, I focus my attention on cache and network resources.

3.1.3.1 Cache Space

During periods of disconnected operation, Venus exploits the cache to provide availability; during periods of weakly connected operation, Venus exploits it to avoid network-consuming demand fetches. Because the contents of the cache are so crucial to the success of these modes of operation, the current system exposes caching decisions to the

[5] The user was running a version of Venus that contained support for weakly connected operation. Because the network is overloaded, bandwidth frequently drops below the threshold that triggers weak connectivity.

user by allowing the user to prioritize file system objects. Venus then uses these priorities to make caching decisions (see Chapter 2 for details).

The current system offers the user 1000 different priority levels and requires her to prioritize individual files and directories. Users currently have difficulty determining all the files and directories they need to hoard. In fact, even expert users fail to hoard critical files occasionally. Users also have difficulty determining whether or not their hoard contents are available for use during an upcoming disconnected session. Thus, they tend to practice disconnecting one or more times prior to leaving their office.

Because preparing the cache for periods of disconnected and weakly connected operation continues to be an important duty, users continue to need a way to specify which file system objects are important. Further, they need feedback regarding which objects are currently available and which ones are not. An interface that makes caching translucent to the user needs to reduce the burden hoarding places on users and increase the feedback provided to users.

3.1.3.2 Network Bandwidth

Venus uses network resources to satisfy four important requirements, each vital to the integrity of the file system. Venus must:

1. Maintain cache consistency
2. Service fetch requests
3. Maintain cache equilibrium
4. Replay updates

Each of these activities is critical to the file system's integrity, but their relative importance to the user may vary over time. In this section, I explore each of these requirements.

3.1.3.2.1 Maintaining Cache Consistency

Venus' first responsibility is to maintain cache consistency. It does so by maintaining callbacks while strongly or weakly connected and by pure optimism while disconnected. While connected, the network costs associated with maintaining cache consistency are minimal. When a callback is lost, Venus simply invalidates the object or objects involved. While a callback is held, Venus must periodically verify that the server that granted the callback is up and able to maintain its promise. If Venus loses contact with the server, it must assume that the server is unable to maintain its callback promise; thus, the consistency of the object is suspect, but the object is not invalidated. Upon reconnection, Venus must validate its entire cache by reestablishing callbacks on all its objects. Because volume callbacks make the cost of validating the entire contents of even a large cache minimal (see Section 2.2.3), the expected *benefit* of exposing cache validation to the user is not high. For this reason, I chose not to explore this option.

3.1.3.2.2 Servicing Fetch Requests

One of Venus' most noticeable responsibilities is servicing fetch requests made on behalf of the user. The network costs associated with a request depend upon the size of the requested object and the speed of the network. Let us consider a simple example. Suppose the user has requested a 1 MB file. On a 10 Mb/s Ethernet, this file can be fetched in a second or two. Yet on a 14.4 KB/s modem, this file requires nearly 10 minutes to retrieve.

When considering this problem, I must remember three important points. First, a 1 MB file is not particularly large, especially with the increasing importance of audio and video. Second, because of the nature of a demand request, the user is most likely sitting around twiddling her thumbs until the fetch has completed. Finally, the user may have referenced the file indirectly or even accidentally. For example, the missing file could be a large postscript figure that is included in an area of a document the user is not currently working on. Rather than waiting for the figure to be fetched, the user may prefer to exclude the figure from the document temporarily. Because of the potential for excessively lengthy delays while Venus services demand requests, the user needs to have control over whether or not an object is fetched over a weak network connection.

3.1.3.2.3 Equilibrating the Cache

The third responsibility of Venus is to maintain cache equilibrium. Venus must balance the user's demand requests with the need to prepare the cache for an unexpected disconnection. The hoard daemon reequilibrates the cache periodically (during hoard walks) by recalculating the priority of each cached object and comparing those priorities to hoarded, but uncached, objects. This process ensures that the highest priority objects are cached.

The network costs associated with performing such a hoard walk can be quite substantial. Recall that a hoard walk occurs in two distinct phases. During the first phase (the status walk), Venus validates the status block of each cached object, refetching the status if necessary. Since each status block is only about 100 bytes and since most cached objects are valid, this phase is relatively inexpensive. The second phase, however, is a different matter. In this phase, Venus fetches data for each invalid or missing object identified in the first phase. For each of these objects, Venus could incur substantial network costs. These costs are directly proportional to the cumulative size of the file system objects that must be fetched. Because of the potential for extremely high costs associated with equilibrating the cache, the user needs to have control over the amount of data that will be fetched.

3.1.3.2.4 Reintegrating Updates

The final responsibility of Venus relevant to this discussion is the replaying of updates that have occurred during either disconnected or weakly connected operation. Objects may be reintegrated *in toto* or in chunks (as with trickle reintegration). Almost all types of updates can be replayed with only minimal network resources. The one exception is the

`store` request, which requires that the file contents be sent to the servers. As with servicing fetch requests, the time required to transfer the contents of these files depends upon both the size of the file and the current network bandwidth. Again, these costs can be substantial for large files or slow links. Because weak connectivity only offers a limited amount of bandwidth, the user may prefer to delay propagation of updates to allow demand fetches or cache equilibration to have full use of those resources. Thus, users need to have control over whether reintegration occurs and, if so, how much.

3.2 Burdens of Translucence

The question I consider in this section is that of identifying threats to usability—in essence, the risks inherent in exposing aspects of caching to the user. The primary threat to usability for a mobile client of a distributed file system comes from critical cache misses—that is, objects that are critical to the user's work, but that are missing from the cache. Other threats result from the demands the system places on a user's valuable time and relatively limited cognitive capacity. In this section, I will discuss each of these potential threats.

3.2.1 Failure of Availability

A critical cache miss is the ultimate "sin" committed by the cache manager because it interrupts the user's train of thought and forces the user to change tasks or work around the missing object. A goal of this thesis is to reduce the likelihood of critical cache misses in ways that do not compromise the overall usability of the system.

3.2.2 Excessive Time Investment

The second threat to usability is the risk of requiring too much time to prepare for operating away from the office. The system requires time from the user in several distinct ways, including

- Learning to use the system and its interface
- Waiting for the system to service a variety of common requests
- Offering advice to the system to help it better provide high availability

User-assisted solutions to the high availability problem must remain cognizant of the time investment required of the user.

3.2.3 Overloading the User's Cognitive Capacity

The final threat to usability that I consider is the risk of overloading the user's cognitive capacity. This includes activities that require the user to remember more than short-term memory allows, that break concentration, or that demand immediate attention. Each of these "sins" centers the user's attention not on her own activities, but rather on the job of controlling the cache manager.

3.3 Principles of Translucence

A number of principles guided the design of the interface. In this section, I present each of these principles. At times, one principle contradicts another. The job of the interface designer then is to juggle these competing goals gracefully.

Principle #1: Above all, do no harm

The first principle summarizes the words of Hippocrates:

> The physician must ... have two special objects in view with regard to disease, namely, to do good or to do no harm [12].

Rephrasing these apropos words, the designer must have two goals in developing interfaces, namely, to improve the lot of users, or at least not make it worse.

Principle #2: Work within the Unix model

Coda offers Unix transparency to applications accessing it. The system must remain consistent with this Unix model. As such, if the system can perform better with user assistance, it should be allowed to take advantage of that assistance. However, if no user is available to offer assistance, the system must continue on to the best of its ability. This requirement distinguishes my work from work in which designers freely assume the user is attending to the interaction.

Principle #3: Follow generally accepted principles of HCI designers

Interface designers need to follow generally accepted guidelines for building good human-computer interfaces. Although many guideline documents contain hundreds of pages, a few general heuristics provides more guidance in the "real world" simply because designers can remember them. These heuristics [31, 37] are:

1. *Simple and natural dialogue*: Minimize the information presented to that which is absolutely necessary and present that information logically.
2. *Speak the user's language*: Present information in words, phrases and concepts familiar to the user.
3. *Minimize the user's memory load*: Make information available when and where it will be used.
4. *Consistency*: Represent different words, situations, and actions differently.
5. *Feedback*: Keep users informed of the system's activities.
6. *Clearly marked exits*: Mark exits clearly so that users can easily back out of unwanted states.
7. *Shortcuts*: Offer accelerators to speed the interaction for the expert user.
8. *Good error messages*: Present error messages using plain language, indicate the problem precisely, and suggest possible solutions.
9. *Prevent errors*: Avoid error states whenever possible.
10. *Help and documentation*: Offer search capability, focus on the user's task, list necessary actions explicitly, and avoid verbosity.

Principle #4: Be unobtrusive

The third principle, a corollary of Principle #1, is simply to be as unobtrusive as possible. The user's goal is to write a paper or to implement an algorithm—not to babysit the cache manager. The system should alert the user to specific, important events and should allow the user to influence those events (when reasonable). The system must not demand immediate attention, must not be annoying in its interactions, and it must not "cry wolf". If the system fails in these goals, it risks not serving its users well.

Principle #5: Balance Costs with Benefits

By definition, a user-assisted approach to the problem of providing high availability requires that the user spend time on meta-level issues that would otherwise be spent on her primary goals. The user is willing to invest that time for the ability to access her data while mobile. Economics suggest that for this approach to be successful, the benefits must outweigh the costs. Otherwise, users will not waste their time. This principle serves to remind system designers of this fundamental law.

3.4 Overview of Translucent Caching

In this section, I provide an overview of which aspects of the system I expose to the user and how I reduce the costs of that translucence. I begin by discussing how to provide the user with feedback regarding both error conditions and modal behaviors. I then continue by discussing how to give users control over critical system resources—in particular, cache and network resources.

3.4.1 Feedback

The first design goal for the translucent caching interface is to provide the user with feedback about the system, in a way that does not interfere with the user's primary task. Users need two types of feedback. First, they need to be aware of error conditions (see Section 3.1.1). Second, they need to be aware of modal behaviors (see Section 3.1.2).

The distinction between error conditions and modal behaviors in Coda is blurred. Coda's goal is to hide error conditions, such as the lack of a network connection to the server, by transitioning to a new mode of behavior. These two aspects of Coda are very closely related. When an error condition causes a change in behavior, I consider that error a modal behavior rather than a true error.

One of the most common error conditions users experience is the expiration of their Coda tokens. When this happens, the system reduces the user's privileges to those of an unauthenticated user. In essence, the system has entered a new mode of behavior without reflecting that state in any obvious way to the user. In contrast, the translucent caching interface must provide the user with an indication of the current mode. This indication should distinguish between token expiration and activity pending tokens.

Other modes of operation that the interface must indicate to users relate to network connectivity. The interface must make users aware of the current mode of operation: strongly connected, weakly connected, or disconnected. If users are aware of the current mode of operation, their expectations for system behavior will change in subtle, but important, ways.

One class of true errors is related to storage allocation. If the system's local disk or Venus' RVM [47] space fill to capacity, Venus is likely to terminate ungracefully. When these space areas begin to get too full, the system needs to warn the user. When they become dangerously full, the system needs to warn the user more urgently. Similarly, if Venus' file cache becomes filled with hoarded objects, the user needs to be informed about the situation so that she can take measures to correct it.

This feedback, although important to usability, must not become intrusive or I risk making the system less usable than before. The system must make the user aware of system state without getting in the way.

3.4.2 Control over Critical Resources

The second design goal is to provide users with control over critical system resources. To reduce the burden this control places on users, the interface only exposes those aspects of resource management for which users need control. The two most critical resources users need control over are the cache and the network. In this section, I discuss the limited amount of control the interface offers to its users.

3.4.2.1 Cache Management Decisions

Coda users already have control over cache management decisions. The purpose of giving users this control is to allow them to ensure that their work is available in the case of a disconnection. Although the current system offers users this control, it does so in a way that compromises the usability of the system, particularly for novice Coda users. In essence, there is a mismatch between the information that the system needs and the information that the user can provide and vice versa.

The system compromises usability in two important ways. First, it does not speak the user's language. Users must specify their work at the level of files and directories, but often these files and directories are unfamiliar to the user because they are "owned" by a program that the user wants to use. For example, if the user wants access to the `latex` program, not only must they specify the `latex` binary but they must also specify the `virtex` binary as well as the necessary font and style files. Because users must supply hoard information at the level of individual files and directories, hoarding places an undue burden on them. Second, users are never quite sure whether or not their work is available. Because the cost of a missed file is substantial, this lack of confidence encourages users to perform multiple practice disconnections before actually leaving the office. To make caching translucent to the user while imposing only a minimal burden, I need to change both the granularity at which users provide hoard information and the way in which Venus communicates with users.

The interface addresses this issue by allowing users to define *tasks* and by providing feedback regarding the *availability* of those tasks for use during disconnected operation. To reduce the burden of hoarding, the system also automates a portion of the task definition procedure. A *task* contains user data and programs, as well as other tasks. User data is specified at the level of files and directories, because it is reasonable to assume that users are familiar with it. Programs, on the other hand, are specified at the level of executables. The system automates the process of defining programs by monitoring what file system objects a program accesses. Task-based hoarding is discussed in Sections 4.10 and 6.1.

Defining tasks is likely to be an error prone process. To help the user debug their definitions, the system must offer the user advice. This advice includes identifying objects the user should be hoarding but is not, as well as identifying objects the user is hoarding but should not be. By providing such advice, the interface further reduces the burden hoarding places on the user, thus reducing the costs and increasing the benefits.

Finally, the interface uses two techniques to increase user confidence regarding the ability to operate disconnected. The first, which has already been discussed, is to provide the user with feedback about the availability of tasks. The second, which eliminates the need to perform practice disconnections, is to provide a mode of operation that makes cache misses apparent to the user. While operating in this mode, the user can observe cache misses without having to physically disconnect from the network and without having to wait for the requisite timeout period. By eliminating the need for practice disconnections, I reduce the burden of preparing for disconnected operation.

3.4.2.2 Network Management Decisions

Weakly connected operation introduces a number of decisions that are very difficult to make automatically. In some situations, the user might want the system to do one thing; in other situations, the user might prefer the system to do the opposite. Because the user's preferences are not constant, heuristics will simply not work well in all situations.[6] To solve this problem, the interface makes certain network decisions translucent to the user. When the system needs to make one of these decisions, it issues an *advice request* to the user.

The first type of decision I expose to the user is whether or not to fetch a file over a slow network connection. Depending upon the size of the file and the speed of the network, a demand fetch could cause a substantial pause in processing. Depending upon the importance of the file to the user's work and the expected fetch time, the user may or may not want the system to initiate the fetch. My interface offers the user this control. To reduce the burden this control places on the user, I model the user's patience for fetching objects of various priorities and suppress the interaction when it appears to be of only marginal value. The *user patience model*, which appears in Section 6.3, helps make the interface less intrusive.

The second type of decision I expose to the user is the volume of data to fetch in the event of a hoard walk. During a hoard walk, the system may determine that it needs to fetch a substantial number of objects. If the client is currently operating weakly connected, the hoard walk could monopolize the network connection. Because the user may or may not actually need all these objects, my interface offers the user control over hoard walks. To reduce the burden, I employ the user patience model from above and also allow the user to request that the system stop asking about individual objects or tasks.

The final type of decision I consider exposing to the user is the volume of data to be reintegrated over a weak connection. Because weak connections limit the overall amount of network traffic, users need to have control over what portions of the CML are trickled back to the servers. Although not yet implemented, this aspect of the interface is described in Section 9.2.3.2.

---

[6] This is in direct contrast to a system that need only support programs running unattended. Programs have an infinite amount of patience so will not become frustrated if the system requires ten minutes to service a request. Further, people are far more flexible than programs. If one task is not available, a user is likely to choose a different one that is available instead.

4 Interacting with the User

The unifying theme of the interface, called the *CodaConsole*, was borrowed from another complex system, the automobile. Cars communicate with their drivers through a dashboard interface. Typically, the dashboard contains a number of indicator lights as well as a small number of gauges. The purpose of the lights is to grab the driver's attention and, thus, alert them to the current state. The purpose of a gauge is to offer more detailed information. For example, when the engine temperature gets dangerously high, a light might illuminate to alert the driver to the problem. The driver could then consult the engine temperature gauge to determine the severity of the problem.

Dashboard indicator lights have been reasonably successful and widely adopted. The strength of this interface lies in its ability to keep the user informed while remaining unobtrusive and allowing the driver to concentrate on the primary task of getting from here to there. But, even with its overwhelming acceptance, it is far from perfect and has at least two serious deficiencies.

The first deficiency results from not providing sufficient warning to fix certain problems before they can cause serious damage. In particular, when the oil light illuminates, the driver must stop immediately or risk serious engine damage. Although this problem may have developed suddenly, it is more likely to have been caused by a slow oil leak that went undetected for a long period of time. This lack of sufficient warning has given the dashboard interface the nickname *idiot lights*, supposedly because you are an idiot if you wait until the indicator lights before noticing a problem. This is the unfortunate consequence of a poorly designed implementation of what is an otherwise well-conceived idea.

The second of these deficiencies is its extremely limited vocabulary. A car is a complex piece of machinery. The dashboard is simply not large enough to offer an indicator and gauge for every conceivable ill. Because of this constraint, automobile designers have had to balance two competing goals. Their first goal is to alert the driver to problems. Their second is to give the driver more detailed information. They have wisely made the first goal their top priority. The number of indicator lights on the dashboard of a typical car is larger than the number of gauges. Thus, when a subsystem that only has an indicator light (and not a gauge) experiences a failure the interface offers an extremely limited vocabulary for expressing that problem. An indicator light generally expresses just three pieces of information. These three bits of information, regarding the subsystem that it monitors, are:

operating normally: indicator remains off in its steady state

needs attention: indicator remains on in its steady state

indicator operational: indicator turns on briefly upon ignition

Because of this limited vocabulary, the user may be completely unable to determine the severity of a problem without either taking the car to a service center or getting under the hood.

Dashboard indicator lights make a complex mechanical device translucent to its user. They allow naïve drivers to recognize problems easily. I borrowed the idea for the CodaConsole interface because my goal was synonymous. The fact that so many users (particularly in the United States) are already familiar with indicator lights only makes the idea more appealing. After borrowing the basic idea however, I then refined it to address the deficiencies described above.

The CodaConsole interface strives to warn the user of impending problems so that she can take a proactive role in their solution or can actively prepare for the inevitable. Obviously, some problems occur instantaneously and the system cannot always forewarn the user. Still, the goal is to offer warnings whenever possible. Second, I have increased the size of the interface's vocabulary. Rather than expressing just one of two stable states, each indicator expresses four stable states—not operational, fully operational, warning, and critical. In addition, the indicators are dynamic. If the user wants more information about a problem, she can double-click on the indicator light and a "gauge" will appear. The overriding goal of the interface was to make caching translucent without imposing too great a burden on the user. In this chapter, I present the design and implementation of the interface.

4.1 The Indicator Lights Interface

The indicator window itself consists of nine indicator lights[7] as shown in Figure 4.1. The nine indicators and their functions are

Control Panel: allow the user to customize the behavior of the interface

Tokens: alert the user to problems related to authentication

Space: alert the user to problems related to storage space

Network: alert the user to problems related to network connectivity

Advice: alert the user to opportunities to give or receive advice

Hoard Walk: alert the user to the state of the hoard daemon

Reintegration: alert the user to the state of the reintegration daemon

Repair: alert the user to file system objects that are currently in conflict

Task: alert the user to changes in the availability of hoarded tasks; allow the user to define and hoard tasks

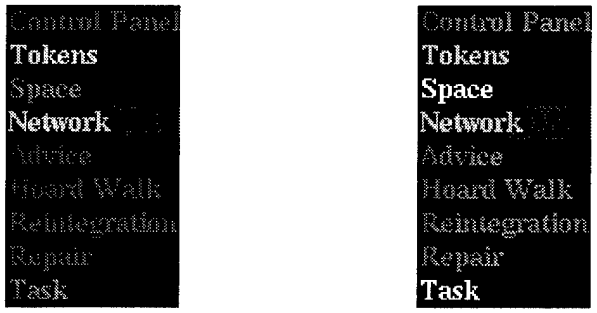
[7] Technically, the first of these, the *Control Panel*, is not an indicator light because it does not actually indicate events to the user. For simplicity, however, I will refer to it as an indicator light.

These nine indicators appear in a small window that measures about 1" x 2" on the user's display. These indicators require a modest amount of screen real estate, a resource that is precious to users, during normal operation. Because the expected benefit of the indicators outweighs its cost in screen real estate, I expect the user will keep this indicator window visible at all times. Note, however, that while the user investigates a problem indicated by the interface, the screen real estate requirements increase. This change in resource consumption is appropriate during periods when the user is actively engaged with the interface [8].

Each indicator light can signal one of four states about the subsystem it monitors:

- Operating normally
- Developing a problem or experiencing a noncritical problem
- Experiencing a critical problem
- Indicator not operational or status unknown

The first three of these states are color coded to one of three urgency levels: normal, warning, and critical. By default, the colors associated with these urgency levels are green, yellow, and red. I chose this color scheme to make it easy for users to remember since green, yellow, and red correspond to the colors used to signify similar meanings in many situations in the United States—most notably traffic signs and signals. This color scheme is not without its own difficulties. For instance, some users may be using a monochrome monitor, others may be color blind, and not all cultures use green, yellow, and red for these meanings. For these reasons, the user can easily customize the color scheme used to represent these three urgency levels. In fact, the user can forego the use of color and instead use different grayscale levels to signify the different urgency levels. Figure 4.1 shows two views of the indicator window: one in the default green-yellow-red color scheme and one in a monochrome scheme.[8]



(a) Color Scheme (b) Monochrome Scheme

Figure 4.1: Indicator Lights

This figure shows two views of the indicator lights. The indicator lights give users a peripheral awareness of system state. View (a) shows the default green-yellow-red color scheme. View (b) shows a monochrome scheme. In both views, the *Tokens* and *Network* indicators are shown with a warning urgency level and the *Space* and *Task* indicators are shown with a critical urgency level. All other indicators are normal.

[8] A color supplement to this thesis is available at http://www.cs.cmu.edu/Reports

In this section, I present each of the nine indicator lights and their associated informational windows. For each indicator, I begin by describing its purpose and the meaning of its color changes. I then describe the layout and operation of each window associated with that indicator. Some of these indicators (*Control Panel*, *Tokens*, *Advice*, *Hoard Walk*, and *Task*) have multiple supporting windows. Others (*Space*, *Network*, *Reintegration*, and *Repair*) have just a single one. I begin the presentation with the *Control Panel* indicator and all of its supporting windows, and continue through the other indicators in the order shown in Figure 4.1.

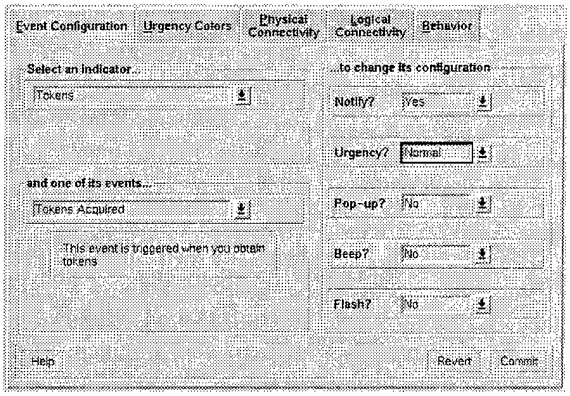
4.2 Control Panel

The first indicator light, a special case, does not actually indicate problems to the user (it is always green). Instead, it gives the user a way to control the handling of events, the colors of the urgency levels, the physical and logical connectivity to servers, and other behaviors of the interface. In the sections that follow, I will explore each aspect of the control panel.

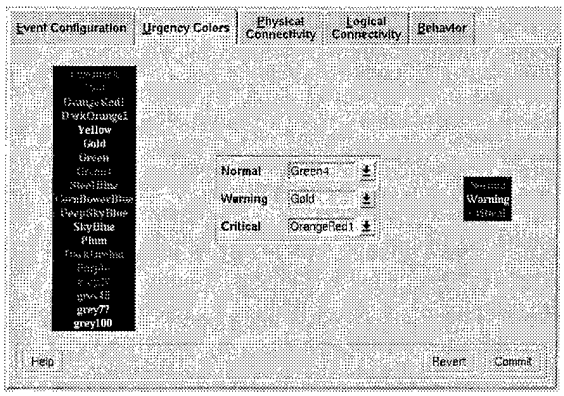
4.2.1 Event Configuration

When the user double-clicks on the *Control Panel* indicator, the *Event Configuration* window (one tab of the control panel window) appears. From this window (shown in Figure 4.2a), the user can customize event notification. In particular, the user can specify whether or not she wants to be notified of a particular event. If so, she can specify how the system will indicate its occurrence to her. Her notification options range from automatically popping up the appropriate window to quietly changing the color of the given indicator light.

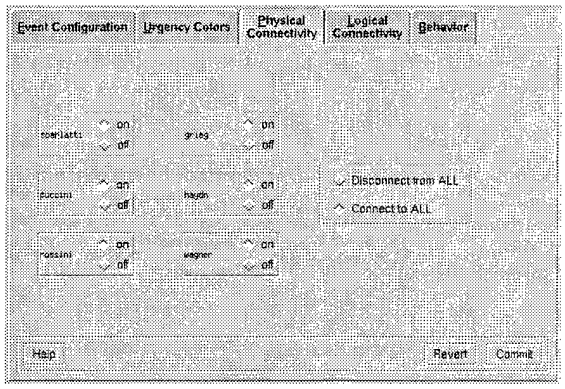
The layout of this window organizes the events by the indicator that signals them. Thus, only those events associated with the selected indicator can be chosen, and selecting a different indicator changes the list of events that can be chosen. Once an event is selected, the user can change its configuration using the right-hand side of the window. The user must first decide whether or not she wishes to be notified of this event. If she chooses not to be notified, all of the other selections are grayed out. If she chooses to be notified, she must then define the urgency of the event and how she wishes to be alerted to this event. The minimum (and default) action the system takes to alert the user of an event is to change the color of the event's indicator to reflect the urgency of the event (e.g., red, yellow, or green). In addition, the user can customize the alert to pop up the appropriate window automatically, to beep, and/or to flash the indicator light on and off a few times.



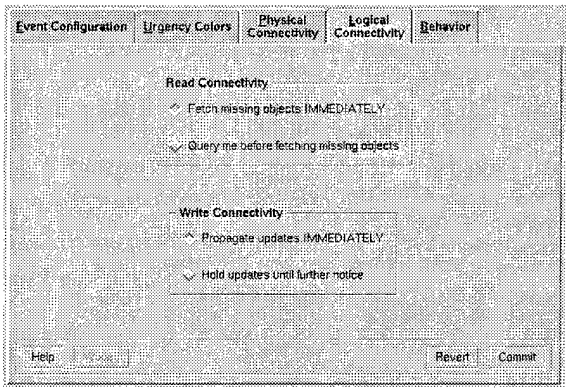
(a) Event Configuration Tab



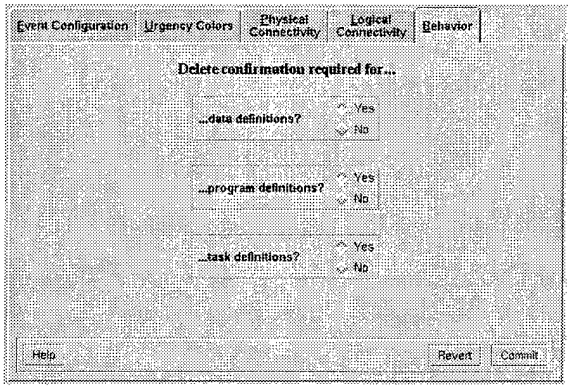
(b) Urgency Colors Tab



(c) Physical Connectivity Tab



(d) Logical Connectivity Tab



(e) Behavior Tab

Figure 4.2: The Tabs of the Control Panel

This figure shows the supporting windows that appear when the *Control Panel* indicator is double-clicked. From these windows, the user can control various aspects of the interface. The *Event Configuration* tab, view (a), allows the user to control the way events are notified. The *Urgency Colors* tab, view (b), allows the user to customize the color scheme used to indicate the three levels of urgency. The *Physical Connectivity* tab, view (c), allows the user to control connectivity to each Coda server. The *Logical Connectivity* tab, view (d), allows the user to control the servicing of cache misses and the propagation of updates while remaining fully connected to the network. The *Behavior* tab, view (e), allows the user to disable confirmation dialogue boxes.

4.2.2 Urgency Colors

If the user clicks on the *Urgency Colors* tab of the *Control Panel*, she will see the window that appears in Figure 4.2(b). From this window, the user can choose what colors the interface will use to indicate each of the different urgency levels. By default, the normal urgency level is indicated by the color green, the warning level is indicated by yellow, and the critical level is indicated by red. The user has a choice of about a dozen colors and four different levels of grayscale.

The layout of this window shows three vertical sections. The leftmost section shows a list of color names.[9] Each name is shown in the named color on a black background. This section gives the user an idea of how the color will look in the indicator light window. The rightmost section shows the three colors selected to represent the three different urgency values. This section allows the user to see how easily the three chosen colors can be distinguished from one another. The middle section provides widgets that allow the user to change the color representing each urgency level. As the user plays with her color choices, they are reflected in the sample indicator area on the right. The actual indicator lights are not updated until the user commits her choices by selecting the Commit button.

4.2.3 Physical Connectivity

If the user clicks on the *Physical Connectivity* tab of the *Control Panel*, she will see the window that appears in Figure 4.2(c). This window allows the user to toggle the network connection to individual servers, as if the interface could simply unplug a wire running between this client and each individual server. Obviously, this window does not control the physical network cables. Instead, it instructs the communication package that is linked into the Venus binary to drop packets to and from the indicated server. Controlling the connectivity in this way allows us to stop communication without physically unplugging the network connection; furthermore, it gives a finer level of control than physically unplugging the network connector. By default, the connection to each server is turned on.

Controlling file system activities at this level is an advanced feature, not one I would ideally expect novice users to master. It is, however, useful for performance reasons, when operating weakly connected, to disconnect from all but one server in each replication group. This tab could also be extended to allow the user to control the available bandwidth to or from a server by issuing the appropriate instructions to the communication package, though this has not been implemented.

---

[9] Although it would be simpler to make the drop-down list boxes contain the colored listing shown on the left, this is not currently possible in the `tix` widget library.

4.2.4 Logical Connectivity

If the user clicks on the *Logical Connectivity* tab of the *Control Panel*, she will see the window that appears in Figure 4.2(d). This window allows the user to control read and write connectivity to the Coda servers. *Read connectivity* is the ability to service cache misses—in essence, the ability to read data from the servers. *Write connectivity* is the ability to propagate updates—in essence, the ability to write data to the servers. Logically, the system can operate connected or disconnected with respect to each of these forms of connectivity. When a client is operating disconnected, it is both read and write disconnected. When a client is operating connected, it is both read and write connected. When a client is operating weakly connected, it could be operating read and write connected (the default) or it could only be operating connected with respect to one of these activities. This window gives the user control over which types of file system requests the server handles and which the client handles.

The ability to control read connectivity to the servers allows users to test their preparation for an upcoming disconnected session without the inconvenience of waiting for the connections to servers to timeout. The ability to control write connectivity to the servers allows users to improve performance during periods of poor network connectivity (or poor server performance), as well as when their activities involve numerous updates that are temporary in nature (e.g., `make venus` to build the Venus binary, followed immediately by `make clean` to remove object files). Controlling file system activities at this level is an advanced feature, not one I would initially expect novice users to master.

4.2.5 Behavior

If the user clicks on the *Behavior* tab of the *Control Panel*, she will see the window that appears in Figure 4.2(e). This window allows the user to turn off the confirmation dialog boxes that appear when she deletes a program, data, or task definition. By default, such deletes require confirmation. This tab could also be extended to give users control over other behaviors, though this has not been done.

4.3 Tokens

The second indicator light, labeled *Tokens*, alerts the user to problems related to authentication. When the indicator is shown in green[10], the user knows that she has valid authentication tokens. If the user double-clicks on the *Tokens* indicator, the window shown in Figure 4.3(a) appears. This window informs the user as to when her tokens expire and allows her to reauthenticate or destroy her tokens. When the indicator light

[10] Throughout this thesis, I assume the user is operating with the green-yellow-red color scheme.

(a) Tokens Valid

(b) Tokens Expired

(c) Tokens Expired and Reintegration Pending

Figure 4.3: The Windows of the Tokens Indicator

This figure shows three windows associated with the *Tokens* indicator light. If the indicator light is shown with a normal urgency level and the user double-clicks on the indicator, then the window shown in view (a) will be displayed. View (b) will be shown when the indicator is at a warning urgency level and view (c) will be shown at a critical urgency level.

changes to yellow, the user knows that her tokens have expired. If she double-clicks on the *Tokens* indicator at this point, the window shown in Figure 4.3(b) will appear allowing her to reauthenticate. If the user does not have valid tokens and either a file system request or a reintegration is pending, the indicator light turns red. Double-clicking on it will cause Figure 4.3(c) to be displayed. This window describes any pending activities and allows the user to authenticate.

An alternative design would use yellow to indicate that a user's tokens were *about* to expire and red to indicate that they *had* expired. This design, however, loses the indication that an activity is actually waiting for the user to obtain tokens. I chose not to use this alternative design because of a problem that I had observed among Coda users. Occasionally, a user would reconnect to the system, but fail to authenticate. Without tokens, the client could not reintegrate their updates with the servers. Users would then become confused because their updates had not yet appeared on the servers. In addition, though, I felt that there was nothing inherently bad in not having tokens and so changing the indicator to red just because the user's tokens had expired seemed too alarmist. It seemed more appropriate to warn them and then turn the indicator red when the user's activity was stalled because of a lack of tokens. Upon further reflection, I would probably chose the alternative design if I were to redesign the system today. People seem to find these meanings more intuitive than those used in the current design.

4.4 Space

The third indicator light, labeled *Space*, alerts the user to problems related to a lack of space. The space indicator light monitors three distinct areas. The first area is the file cache. The user is alerted when the percentage of cache space devoted to hoarding file system objects exceeds either the warning or critical thresholds. The second area is the local disk. The user is alerted when the available space remaining on the local disk is less than that required for the remaining Venus cache.[11] The third area is RVM, an internal area required for proper Venus operation.

The window displayed when the user double-clicks on the *Space* indicator contains three gauges as shown in Figure 4.4. The length of the gauge is proportional to the size of the cache, disk, or RVM, respectively. The length of the colored bar represents the amount of space currently in use; the color of the bar represents the urgency of the problem. Green means there are sufficient resources to continue normal operations. Yellow alerts the user to a potential problem. Red alerts the user to a critical problem. The help window explains what corrective action, if any, the user can take to resolve the problem. The contents of this help window changes as the *Space* indicator state changes.

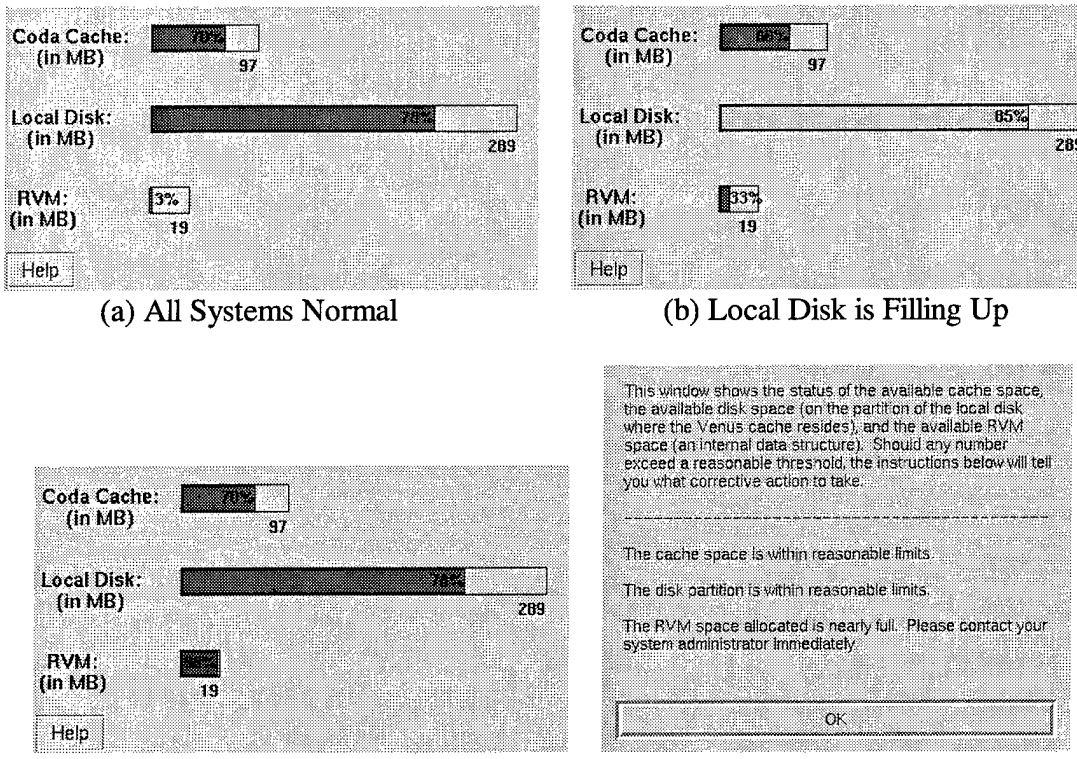
4.5 Network

The fourth indicator light, labeled *Network*, alerts the user to problems related to network connectivity. When this indicator is shown in green, the user knows that Venus is operating strongly connected to all Coda servers from which it has cached data. When this indicator is shown in yellow, the user knows that Venus is operating weakly connected to at least one Coda server, and that it is *not* disconnected from any servers. When this indicator is shown in red, the user knows that Venus is operating disconnected to at least one Coda server.

Double-clicking on the *Network* indicator light will cause a window similar to those in Figure 4.5 to appear. This window displays a gauge representing the current bandwidth available to each Coda server. The label to the left of the gauge is the name of the server. The length of the bar represents the maximum Ethernet bandwidth. The length of the colored bar represents the current bandwidth to this server as a percentage of that maximum. The color of the bar represents the state of connectivity to this server: strongly connected (green), weakly connected (yellow), or disconnected (red). If the name of the server is shown in gray, then the user has artificially manipulated the network connection to this server using the *Physical Connectivity* tab of the *Control Panel*. The user may click on the Control Panel button to bring up this tab to resolve the situation.

---

[11] For example, Venus is typically "allocated" space for its cache on a local Unix partition shared with other programs and users. If a misbehaving program or a malicious (or naïve) user fills that partition and leaves Venus inadequate space for this cache, Venus behavior is undefined. At worst, Venus will fail. At best, its cache performance will suffer. Alerting the user to this problem allows the user the opportunity to resolve the problem and either prevent Venus from failing or improve its cache performance.



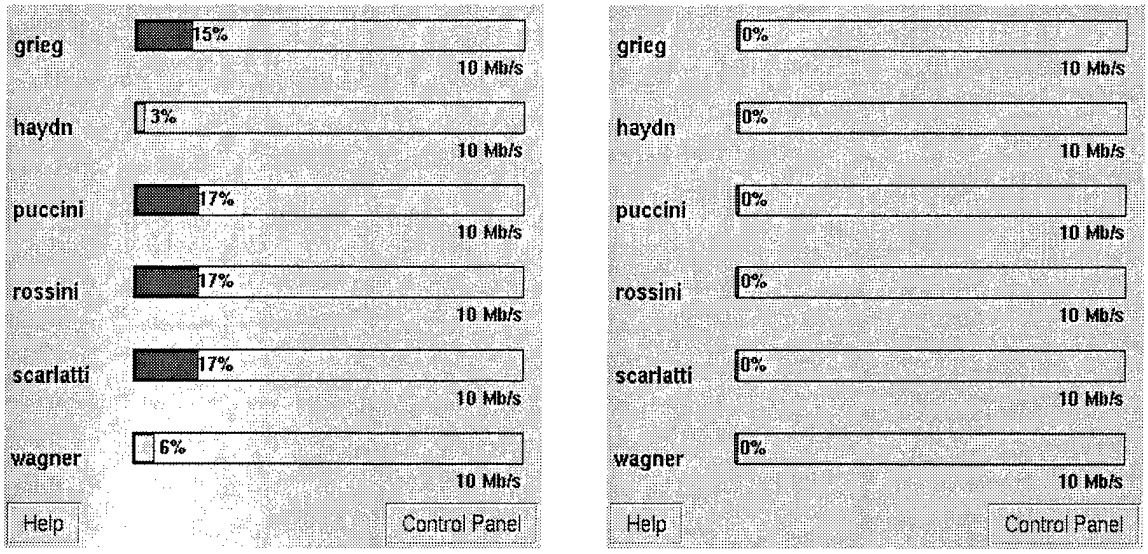
(a) All Systems Normal (b) Local Disk is Filling Up

(c) RVM Space Almost Full (d) Help Window for (c)

Figure 4.4: Space Information Window

This figure shows three views of the *Space Information* window as well as its help window. View (a) shows the *Space Information* window when all three areas are within normal limits. The three gauges in this view are all green. View (b) shows the window when the local disk is beginning to fill up. Because there is still space available, the local disk gauge is yellow (as would be the *Space* indicator light). View (c) shows the window when the RVM space is almost full. Because there is almost no RVM space available, its gauge (and the indicator light) would be red. View (d) shows the help window that would appear if the user were to click on the *Help* button of view (c). This window explains that the user should contact their system administrator to resolve this problem.

Currently, this window shows one gauge for each server of which the client is aware. As the number of Coda servers increases, the content of this window becomes lost to the user. This window should be modified to list only those servers the user is actively accessing. If the user is accessing more servers than can fit on the window, the interface should allow them to be scrolled. Neither of these extensions would be difficult. Further, the gauge shows the network bandwidth as a percentage of the maximum Ethernet bandwidth, a metric that is not meaningful to the user. Rather than showing the relative percentage of Ethernet bandwidth (e.g., 3%), the interface could show the absolute bandwidth estimate (350 Kb/s). Neither of these modifications to the interface would be difficult.



(a) Client operating weakly connected (b) Client operating disconnected

Figure 4.5: Network Information Window

This figure shows two views of the *Network Information* window. View (a) shows that this client is operating weakly connected to `haydn` and `wagner`, but strongly connected to all the other Coda servers. The gauges for `haydn` and `wagner` are yellow; those for the other servers are green. The *Network* indicator light would be yellow in this situation. View (b) shows that this client is operating disconnected to all servers. The gauges show a thin sliver of red. The *Network* indicator would also be red.

The distinction between strong and weak connectivity is made based upon comparing the current network bandwidth estimate to a threshold. If the current bandwidth is above the threshold, the system considers itself to be strongly connected; if it is below, the system considers itself weakly connected. Ideally, users should have the ability to control the value of this threshold, though novice users should not be required to manipulate it. Such an extension is discussed in Section 9.2.2.1.

The behavior of this indicator, as described here, is probably not what I would design today. The problem is that the indicator changes state before it is really appropriate to do so. For example, as soon as the system realizes that a server has crashed, the indicator turns red. However, because most Coda files are replicated, the fact that a single server has crashed is not terribly important to the user. It would be more appropriate to change the state of the indicator when a volume has transitioned into the disconnected state (of Figure 2.1). The informational window would then need to expose the pathnames of volumes operating disconnected (and weakly-connected) as well.

4.6 Advice

The fourth indicator light, labeled *Advice*, alerts the user to pending requests for advice from Venus and of hoard hints. At critical decision points during operation, Venus may request advice from the user. These requests frequently relate to network usage. At other times, Venus may notice anomalies in the user's hoard database and may alert the user to them by way of a hint.

When the user double-clicks on this indicator light, the window shown in Figure 4.6 appears. It has two sections. The top section, labeled Advice Needed, contains a list of advice requests. The bottom section, labeled Advice Offered, contains a list of hoard hints. Each type of request and each type of hint are marked with either a warning or critical urgency level. By default, requests are marked as critical only when a thread is blocked waiting for a response. All other requests and all hints are alerted at the warning level. The color of the indicator light is red if any critical requests are pending user attention, and yellow if any noncritical requests or hints are pending attention. It is green if there are no requests or hints outstanding.

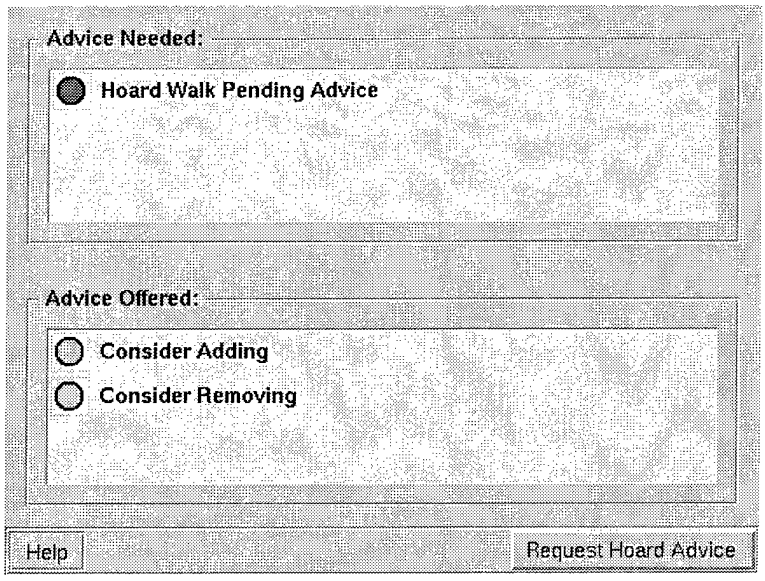


Figure 4.6: Advice Information Window

This figure shows the *Advice Information* window. It is displayed whenever the user double-clicks on the *Advice* indicator light. The top portion of the window, labeled Advice Needed, contains queries posed by Venus. By answering these questions, the user may influence the behavior of Venus. The bottom portion of the window, labeled Advice Offered, contains hints offered to the user regarding the content of the cache and of the hoard database. The circles to the left of each entry indicate the urgency of the request.

4.6.1 Requesting Advice from the User

Venus requests advice from the user in two situations. The first occurs when a cache miss occurs. The necessary advice comes in three flavors depending upon the current network connectivity. If Venus is disconnected from the servers of this object, then the request shown in Figure 4.7 arrives. The purpose of this advice request is to obtain evaluation data from the user. If, instead, Venus is operating weakly connected to all of the servers of this object, then the request shown in Figure 4.8 arrives. The weak-miss query allows the user control over demand requests. Finally, if Venus is strongly connected to the servers of the missing object and the user has put the system in *read disconnected* mode, then the request shown in Figure 4.9 arrives. The read disconnected miss query makes caching translucent to the user, making cache misses observable events. The second type of advice request occurs during hoard walks when Venus is operating weakly connected. During the course of a hoard walk, Venus may need to fetch substantial amounts of data—not all of which is strictly necessary from the user's point of view. The request shown in Figure 4.10 gives the user control over the quantity of data that is fetched during the data walk phase of a hoard walk. Each of these advice requests is discussed in the following sections.

4.6.1.1 Disconnected Cache Miss Query

The disconnected cache miss query arises from a cache miss that occurs while Venus is operating disconnected. The purpose of this query is to determine the severity of a given cache miss. Because the severity of a miss is highly dependent upon the user's current activities and goals, the system cannot determine this information automatically. Instead, the system requests the user's assistance. An important use of such data would be to evaluate the CodaConsole interface under conditions of actual use, as will be discussed in Section 9.2.1.4.

As shown in Figure 4.7, the query contains the name of the missing object as well as the name of the program requesting that object. The user is asked to estimate the severity of the miss using a scale. The user may indicate uncertainty if she cannot determine the severity of the miss. The user can also indicate that she is testing a hoard database for a future disconnected session. Finally, the user is given an opportunity to provide additional information, though this is optional.

4.6.1.2 Weakly Connected Cache Miss Query

The weakly connected cache miss query (see Figure 4.8a) results from a cache miss that occurs while Venus is operating weakly connected from all of the servers of the missing object. The purpose of this advice request is to allow the user control over network resources when those resources are scarce. Once again the system tells the user the name of the missing object and of the program requesting that object. In addition to this information, the system also tells the user the estimated fetch time of the object. The system queries the user to determine whether or not to fetch the object.

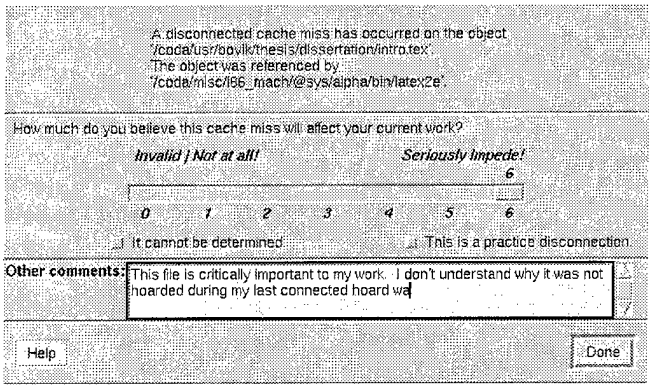


Figure 4.7: Disconnected Cache Miss Questionnaire

This figure shows a query that results from a disconnected cache miss. The top portion of the window contains the name of the missing object and of the program requesting it. The middle portion allows the user to indicate how much the cache miss will affect the user's work. The bottom portion allows the user to provide any additional comments she might have regarding this miss.

A complication of this design arises because the user may be unavailable to answer the request. Perhaps the user has stepped away from her laptop to get dinner. This is where the Unix paradigm is important. If the user is unavailable to answer a request, she will want the system to have made progress during her absence. Imagine her frustration if she returns from dinner only to find a dialog box blocked waiting for an answer! To alleviate this problem, the system times out after a reasonable period (currently, about 15 seconds). Because the system cannot determine whether the user is considering her options or is unavailable, it presents the window shown in Figure 4.8(b). If the user was simply slow in making a decision, she will notice the prod and can request that the system continue to wait (at which point the window shown in Figure 4.8c is displayed). If the user was, indeed, unavailable, this prod will also time out and the system can take the default action (currently, to fetch).

An obvious design alternative is to initiate the fetch and provide the user with the name of the object, the name of the requesting program, a progress meter, and an abort button. Such interfaces are extremely common, and given current network access characteristics, more appropriate. In the current implementation of Venus however, aborting a fetch (once it has been started) is not possible without tearing down the entire RPC connection. The ability to abort a fetch in progress would certainly be useful, but it would require a fair amount of programming for which the research contributions are minimal. Further, in the future, I expect networks to charge per packet (or per byte). Once money is involved, users will demand not only the ability to abort a request but also the ability to approve charges before they are incurred. For these reasons, I choose to explore the "ask permission" design alternative and leave the "beg forgiveness" one for future extension.

An interesting extension would be to recognize when the system is running unattended. For instance, after the third time a request for advice times out, the system might decide that the user must not be available. When the user is unavailable, the system could stop asking for advice and act on the best available information. The difficulty here is

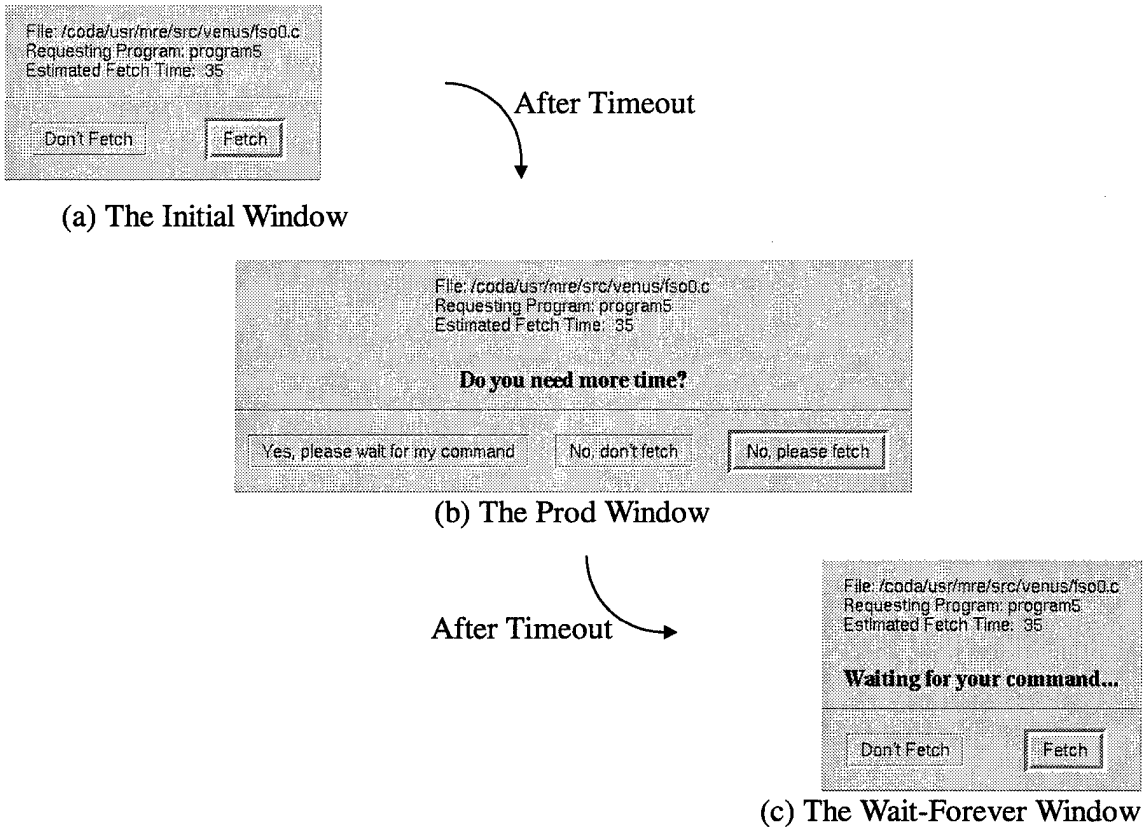


Figure 4.8: Weak-miss Interaction

This figure shows the progression of a weakly connected cache miss query. View (a) is triggered when the user requests (perhaps indirectly) a file system object that is not currently cached and whose estimated fetch time exceeds a certain limit (as described in Section 6.3) while the client is operating weakly connected. This view displays the name of the missing object, the name of the requesting program, and the estimated fetch time. If the user has not answered the query after a certain amount of time (15 seconds by default), then view (a) is replace by view (b). This new view displays the same information, but offers the user the opportunity to consider the request without time pressure. If the user has not responded to this second query within another timeout period, the system will destroy this window and immediately begin fetching this object. If the user wants more time to consider her decision, she can click on the Yes, please wait for my command button. View (b) will then be replaced by view (c) and the system will wait indefinitely for her response.

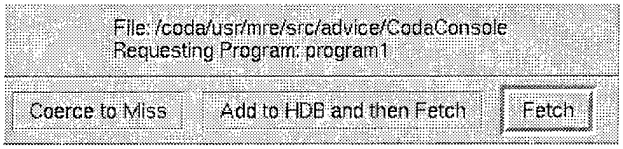


Figure 4.9: Read Disconnected Cache Miss Questionnaire
This figure shows a read disconnected cache miss query. The top portion of the window contains the name of the missing object and of the program requesting it. The buttons at the bottom allow the user to specify what action the system should take. From left to right, the system will coerce the request into a cache miss (and thereby not fetch the file), fetch the file and add the object to the hoard database, or simply fetch the file.

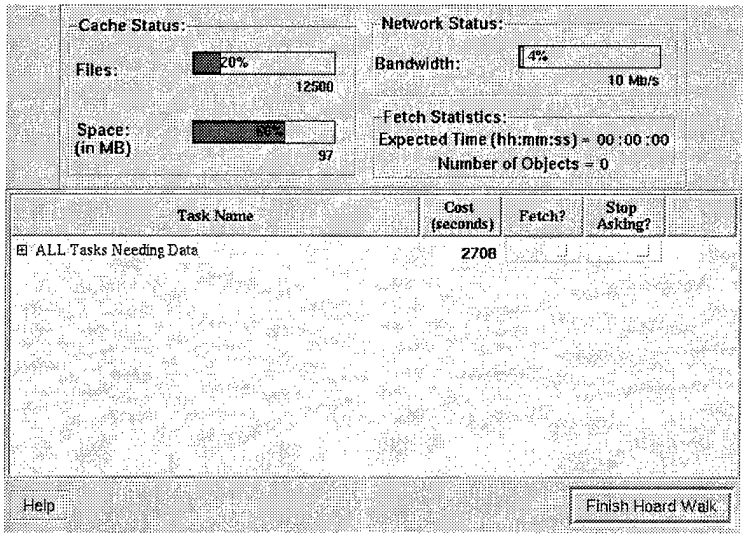
determining when the user returns and is again able to request advice. One solution to this problem is for the interface to pop-up a "Click here when you return" window on the screen. This extension allows the system to handle both attended and unattended operation, while minimizing the time wasted waiting for the user to respond to queries.

4.6.1.3 Read Disconnected Cache Miss Query

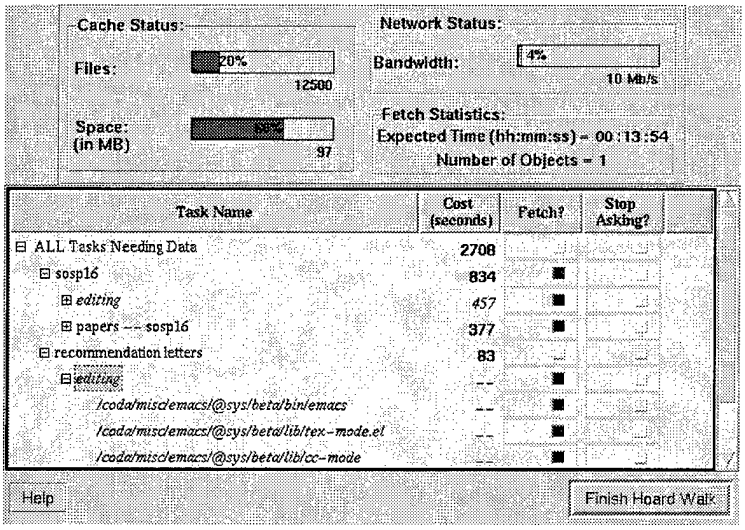
The query that is shown in Figure 4.9 results from a cache miss that occurs while Venus is operating read disconnected (see Section 4.2.4). Read disconnected operation occurs only at the request of the user and helps the user test her hoard database in preparation for a future disconnected session. The purpose of this request is to make cache misses, which are normally transparent to the user, observable. By making cache misses translucent to the user, she is better able to notice omissions in her hoard database. As is true of all the cache miss queries, this request also presents the user with the names of the missing object and the requesting program. In this case, however, the user is presented with three options. She can advise Venus to fetch the object, to fetch the object and add it to the hoard database, or to coerce the request to a miss. The latter option allows the user to determine the effect of not having the object resident in the cache during a disconnected session and allows her to make more informed hoarding decisions.

4.6.1.4 Hoard Walk Advice

The hoard walk advice request occurs during weakly connected operation between the status walk and the data walk when the amount of data to be fetched is expected to take a substantial amount of time. The purpose of this request is to allow the user to control the amount of data fetched over the weak network. After performing the status walk, Venus knows which objects need to be fetched and their sizes. From this information, it can determine the expected fetch times and request advice from the user appropriately. As seen in Figure 4.10, the user is presented with cache status information, current network conditions, total expected fetch time and a hierarchical listing of tasks that require data to



(a) Initial View



(b) Expanded View

Figure 4.10: Hoard Walk Advice

This figure shows a hoard walk advice request. View (a) is displayed initially. The upper left corner shows the status of the cache: the percentage of cache container files and cache blocks dedicated to hoarded objects. The upper right corner shows the current average bandwidth to the servers. The expected time to complete the hoard walk is shown below the network status. The main area of the window shows a list of the tasks that need data to be fetched. Initially, only the "All Tasks Needing Data" task is shown. If the user expands this task as well as the tasks one level below it (by clicking on the "+" signs or double-clicking their names) and then selects the `sosp16` task, the window shown in view (b) will be displayed. Italicized text names indicate a task that is contained in more than one task definition. For each element of this hierarchical list, the expected cost (currently shown only in units of time) is listed to the right of the task name. Further to the right are two checkboxes. The left checkbox allows the user to select the item to be fetched. The one to the right allows the user to instruct the system to not fetch the item during the current hoard walk and, furthermore, to not ask about this item in future hoard walks during this weakly connected session.

be fetched.[12] The task listing is an expandable tree. At the top-most level, a pseudo-task represents all tasks needing data. Expanding this pseudo-task reveals hoarded tasks needing data. The bottom-most level shows individual files and directories that need to be fetched. For any task or object, the user can choose one of three options:

Option:	**Action to choose:**
Do not fetch during this hoard walk	No action required
Fetch during this hoard walk	Click on Fetch? checkbox
Do not fetch until strongly connected	Click on Stop Asking? checkbox

When the user requests that a task or object be fetched, the estimated cost of fetching that task or object is added to the *Fetch Statistics* section of the window. As Figure 4.10(b) shows, *editing* is a subtask of both *sosp16* and *recommendation letters*. Tasks and objects duplicated in other parts of the tree are shown in italics. The cost of fetching these objects is charged only to the first task to request they be fetched as shown by the *editing* subtask in Figure 4.10(b); other tasks obtain such data "for free", as indicated by the "--" in the figure. Once the user has specified those tasks or objects that should be fetched, she can request that the hoard walk complete by clicking on the Finish Hoard Walk button in the lower right corner of the window.

4.6.2 Offering Advice to the User

Venus offers two types of advice to the user. Both relate to task definitions. The first type identifies files that the user may have forgotten to include in a definition. These are files that the user should, perhaps, be hoarding, but is not currently. The second type identifies files that the user may have included unintentionally in a definition or that the user no longer needs. These are files that the user is hoarding, but probably should not be. In both cases, files are identified by heuristics. In this section, I will present the windows used to offer the advice to the user. I will postpone discussing the specific heuristics until Section 6.2.1.

Unlike requests for advice, which are triggered by specific events, offers of advice are presented only when the user specifically requests the information.[13] After the user clicks on the Request Hoard Advice button on the *Advice Information* window, the interface examines usage statistics maintained by both Venus and the advice monitor. The results of this analysis appear in the Advice Offered section of the *Advice Information* window in

[12] The original version of this request, as presented in [34], listed all files and directories needing to be fetched. This design violates the third principle, "speak the user's language", discussed in Chapter 3. By present this information in a task-based hierarchy, not only can the user control how much detail they examine but they also have an idea of whether or not a file is important to their work.

[13] The interface could trigger these offers of advice automatically. Two obvious triggers are after the user requests a hoard walk and after the user reconnects following a disconnected session. I chose not to trigger this advice automatically, however, to minimize the frequency with which we interrupt the user with advice.

the form of two entries: Consider Adding and Consider Removing. The windows associated with these two entries appear in Figure 4.11.

4.6.2.1 Consider Adding

The purpose of this window is to alert the user to file system objects that she might want to hoard. This information is presented in tabular form as seen in Figure 4.11(a). The table shows the pathname of the object as well as an indication of why the object should, perhaps, be hoarded. In addition, each object has an associated check box that allows the user to indicate that the object should not be hoarded (and thus, prevent the system from identifying this object in the future).

4.6.2.2 Consider Removing

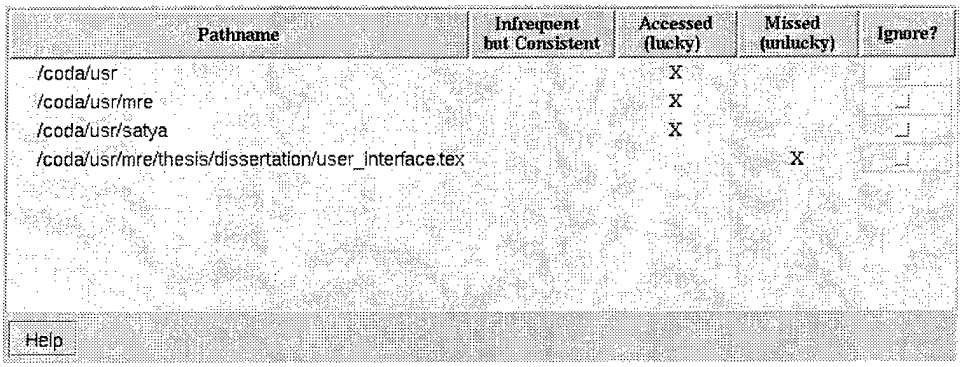
The purpose of this window is to alert the user to file system objects that are currently hoarded, but that are not being used. This information is presented in tabular form as seen in Figure 4.11(b). As before, the table shows the pathname of the object, an indication as to why the object should not be hoarded, and a check box that prevents future identification of this object.

4.7 Hoard Walk

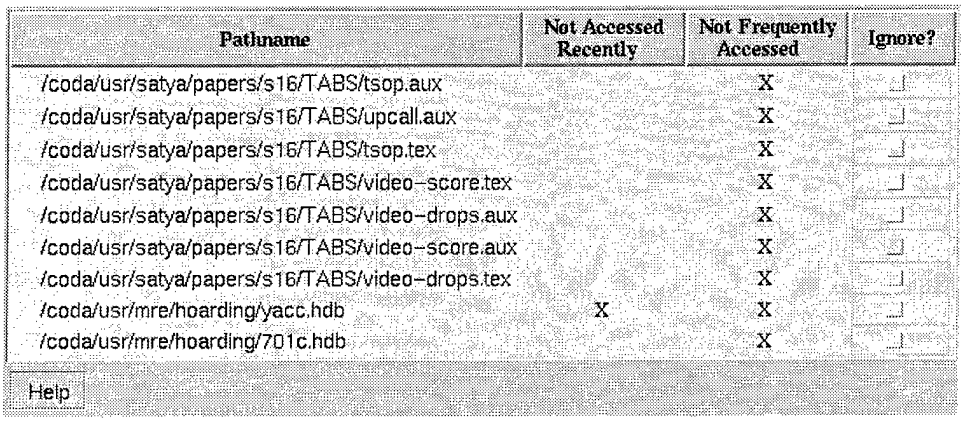
The sixth indicator light, labeled *Hoard Walk*, alerts the user to the status of the hoard daemon. When this indicator is shown in green, the hoard daemon is inactive. If the user double-clicks on the indicator light, the window shown in Figure 4.12(a) will appear. When the hoard daemon begins walking the hoard database, the indicator light turns yellow. Double-clicking the indicator during a hoard walk displays the progress meter shown in Figure 4.12(b).

During the course of a hoard walk, a large amount of data may need to be fetched. When Venus is operating weakly connected, a hoard walk can take a substantial amount of time, monopolizing the network for the duration of the walk. To allow the user to control the amount of data actually fetched during any particular hoard walk, Venus asks the user for advice after the hoard daemon completes the status walk and before it begins the data walk. When Venus requests this advice, both the *Hoard Walk* and *Advice* indicator lights turn red. The advice window, shown in Figure 4.10(a), is displayed when the user double-clicks on the *Hoard Walk* indicator. (It is also available through the *Advice* indicator window shown in Figure 4.6.) For more information regarding the specifics of the hoard walk advice request, see Section 4.6.1.4.

Hoard walks occur periodically during the course of normal operation. Currently, the period is approximately 10 minutes. However, at times, users prefer that the hoard daemon walk the database only on demand. For this reason, the *Hoard Walk Information* window allows the user to turn periodic hoard walks off (see Figure 4.12a).



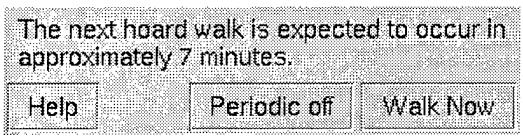
(a) Consider Adding Advice



(b) Consider Removing Advice

Figure 4.11: Advice Regarding Hoard Content

This figure shows the windows used to offer advice to the user regarding the content of the hoard database. View (a) suggests files and directories that the user might want to consider adding to one or more task definitions. View (b) suggests files and directories that the user might want to consider removing from one or more task definitions. In both cases, the user is presented with a table of information. The table shows the pathname of the object as well as an indication as to why the given file or directory might need to be added or removed. In addition, each object has an associated check box that allows the user to tell the system to ignore this object in the future.



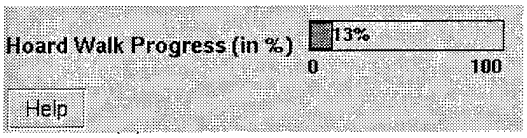


Figure 4.12: Hoard Walk Indicator Windows

This figure shows two views of the *Hoard Walk Information* window. If the hoard daemon is inactive, the window shown in (a) is displayed when the user double-clicks on the *Hoard Walk* indicator light. This window informs the user when the next walk is expected to begin, and allows the user to begin a hoard walk. If the hoard daemon is active, the window shown in (b) is displayed when the user double-clicks on the indicator light. This window gives the user an indication as to the progress of the hoard daemon.

4.8 Reintegration

The seventh indicator light, labeled *Reintegration*, alerts the user to volumes (called *subtrees* in the interface) that need to be reintegrated with the Coda servers. The indicator light turns yellow when a reintegration is currently in progress. The indicator light turns red when a volume needs to be reintegrated but cannot be because the user is not currently authenticated. For example, if the `u.mre` volume (mounted at `/coda/usr/mre`) needs to be reintegrated but user `mre` does not have valid tokens, the reintegration indicator light would turn red. If the user then double-clicked on the indicator light, the window shown in Figure 4.13 would appear. Once all pending reintegrations complete, the indicator light returns to green.

Exposing the state of reintegration to the user is important because it addresses two usability problems observed in the original Coda system. The first of these occurred when users returned from a period of disconnection and failed to authenticate. Without tokens, the client could not propagate the updates to the servers. By turning the "Reintegration" indicator red in this situation, the user is made aware that a reintegration is pending. The second problem occurred when a user who is operating weakly connected makes updates and expects to see those updates reflected on the servers before they have actually been propagated. By turning the indicator yellow, the user can see that a reintegration is in progress. In this case, the indicator serves to influence the user's expectations about the system's behavior and state.

An interesting extension to this indicator light window is to allow the user to have some control over how much of a volume is reintegrated at one time. Such an extension is presented in Section 9.2.3.2, including a paper design of the graphical interface offering this control.

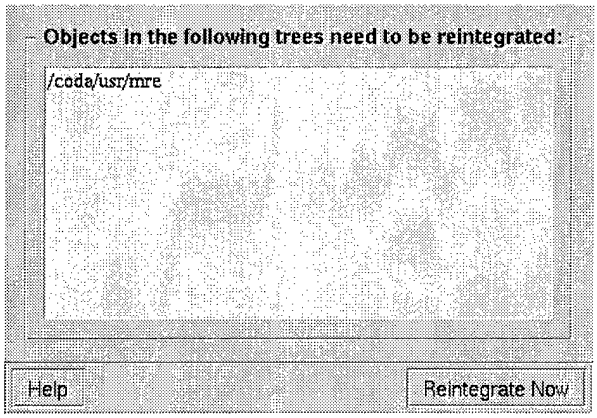


Figure 4.13: Reintegration Information

This figure shows the *Reintegration Information* window. It is displayed when the user double-clicks on the *Reintegration* indicator light. When the indicator light is red, this window shows which subtrees need to be reintegrated. Although Venus performs reintegration at the granularity of volumes, the interface presents the user with the pathname to the mount point of the volume.

4.9 Repair

Coda uses an optimistic replica control strategy. This strategy allows any client to modify data on any accessible server. One consequence of this design is that replicas may come in conflict with one another. The situations that lead to conflict are infrequent, though not impossible, in the environments for which Coda is most appropriate. When a file or directory becomes inconsistent, the system attempts to resolve the conflict automatically. Should automatic resolution fail, the user must manually repair the problem.

The eighth indicator light, labeled *Repair*, alerts the user to file system objects that are currently in conflict. The indicator light turns red when an object needs to be repaired manually and its supporting window lists those objects needing such repair. For example, if the directory `/coda/usr/mre/thesis/dissertation` were to become in conflict, the *Repair* indicator light would turn red and the window shown in Figure 4.14 would appear when the user double-clicked on the indicator light. Once all known objects needing repair are fixed, the indicator light returns to green. There is no notion of yellow for this indicator.

A simple, but useful, extension to the interface would present summary information about how the object came to be in conflict. In particular, the user might appreciate knowing who updated the files and from where those updates were made. Such information would help the user understand why the object is in conflict and would also help them to resolve the conflict. This data, however, is not readily available in Coda.

A further extension to this indicator light window is to offer a graphical interface to the repair program. When the window shown in Figure 4.14 is visible, the user could double-click on any object needing to be repaired. This action would open a repair window that would allow the user to repair the object. This extension is discussed in Section 9.2.3.1.

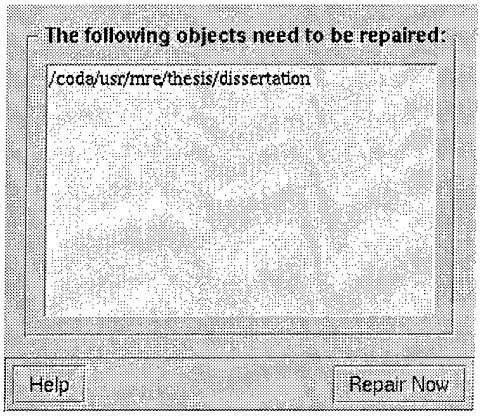


Figure 4.14: Repair Information

This figure shows the *Repair Information* window. It is displayed when the user double-clicks on the *Repair* indicator light. When the indicator light is red, this window shows which objects are in conflict and need to be repaired.

4.10 Task

As discussed in Sections 3.1.3.1 and 3.4.2.1, I must improve the model through which users identify what objects should be available during a disconnected or weakly connected session. The realization of that model appears in the final indicator light, which is labeled *Task*. This indicator allows the user to define and hoard tasks, and also alerts the user to the availability of those tasks. A task is simply the file and directories a user needs to complete a given project (e.g., writing a paper or compiling a system). It is a hierarchical structure consisting of user data, programs, and other tasks.

When the indicator is shown in green, the user knows that all hoarded tasks are currently available. When the indicator is shown in red, the user knows that at least one hoarded task is partially or completely unavailable. The indicator is never shown in yellow.

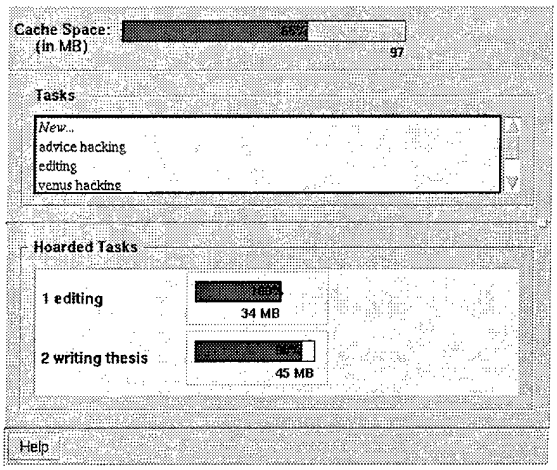
Double-clicking on the indicator light will cause a window similar to that shown in Figure 4.15(a) to appear. This window allows users to hoard (and unhoard) tasks. The top of this window shows the percentage of cache space dedicated to hoarded tasks. The middle section shows a list of all defined tasks as well as the pseudo-task "New..." that allows the user to define new tasks. The bottom portion of this window shows those tasks that are currently hoarded in priority order, where the task identified with a "1" is the highest priority.

Double-clicking on the name of a task in either of the lists of Figure 4.15(a) will display a window similar to that shown in Figure 4.15(b), allowing the user to view, modify, or create a task definition. A task has three main components: data, programs, and subtasks. For each of these components, the task definition shows two lists. The *predefined* list contains a list of all currently defined data sets, programs or tasks, respectively. The *contains* list shows those data sets, programs or tasks that are included in the current definition. Clicking on the name of an element in a predefined list includes it in the current task definition.

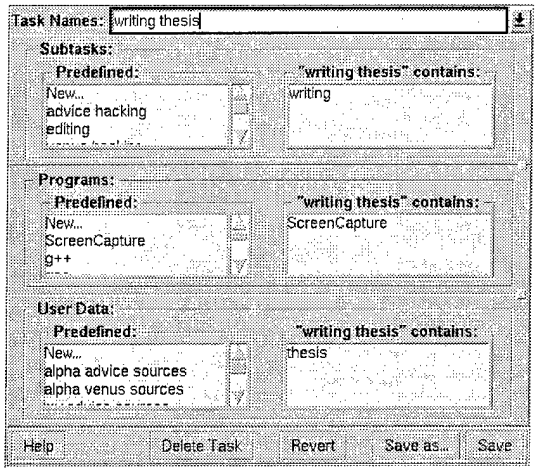
Double-clicking on an element shown in the predefined or contains lists causes a window containing the appropriate definition to appear, allowing the user to view a definition or create a new one. In particular, double-clicking on the name of a data definition will cause a window similar to that shown in Figure 4.15(c) to appear. Similarly, double-clicking on the name of a program definition will cause a window similar to that shown in Figure 4.15(d) to appear.

The data definition window allows users to specify which of their files and directories should be included within a task definition. If the pathname is a directory, then the user may specify whether or not the children and/or descendants of this directory should also be hoarded. This window is essentially a graphical interface to the hoard program, as described in Section 2.3.2. Unlike previous versions of Coda, the user is required to have detailed knowledge only of their own areas of the file system.

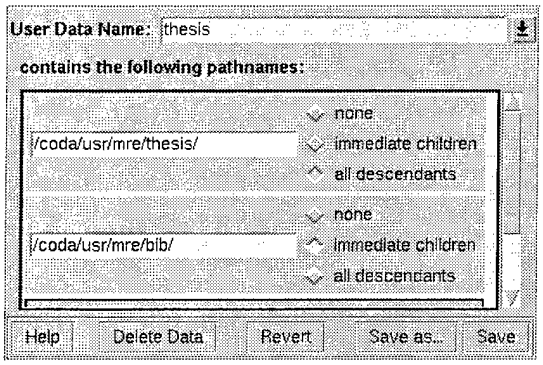
The program definition window allows users to specify programs that should be included within a task definition. Once included, the system automatically tracks file references



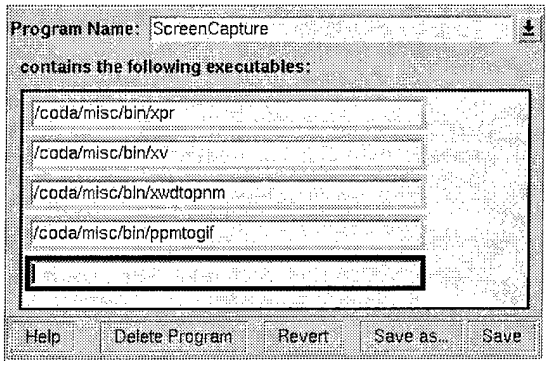
(a) Task Information Window



(b) Task Definition Window



(c) Data Definition Window



(d) Program Definition Window

Figure 4.15: The Windows Associated with the Task Indicator

This figure shows the four primary windows associated with the *Task* indicator light. View (a) shows the window displayed after clicking on the indicator. The top section of this window shows the status of the cache; the middle section shows a list of all defined tasks; and the bottom section shows those tasks that have been hoarded, their priorities, and their availability. View (b) shows the definition for the "writing thesis" task. This definition contains the "writing" task, the programs for "ScreenCapture", and the "thesis" user data set. View (c) shows the definition for the "thesis" user data. This definition includes the `/coda/usr/mre/thesis` directory and all its descendants as well as the `/coda/usr/mre/bib` directory and its immediate children. View (d) shows the definition for the "ScreenCapture" programs, including `xpr`, `xv`, `xwdtopnm`, and `ppmtogif`.

made by the program for automatic hoarding. The details of how automatic program hoarding works will be discussed in Section 6.1. With automatic program hoarding, users are no longer required to specify the files and directories required internally by these programs (e.g., library, font, and style files).

The distinction between user data and program data is crucial for reducing the burden of hoarding. Although it is not unreasonable to assume that users have detailed knowledge of their own areas of the file system, they must not be required to learn about the internal workings of application programs to make effective use of Coda. By distinguishing between user and program data, the interface allows the program profiles to be shared between tasks (e.g., not specialized to a single project). Of course, if the entire hoarding process were automated in such a way that the tasks were meaningful to users, even this burden could be eliminated. This interface does not in any way preclude such automation; in fact, this is an area identified for future research and is discussed in Section 9.2.2.4.

5 Implementation

In the previous chapter, I described the graphical user interface that makes caching translucent to the user. In this chapter, I focus on the implementation of that interface. The first section presents an overview of the system's architecture. I continue this chapter by describing a number of implementation details. These details include descriptions of some finite state machines that control the user interface as well as an overview of the modifications required within Venus. I conclude the chapter with an overview of the Translucent Caching API. This API allows Venus to notify the CodaConsole of important events and allows the CodaConsole to make the user's wishes known to Venus.

5.1 Architecture of System

Logically, the functionality of the CodaConsole belongs within Venus (see Chapter 2) because Venus alone has knowledge of the various events needed to drive the system. However, other considerations led me to make this functionality external to Venus. The benefits of translucent caching extend beyond Coda. Moving this functionality external to Venus allows the CodaConsole to be used with any highly available file system implementing its API. A second consideration was purely practical. Venus is a large and complex body of code. Further, at the time of this research, it was under development by a number of individuals. Making my interface as independent of Venus as possible isolated me from these other development efforts. For these reasons, the CodaConsole is located outside the scope of Venus and communicates with Venus via a well-defined API.

Placing the CodaConsole external to Venus imposes additional communication overhead by adding a local RPC2 call [45]. Because daemons invoke many of these calls, the user does not directly observe this additional overhead. Those calls that are invoked in the process of servicing a user request are generally infrequent or long running; thus, the additional overhead is inconsequential. A few calls would impose too great a burden and are thus batched.

The architecture, shown in Figure 5.1, is implemented in three pieces. I refer to the first two pieces, the user interface and the Advice Monitor, jointly as the CodaConsole. They implement the graphical user interface described in the previous chapter. The user interface interacts with the user. The Advice Monitor acts as a liaison between this user interface and Venus. The third piece consists of a set of hooks within Venus that inform the CodaConsole of various events.

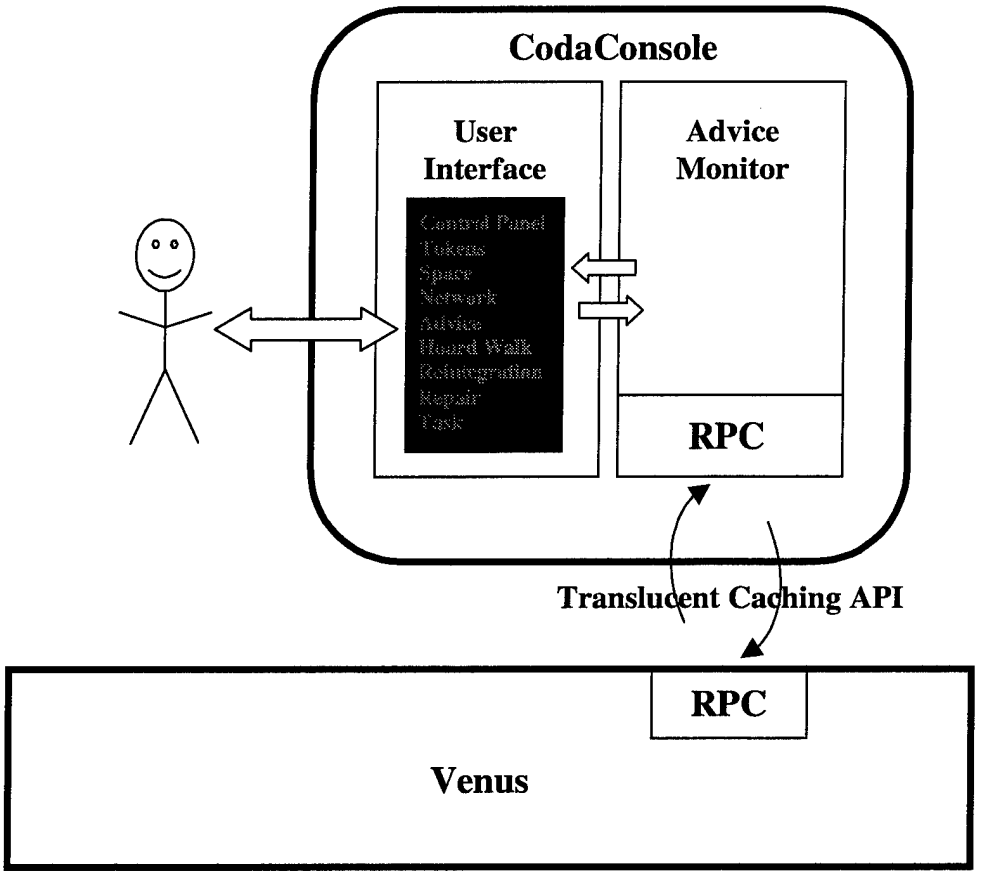


Figure 5.1: Architecture

This diagram shows the architecture of the system. The CodaConsole consists of two parts: the user interface and the Advice Monitor. The user interface implements the graphical interface described in the previous chapter. The Advice Monitor acts as a liaison between Venus and this interface. Venus notifies the Advice Monitor of various events via the RPC interface implementing the Translucent Caching API.

These three pieces communicate using two different mechanisms. The user interface communicates with the Advice Monitor via a pair of unidirectional pipes, as shown in the figure. The Advice Monitor communicates with Venus via a pair of RPC2 connections, also shown in the figure.

5.2 Details of Implementation

This section describes the implementation of each of the three components described above. I begin by describing the implementation of the user interface, focusing on the finite state machines that drive the indicator lights. I then present a brief description of the Advice Monitor. I conclude with a summary of the modifications necessary to implement the Translucent Caching API within Venus.

5.2.1 User Interface

Chapter 1 described the design of the user interface component of the CodaConsole. The user interface, which is implemented in Tcl/Tk [39, 56] using the Tix widget library [29], assumes the user is running a windowing system[14]. It is entirely event driven. As events arrive, the interface notifies the user via the indicator lights. As the user requests information, it displays the appropriate information windows. If it needs data from Venus, it contacts the Advice Monitor to request that data.

The interface presented to the user contains a set of indicator lights. These indicator lights are implemented as small, event-driven, finite state machines. The arrival of an event potentially causes a transition from one state to another. These state changes may then cause the appearance of the indicator lights to change. I will now describe three of these state machines. The remaining ones are similar in spirit, though the details differ.

5.2.1.1 Tokens Indicator

The finite state machine for the *Tokens* indicator has three states: Valid, Invalid, and Expired&Pending. The user is required to have tokens before the advice monitor begins executing so the finite state machine will start in the Valid state. When the interface receives notification that the user's tokens have expired, the machine transitions into the Invalid state. If the interface then receives a notification that an activity, a reintegration for example, is pending tokens, the machine transitions into the Expired&Pending state. The machine transitions to the Valid state from either of the other two states upon receiving notification that the user has obtained valid tokens. These states and the transitions between them are shown in Figure 5.2.

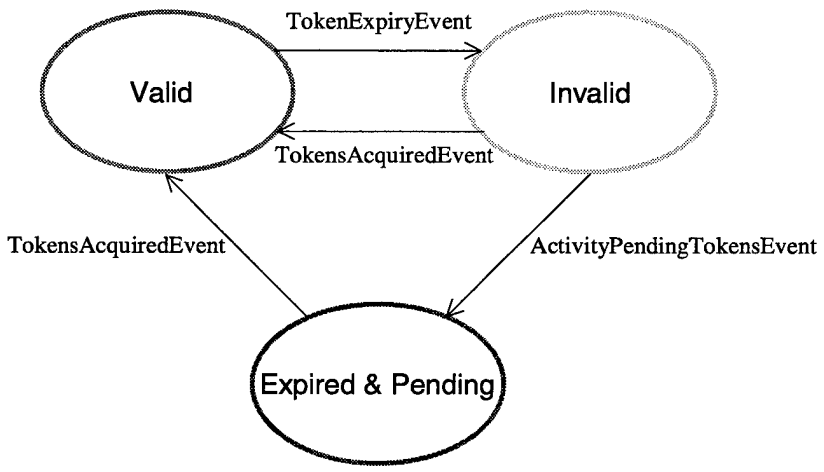


Figure 5.2: State Machine of the Tokens Indicator

[14] Currently, the system works under X windows, though it does not require that windowing system.

5.2.1.2 Hoard Walk Indicator

The finite state machine of Figure 5.3 drives the *Hoard Walk* indicator and contains four states: Inactive, Active, Stalled, and Suspend. The machine initially begins in the Inactive state. If the interface is then notified that a hoard walk has begun, the machine transitions to the Active state. Should Venus request advice, the machine transitions to the Stalled state until the user provides that advice and causes the machine to transition back to the Active state. When Venus announces the completion of the hoard walk, the machine transitions back to the Inactive state. Should the user request that Venus refrain from performing periodic hoard walks, the machine will transition to the Suspend state until periodic hoard walks are once again enabled.

5.2.1.3 Task Indicator

A smaller but slightly more complex pushdown automaton drives the Task indicator. This machine, which is shown in Figure 5.4, contains just two states (All Tasks Available and At Least One Task Unavailable), a counter, and a list of available tasks. At the end of each hoard walk, Venus announces the availability of each hoarded task owned by the user. This availability information is presented to the user in the *Task Information* window.

Suppose that Venus just completed a hoard walk and announced that all hoarded tasks were available. Further suppose that a hoarded file is invalidated (because, for instance, it was updated remotely). Venus would then announce that an object has become unavailable. In this message, it would also specify the task and the size of the object. The interface would now transition to the At Least One Task Unavailable state, set the counter of unavailable tasks to 1, remove this task from the list of available tasks, update the information about this task, and, finally, notify the user of this event. Now suppose that Venus must replace a hoarded file (for instance, due to space limitations). When this event occurs, Venus announces that the object has become unavailable as before. The interface, however, must be more careful in responding to this event. It must first determine whether or not this task was already unavailable. If the task was previously unavailable, then the system simply adjusts the percentage of the task available based upon the new information. If the task was previously available, then the system increments the counter of unavailable tasks, removes this task from the list of available tasks, and adjusts the percentage of the task available. Because the interface is already in the At Least One Task Unavailable state, it does not take a transition.

5.2.2 Advice Monitor

The primary purpose of the Advice Monitor is to act as a liaison between the user interface and Venus. As events arrive from Venus, the Advice Monitor translates the request into an appropriate procedure call within the interface, and then ships that call to

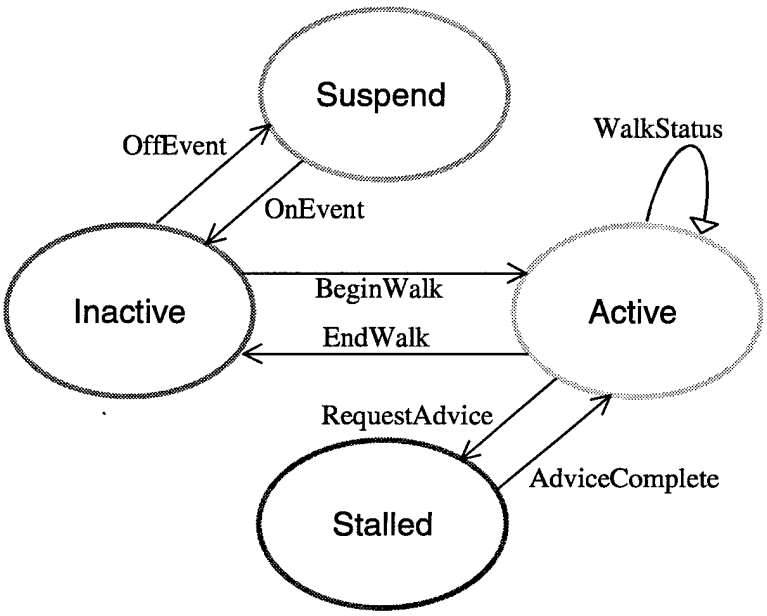


Figure 5.3: State Machine of the Hoard Walk Indicator

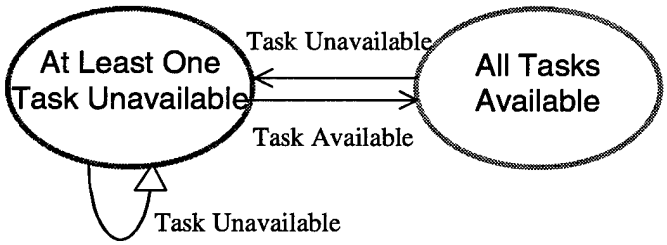


Figure 5.4: State Machine of the Task Indicator

the interface. As requests arrive from the interface, the Advice Monitor translates the request into an appropriate RPC2 call.

The Advice Monitor also has a number of secondary responsibilities. These include moving data files (the results of small questionnaires) into the Coda file system, processing certain usage statistics provided by Venus for the purpose of offering advice (as discussed in Sections 4.6.2 and 6.2), and controlling the Application Specific Resolvers, as described in [28].

The Advice Monitor uses a total of eight light-weight processes [45], or threads, to fulfill its obligations. Each thread and its primary responsibilities are described below:

Main: Performs initialization, loops waiting for incoming RPC requests and incoming messages from the user interface, and also dispatches other threads

CodaConsoleHandler: Handles messages arriving from the user interface, making the appropriate RPC to Venus (if required)

WorkerHandler: Handles RPCs arriving from Venus

ProgramLogHandler: Analyzes incoming program logs

ReplacementLogHandler: Analyzes incoming replacement logs, adding entries to the database

DataHandler: Moves data from temporary locations into permanent storage in Coda file system

EndASREventHandler: Watches for completion of Application-Specific Resolvers (ASRs) and returns the results to Venus

ShutdownHandler: Handles shutdown requests, closing connections and exiting the user interface

Two of these threads warrant further attention and will be discussed in the following chapter. These two threads, the Program Log Handler and the Replacement Log Handler, help to reduce the burden hoarding places on users.

5.2.3 Venus Modifications

Supporting the CodaConsole required modifications to Venus in two areas. First, I added the advice module, which implements the Venus side of the Translucent Caching API. The advice module exposes certain Venus *data* to the user and provides wrappers around the calls of the API. Second, I added hooks throughout Venus that invoke the wrappers within the advice module. These hooks expose certain *events* to the user by notifying the CodaConsole. This section describes both modifications.

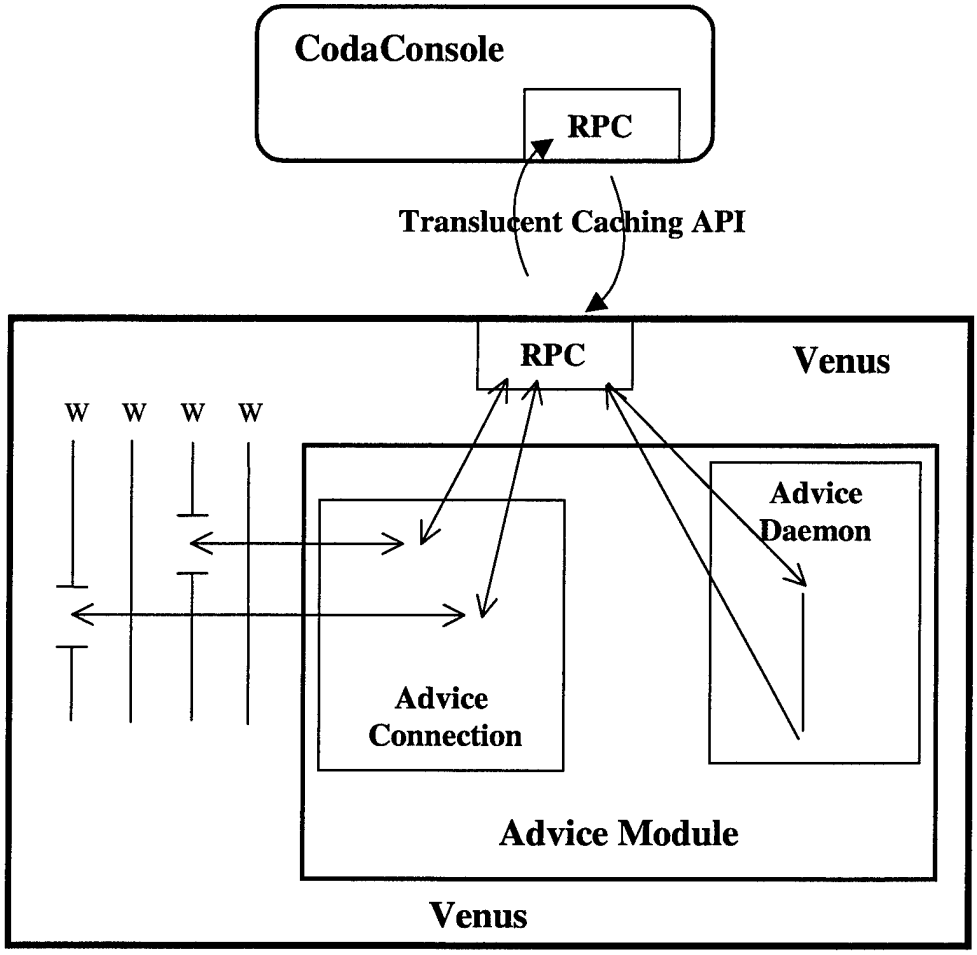


Figure 5.5: Relevant Venus Internals
This diagram shows a detailed view of the Venus component of Figure 5.1. The details include only those aspects of Venus relevant to supporting the CodaConsole.

5.2.3.1 The Advice Module

The advice module contains two pieces: the Advice Connection and the Advice Daemon. These two pieces, which are shown in Figure 5.5, handle the communication to and from the Advice Monitor respectively. The Advice Connection maintains a connection to a user's Advice Monitor and handles requests from Venus intended for that user. The Advice Daemon handles requests arriving from the user interface via the Advice Monitor.

5.2.3.1.1 The Advice Connection

Venus maintains at most one advice connection, the adviceconn class, per user. The connection is made on a per-user basis for security reasons—information flow to the

interface includes pathnames of referenced and hoarded files, information that would be inappropriate to release to other users.

Venus maintains at most one connection for each user simply to avoid confusion. It supports multiple simultaneous advice connections by different users with two caveats. First, other aspects of Venus (in particular, disconnected and weakly connected operation) assume a single-user workstation. Second, certain activities are restricted to authorized users, as defined by the user currently logged in on the console or the user specified (at startup) to Venus.

Venus requires that the connection be made from the local host. This decision avoids any security concerns associated with remote control of Venus, improves performance because the RPCs between Venus and the interface are now local, and avoids the possibility of the Advice Monitor becoming disconnected from Venus. Finally, it is consistent with the philosophy that the Advice Monitor is logically a part of Venus.

An advice connection maintains a small amount of state, a subset of which is shown in Figure 5.6 and described below.

User Lock: Controls communication with the interface and, thus, the user

Connection Information: Allows the Advice Daemon to establish a connection from Venus to the Advice Monitor, includes the state, hostname, port, and handle

Process Group ID: Prevents deadlock in the case that a request made on behalf of the user generates a request back to the user

Interest Array: Allows Venus to suppress event notifications of which the CodaConsole is not configured to notify the user, an optimization that potentially removes a local RPC from the critical path of the call.

Logging State: Allows Venus to log each file access and the program that made it, as well as each cache replacement, includes file descriptors, pathnames, and line counts

Methods: Provides a set of methods that are, essentially, wrappers around the API; provides a clean interface to the rest of Venus; and ensures that all locking and counting occur as expected

The methods provided by the advice connection wrap around half of the Translucent Caching API, the half that is serviced by the Advice Monitor. The calls defined by this API, which are summarized in Figure 5.7 below and described in detail in Appendix A, expose certain events to the Advice Monitor, and thus the user.

The Translucent Caching API, which includes all of the calls shown in Figure 5.7 as well as those shown later in Figure 5.8, is key to the CodaConsole's independence from Venus, and by extension from Coda. Any distributed system that offers high availability should be able to take advantage of the CodaConsole simply by supporting this API.[15]

[15] Of course, not all distributed file systems offering high availability will necessarily support all of the conceptual ideas of Coda. Any conceptual differences would obviously need to be handled separately.

```
class adviceconn {
    // State related to controlling access to the user
    struct Lock userLock;

    // State related to the connection to the Advice Monitor (A.M.)
    AdviceState state;              // of this connection
    char hostname[MAXHOSTNAMELEN]; // on which A.M. is running
    unsigned short port;           // on which A.M. is listening
    RPC2_Handle handle;            // to active RPC connection

    int pgid;                      // of the Advice Monitor

    // State related to events in which the user is interested
    int InterestArray[MAXEVENTS];

    // State related to logging various events
    FILE *programLog, *replacementLog;
    char programLogName[MAXPATHLEN], replaceLogName[MAXPATHLEN];
    int numProgLines, numReplaceLines;

    .
    .
    .

  public:
    // Methods
    void TokensAcquired(int);
    void TokensExpired();
    void ServerAccessible(char *);
    void ServerInaccessible(char *);

    .
    .
    .
};
```

Figure 5.6: State Maintained by the `adviceconn` Class

This figure shows a subset of the state maintained by the `adviceconn` class. This state includes a lock that controls access to the user, a group of variables that maintain information regarding the current connection to this user's Advice Monitor, an array that records the user's interest in each event, and a group of variables that support logging various events. The complete definition of this class is available in the Coda sources in the file `coda-src/venus/adviceconn.h`.

Remote Procedure Call	Description
LostConnection	Announces that the Advice Monitor has lost its connection
TokensAcquired	Announces that the user has acquired tokens
TokensExpired	Announces that the user's tokens have expired
ServerAccessible	Announces that a server is accessible
ServerInaccessible	Announces that a server is no longer accessible
ServerConnectionWeak	Announces that the connection to a server is weak
ServerConnectionStrong	Announces that the connection to a server is strong
NetworkQualityEstimate	Announces the current server bandwidth estimates
Reconnection	Requests that a reconnection questionnaire be presented to the user
ReadDisconnectedCacheMiss	Requests advice regarding a read disconnected cache miss
WeaklyConnectedCacheMiss	Requests advice regarding a weakly connected cache miss
DisconnectedCacheMiss	Requests that a disconnected cache miss questionnaire be presented to the user
HoardWalkAdviceRequest	Requests advice regarding a hoard walk
HoardWalkBegin	Announces that a hoard walk has begun
HoardWalkStatus	Announces the progress of a hoard walk
HoardWalkEnd	Announces that a hoard walk has ended
HoardWalkPeriodicOn	Announces that periodic hoard walks have been enabled
HoardWalkPeriodicOff	Announces that periodic hoard walks have been disabled
ObjectInConflict	Announces that a file system object is in conflict
ObjectConsistent	Announces that a file system object has been repaired
ReintegrationPendingTokens	Announces that a reintegration is pending, waiting for the user to obtain authentication tokens
ReintegrationEnabled	Announces that a reintegration is ready to proceed
ReintegrationActive	Announces that a reintegration is in progress
ReintegrationCompleted	Announces that a reintegration has completed
TaskAvailability	Announces the availability of hoarded tasks
TaskUnavailability	Announces the unavailability of an element of a hoarded task
ProgramAccessLogAvailable	Announces the availability of a program access log
ReplacementLogAvailable	Announces the availability of a replacement log
InvokeASR	Invokes application-specific resolution

Figure 5.7: Calls of the API Serviced by the Advice Monitor

Remote Procedure Call	Description
NewAdviceService	Announces the existence of a new Advice Monitor
RegisterInterest	Records user's interest in various events
GetCacheStatistics	Retrieves current space information
OutputUsageStatistics	Outputs disconnected usage statistics
SetParameters	Modifies value of internal variables
ResultOfASR	Provides result of application-specific resolver
ImminentDeath	Announces imminent death of Advice Monitor

Figure 5.8: Calls of the API Serviced by Venus

Another interesting issue to consider is what would be required to move Venus into a kernel module, as was done with production versions of AFS. Currently, Venus uses RPC2 to support the Translucent Caching API, but this choice is not critical to its success. Should Coda ever be moved down into the kernel to improve performance, this RPC2 interface can easily be replaced by system calls, leaving the CodaConsole to run in user space where it clearly belongs.

5.2.3.1.2 The Advice Daemon

The implementation of the Advice Daemon is straightforward. The main routine of this Venus process loops waiting for a request to arrive and then handles that request. The daemon is responsible for implementing the seven calls of the Translucent Caching API serviced by Venus. These calls are summarized in Figure 5.8 and described in detail in Appendix A. They expose certain data to the Advice Monitor.

5.2.3.2 Venus Hooks

Venus informs the user of events that occur during normal operation. These events include token expiration, hoard walks, cache misses, and task availability. For each event, I added a relatively straightforward hook. The hook first identifies which user (or users) to notify. It then makes the appropriate call to that user's Advice Connection, including all required information. Each hook generally requires only tens of lines of code to implement. The challenge in adding this functionality lay largely in tracking down where in the code to add the appropriate calls to the API. Occasionally, collecting the necessary information required a minimal amount of additional work.

Hooks were added to the routines shown in Figure 5.9 below. This table serves two purposes. The first is to document the location of these hooks for future Venus developers. The second, and more important one, is to show that these hooks correspond roughly one-to-one to the API calls of Figure 5.7. Only a few API calls require more than one hook and none required more than three hooks. Generally, all of the hooks appear in the obvious location (to someone familiar with Venus internals).

Event	Class::Method
LostConnection	adviceconn::NewConnection
TokensAcquired	userent::SetTokens
TokenExpiry	userent::Invalidate
ServerAccessible	srvent::ServerUp
ServerInaccessible	srvent::ServerError, srvent::Connect
ServerConnectionStrong	srvent::InitBandwidth, srvent::GetBandwidth
ServerConnectionWeak	srvent::InitBandwidth, srvent::GetBandwidth
NetworkQualityEstimate	srvent::InitBandwidth, srvent::GetBandwidth
Reconnection	volent::TakeTransition
ReadDisconnectedCacheMiss	fsdb::Get
WeaklyConnectedCacheMiss	fsdb::Get
DisconnectedCacheMiss	fsdb::Get
HoardWalkAdviceRequest	hdb::Walk
HoardWalkBegin	hdb::Walk
HoardWalkStatus	hdb::StatusWalk, hdb::DataWalk
HoardWalkEnd	hdb::Walk
HoardWalkPeriodicOn	hdb::Enable
HoardWalkPeriodicOff	hdb::Disable
ObjectInConflict	fsdb::Get
ObjectConsistent	lrdb::CheckLocalSubtree, vproc::do_ioctl
ReintegrationPendingTokens	volent::TakeTransition
ReintegrationEnabled	volent::TakeTransition
ReintegrationActive	volent::Reintegrate
ReintegrationCompleted	volent::Reintegrate
TaskAvailability	hdb::PostWalkStatus
TaskUnavailability	fsobj::Kill
ProgramAccessLogAvailable	::VTALRM, fsobj::Get
ReplacementLogAvailable	::VTALRM, fsobj::ReclaimFsos, fsobj::ReclaimBlocks
InvokeASR	fsdb::Get

Figure 5.9: Location of Hooks Supporting Translucent Caching
This table shows the location of hooks that support translucent caching. For each call in the Translucent Caching API, I show each method of Venus that calls it. Note the nearly one-to-one mapping between these hooks and the routines provided by the API (refer to Figure 5.7).

5.3 Use of Translucent Caching API

In this section, I will make the API more concrete by exploring three scenarios. The first one illustrates the initial sequence of messages that startup the CodaConsole. The second one illustrates a basic event notification, in this case a notification that the user's tokens have expired. The third one illustrates the conversation that occurs between Venus and the CodaConsole during a hoard walk.

5.3.1.1 Initialization

Venus begins running during the Unix startup sequence (e.g., out of /etc/rc). Some time later, the user logs in, starts X, and eventually starts the CodaConsole. Before Venus and the CodaConsole, two independent processes, can cooperate to make caching translucent to the user, communication between them must be established. Further, because the CodaConsole also consists of two cooperating processes, communication must be initialized for them as well. As described previously and illustrated in Figure 5.1, communication between Venus and the CodaConsole occurs via a pair of RPC connections while communication between the two components of the CodaConsole occur over a pair of pipes. In this section, I describe how this initialization, which is illustrated in Figure 5.10, occurs.

Because Venus starts long before the CodaConsole, it is the CodaConsole's responsibility to initiate contact with Venus. The main thread of the Advice Monitor initiates contact with Venus during its initialization phase. It issues the NewAdviceServiceAvailable RPC2 call, shown in Figure 5.8, to Venus. Parameters to this call include connection information, version information, and the process group ID of the Advice Monitor. As described previously, the process group ID is used to prevent deadlock (see Section 5.2.3.1.1). The version information is used to detect version skew between the Advice Monitor and Venus, and thus ensure that they are using the same API. The connection information includes the (ephemeral) port on which the Advice Monitor will be listening for the return call from Venus.

Upon receiving this call, Venus initializes the adviceconn for the user based on the incoming parameters. Venus also sets the state of this connection to indicate that it is waiting. Venus then returns from the RPC and signals the initialUserSync variable. At this point, we have unidirectional communication between Venus and the Advice Monitor. The Advice Monitor can issue RPCs to Venus, but Venus cannot yet issue RPCs to the Advice Monitor.

Two more processes must still complete before our initialization is finished. The user interface must be started and Venus must return the connection to complete the bidirectional communication between it and the Advice Monitor. Although either of these things could occur next, I will assume that the interface is started first and that Venus' return connection occurs second.

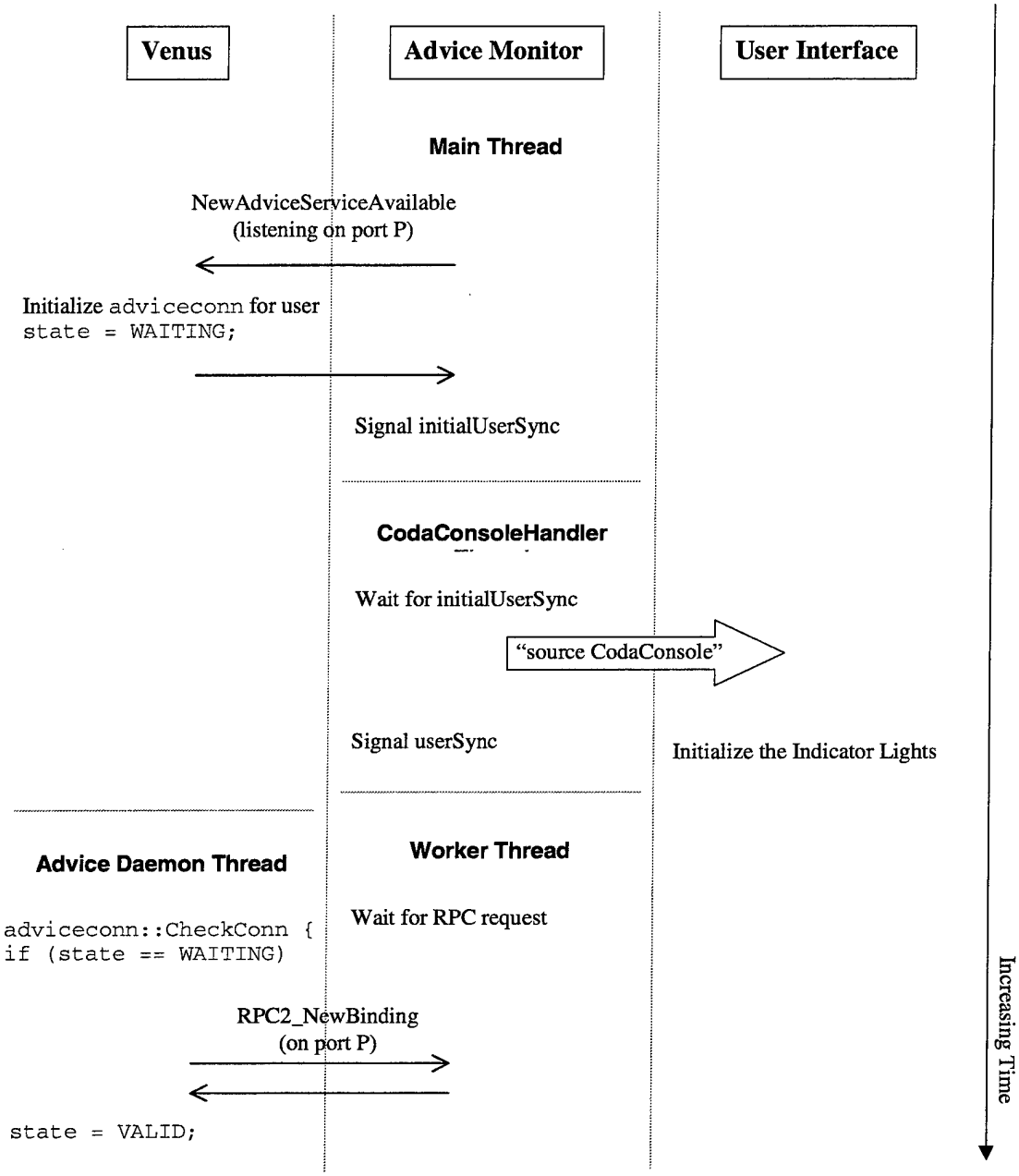


Figure 5.10: Initialization

This figure illustrates the initialization process. Each process is shown in its own column. Interprocess communication is shown as arrows crossing the columns. The thin arrows represent remote procedure calls. The thick arrow represents a pipe.

Signaling the initialUserSync variable allows the CodaConsoleHandler thread to wake up. This thread is responsible for starting the user interface and, later, for processing input received from its pipe[16]. After waiting for this signal, the CodaConsoleHandler issues the command that starts the user interface. It then enters a loop that waits for input from the user interface and processes that input. The user interface is now active.

The final step in the initialization procedure is to establish the connection from Venus to the Advice Monitor. Eventually, the Advice Daemon (a lightweight thread within Venus) discovers that an adviceconn is in the waiting state. It will then issue a RPC2_NewBinding request to the Advice Monitor on the port specified in the original NewAdviceServiceAvailable message. Once this call returns, the state of the connection is set to Valid and Venus can now issue RPCs to the Advice Monitor.

The initialization of communication between all three processes is now complete. Venus can issue event notifications to the interface. The interface can issue requests for information. As important cache-related events occur, the system will notify the user. Caching is now translucent.

5.3.1.2 Basic Event Notification

The majority of calls from Venus to the CodaConsole interface via the Advice Monitor are basic event notifications. In this section, I will walk through the messages involved in performing such a notification. I will use token expiry as my example. Figure 5.11 shows an illustration of this notification.

During token expiry, Venus calls the TokenExpiry RPC2 wrapper. This wrapper checks that the user is interested in receiving notification of this event. If so, it locks the userLock stored in the user's adviceconn structure. It then issues the TokenExpiry RPC. This call is sent to the Advice Monitor. The Advice Monitor translates this call and outputs a message, "TokenExpiryEvent", to the CodaConsole interface via the appropriate pipe. After flushing the pipe, it returns from the RPC2 call. The wrapper routine then releases its lock, checks for errors, and returns to Venus.

Upon receiving the "TokenExpiryEvent" from its input pipe, the user interface first looks up the current configuration of this event. Based upon this configuration, it then notifies the user that this event has occurred. By default, this notification will turn the *Tokens* indicator light yellow.

---

[16] The pair of pipes, which allow the Advice Monitor and the CodaConsole to communicate, were initialized very early in the startup process. Before the user interface process was forked, two pipes were created. The Advice Monitor maintained file descriptors for one end (the read side of one and the write side of the other) of each pipe and mapped standard input and output for the user interface to the other ends of these pipes. One pipe allows the Advice Monitor to send messages to the user interface. The other allows the user interface to send messages to the Advice Monitor.

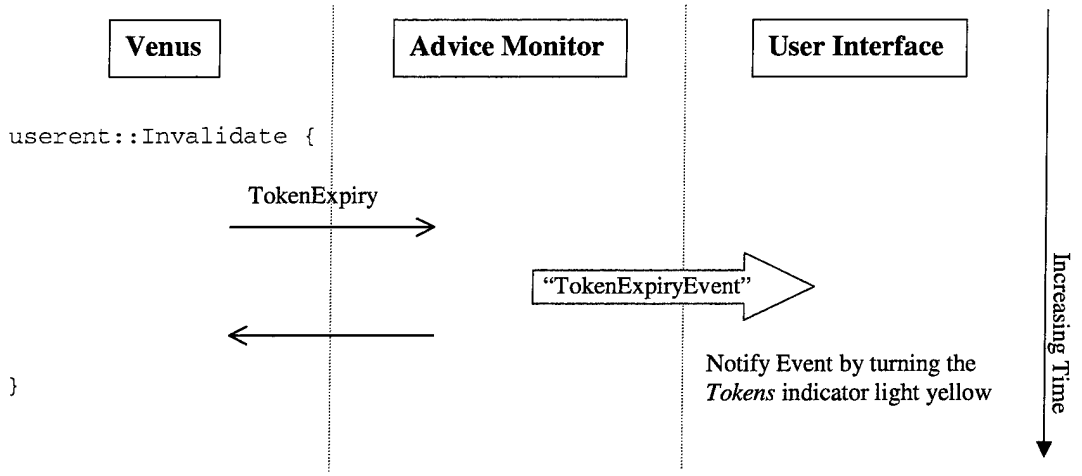


Figure 5.11: Notification of Token Expiration

This figure illustrates a basic event notification. The event shown is the token expiry event. Each process is shown in its own column. Interprocess communication is shown as arrows crossing the columns. The thin arrows represent remote procedure calls. The thick arrow represents a pipe. Return from the RPC occurs immediately after flushing the pipe.

5.3.1.3 Traffic During a Hoard Walk

One of the more complicated interactions occurs as Venus performs a Hoard Walk (see Section 2.2.2). In this section, I describe the many different messages that go back and forth between Venus and the CodaConsole. This conversation is illustrated in Figure 5.12. For the purposes of this example, I assume that Venus requests advice from the user.

When Venus first begins a hoard walk, it notifies the CodaConsole. Venus then performs a status walk, during which it equilibrates the cache fetching all necessary status blocks. After completing the status walk, it informs the CodaConsole as to its progress. Then, it generates a file containing the information for which it would like advice from the user and initiates the request for advice. To request advice, it triggers an RPC to the Advice Monitor, the HoardWalkAdviceRequest call. In this call, it includes the name of the input file it previously generated and the name of the file in which it expects to find the user's advice. When the Advice Monitor receives this RPC, it forwards the request to the CodaConsole and then waits on the hwSync variable.

The user interface notifies the user of the request as specified by the configuration for this event (see Section 4.2.1). It then waits for the user to provide advice. Once the user has done so, it sends a message to the Advice Monitor informing it that advice is now available. This causes the CodaConsoleHandler thread to signal the hwSync variable.

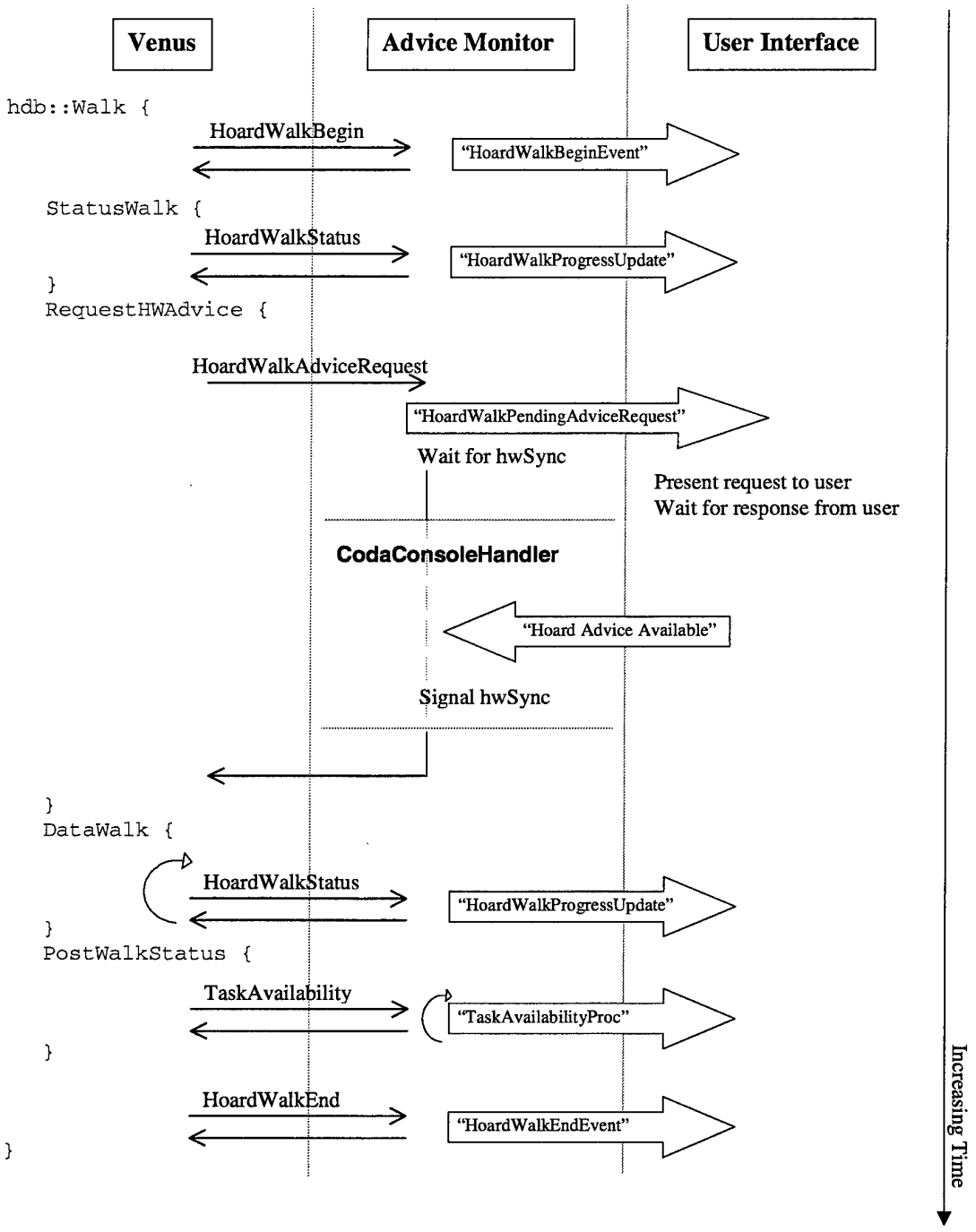


Figure 5.12: Traffic During a Hoard Walk

This figure illustrates traffic that occurs during the process of a hoard walk. Each process is shown in its own column. Interprocess communication is shown as arrows crossing the columns. The thin arrows represent remote procedure calls. The thick arrows represent pipes.

The HoardWalkAdviceRequest RPC then returns to Venus with the resulting advice. Venus is finally able to proceed with the data walk during which it fetches any data that it needs. During this data walk, Venus informs the CodaConsole of its progress using the HoardWalkStatus RPC. This call, another basic event notification, drives the progress meter shown in the *Hoard Walk Information* window (see Figure 4.12).

Upon completing the data walk, Venus has two additional responsibilities. First, it determines the availability of all hoarded tasks. It sends this information to the CodaConsole in the TaskAvailability RPC. Using this information, the Advice Monitor issues messages, one per task, to inform the user interface of the availability of each task. Venus then notifies the CodaConsole that it has completed the hoard walk.

This completes the process of a hoard walk. As you can see, many different messages can be triggered during a single hoard walk. The total number of messages possible is bounded at 104. Four notify the CodaConsole of the beginning of the walk, of the advice request, of the task availability, and of the end of the walk. The 100 additional messages represent the numerous progress updates. One hundred represents an upper bound for this number because progress messages are only sent when the percentage of the hoard walk completed increases by at least 1.[17]

5.4 Summary

This chapter has described the implementation of the CodaConsole. This implementation included the user interface itself, the Advice Monitor, and a number of modifications to Venus. The implementation uses the Translucent Caching API to support communication between Venus and the CodaConsole.

At the beginning of this chapter, I discussed my decision to place the functionality of the user interface external to Venus as much as possible. It isolated me from concurrent Venus development activities, thus allowing a more stable development environment for this research! More importantly, however, it made the interface independent of Venus, and therefore of Coda. Other file systems offering high availability could easily adopt the CodaConsole simply by implementing the API. Overall, I was able to keep Venus modifications to a minimum. I added just 2500 lines of code, making Venus approximately 3% larger than it would otherwise be. Had I implemented the functionality completely within Venus I would have added another 12,500 lines, increasing the size of Venus by more than 15%. Hindsight suggests that this was a wise decision.

[17] The original design informed the user interface of the current progress after the completion of each individual fetch. However, because the system may fetch numerous small files, the number of RPC messages quickly became unreasonable. The revised design limits the total number of messages sent to equal the maximum number the user can distinguish; any messages beyond that represent wasted effort.

6 Reducing the Burden on Users

Translucent caching must carefully balance the cost it imposes on the user with the benefits it offers. In designing the CodaConsole interface, I tried to reduce the burden placed on users wherever possible. The burden of translucent caching largely arises from occasions when the system requests advice. In particular, users are burdened by the need to define tasks and by the need to advise about network usage. If translucent caching requires its users to focus too much of their attention on managing the file cache, their productivity will suffer—a situation that is clearly not our goal. I employ three techniques to reduce the burden of translucent caching.

Two of these techniques reduce the burden required to define tasks and, more generally, to maintain the hoard database. One technique allows the system to automatically produce hoard profiles for programs in which the user has indicated an interest. It reduces the burden by lessening the amount of detail the user must provide. Another technique allows the system to offer advice to users regarding their task definitions. It reduces the burden by identifying mistakes in the user's hoard profile. The remaining technique reduces the number of occasions for which Venus requests advice. It models user patience in an effort to automate the process of determining whether or not to fetch an object over a weak network connection. In this chapter, I present each of these techniques.

6.1 Hoarding Programs Automatically

One burden imposed by the original Coda system was the need to produce hoard profiles for programs. Users created these profiles with the aid of a primitive tool called spy (as described in Section 2.3.3). The spy tool simply recorded all Coda references made during its execution. One problem with the tool arose because any process, not just the one on which the user was attempting to spy, could have made the observed references. Sorting out the references was the user's job. Another burden arose because, after generating the list of references, the user then had to post-process that list (using a procedure that assumed knowledge of Coda file identifiers) to create a profile acceptable to the hoard tool (see Section 2.3.2). Users managed the resulting profiles by hand, updating them occasionally as new requirements were discovered. Unfortunately, new discoveries were typically the result of a disconnected or weakly connected cache miss.

Thus, the first technique, used to reduce the burden that translucent caching places on users, focuses on the problem of identifying files on which programs depend. The goal is

to automate the process of generating program profiles. It reduces the burden that hoarding, and thus translucent caching, places on the user because it takes over a task that was originally the user's responsibility. Further, it improves the accuracy of the generated program profile because it monitors the program continuously. Thus, the automatic monitor can identify references that the user might have missed because the program accesses the file only under certain, infrequent conditions. In this section, I will present the design and implementation of the *automatic program reference monitor.*

6.1.1 Design Requirements

The basic design for the automatic program reference monitor requires the system to observe every Coda reference made by various programs executed by individual users, and to do so continuously. The monitor should also distinguish references made to "program data" from those made to "user data", maintaining information only about program data. (Program data includes font files, style files, and mode files; user data includes source files.) This requirement allows program profiles to be used by many different tasks because it means that they do not depend upon an individual execution of that program.

Because the system is monitoring every reference users make, we must be careful to maintain the user's privacy. Users should not be given information about what files or programs another user accesses. For this reason, I limit the distribution of reference logs to only the user generating the original reference. Sharing of program profiles is not explicitly supported by the interface, though users can do so externally by giving other users access to the files that contain this information.

Further, because monitoring occurs on every file system request, we must be careful to maintain the system's efficiency. Whatever monitoring the system introduces, it must be fast because it occurs on the critical path of every file system request and will have a direct impact on the resulting system's performance.

6.1.2 Instrumentation

An important goal for the automatic program reference monitor was to support continuous monitoring. Continuous monitoring offers a significant advantage to the user. It allows the system to automatically detect the need for files that are referenced only infrequently, reducing the likelihood of cache misses. However, continuous monitoring risks serious performance degradation. For this reason, it became a primary consideration in the implementation. The system minimizes the burden such monitoring places on Venus by asking Venus to do as little work as possible and by requiring the Advice Monitor to pick up the burden.

Continuous monitoring requires Venus to inform the Advice Monitor of every file reference, specifying which program made the reference. The obvious design for such a

```
gnu-emacs 4471
<7f000259.246.23a> 4471 4471
<7f000259.136a.3ad> 4471 4471
gmake 4536
<7f000259.246.23a> 4536 4536
gnu-emacs 4471
<7f000259.246.23a> 4471 4471
.
.
.
```

Figure 6.1: Reference Log Fragment

This figure shows a fragment of a reference log. A reference log contains two different types of entries. One type of entry contains the name and process group identifier of the executable making references to the Coda file system. The first line of this log fragment shows an example of such an entry, in this case for `gnu-emacs`. The other type of entry contains the file identifier of a Coda file object as well as the process identifier and process group identifier of the process making the reference. As seen in this fragment, the `gnu-emacs` process (process group identifier 4471) referenced two file objects (to `<7f000259.246.23a>` and `<7f000259.136a.3ad>`) before being interrupted by the `gmake` process (process group identifier 4536).

system would extend the API by one additional call that would inform the Advice Monitor of these references. Upon receiving such a call, the Advice Monitor would record that the given program referenced the specified file. This design, although simple, imposes too great a burden on Venus. It adds a local RPC call to the critical path of every single file reference made by the user. For this reason, my system batches these requests and periodically ships a group of them to the Advice Monitor.

6.1.2.1 Venus Contribution

To support automatic program monitoring, Venus must record every reference and identify the program making that reference. Thus, when an Advice Monitor begins running, Venus opens a reference log. For each subsequent file reference, Venus outputs an entry containing the file identifier referenced as well as the PID (the Unix process identifier) and PGID (the Unix process group identifier) of the process making that reference. In addition, it must record the name of the program making the reference because that program may exit before post-processing begins. Periodically, Venus starts a new log and then signals the Advice Monitor.

A fragment of a reference log is shown in Figure 6.1. This log records that the `gnu-emacs` program (with PGID = 4471) referenced two files with Coda FIDS of <7f000259.246.23a> and <7f000259.136a.3ad>. Then, the `gmake` program (with PGID = 4536) referenced the file with FID <7f000259.246.23a>. After that, the `gnu-emacs` program began executing again and referenced <7f000259.246.23a> again.

Venus exploits locality of reference by maintaining a record of the most recently active program and its process group. It only outputs the name of the active program when the PGID of the current reference differs from that of the previous reference.

The beauty of this design is its simplicity. It exploits the standard I/O buffer, by using fprintf(), to avoid adding too much overhead to the critical path of servicing file references. Periodically (currently every 1000 references), a file reference must pay the cost of a local RPC call to inform the Advice Monitor that a reference log is available for post-processing. For all other file references, the overhead imposed by continuous program monitoring is the cost of a fprintf() to the reference log file and, if the PGID differs from the previous PGID, the cost of determining the name of the currently active program.

The overhead imposed by this monitoring as well as the monitoring required for providing advice to users, which is discussed in the next section, is small. To estimate this overhead, I performed a simple experiment that compares the time required to compile the Coda cache manager with and without the CodaConsole interface running, as calculated based upon the average of five compiles. This simple experiment showed that the overall overhead was less than 7%. (Without the interface, the compile took an average of 378 seconds; with it, the compile required 403 seconds.) Because the overhead was low, I did not perform a more involved experiment to determine the contributions of individual components of the interface.

6.1.2.2 Advice Monitor Contribution

The Advice Monitor's job begins once Venus signals that a reference log is available. Once this event occurs, the Advice Monitor wakes the ProgramLogHandler thread. This thread examines the log for references made by programs *under watch*. A program is said to be under watch if it is included in a task definition (see Section 4.10). Note that this includes programs contained in any task definition, including those not currently hoarded. The rationale for this decision is simply that the user has indicated an interest in this program and could hoard it at any time. We want to have built a good profile prior to the point at which the user needs it.

As described in the previous section, the reference log contains two types of entries. The first type identifies the active program. The second type identifies both the referenced file and the program responsible for making the reference. As the Advice Monitor post-processes a reference log, it maintains a record of the currently active program and whether it is under watch. If the currently active program is not under watch, then any references it made are skipped. If, however, the currently active program is under watch, then the Advice Monitor ensures that all subsequent references to *program data*, omitting duplicates, have been recorded in the program's profile.

The Advice Monitor distinguishes program data from *user data* using a simple heuristic that exploits the logical structure of the file system. This heuristic recognizes that users typically understand the layout of user and project volumes that they use, whereas they typically do not understand the layout of program volumes that they use. Thus, if the user includes a file or directory in a data definition (see Section 4.10), then the Advice Monitor

considers all data stored in the volume owning that file or directory to be user data. Any references to volumes not considered user data are deemed program data and will be recorded in the program's profile.

This distinction between program data and user data is a key property of automatic program hoarding. It allows users to reuse program definitions in different tasks. For instance, if my Advice Monitor created a program profile for `latex` while I was writing a paper to submit to a conference, I can feel confident in reusing that program profile in a definition to support writing a workshop submission. The ability to reuse program profiles in subsequent task definitions reduces the burden placed on users of translucent caching.

This technique is flexible in handling different logical file system structures as well. For instance, suppose a different site places all program binaries in a volume named `s.local.bin`, all libraries in a volume named `s.local.lib`, and all font files in a volume named `s.local.fonts`. The heuristic would work equally well for this layout.

The limitation to this heuristic is its reliance upon a logical, volume-based structure. A file system that does not support volumes would obviously be unable to exploit this heuristic. However, as long as such a file system has a logical separation between user data and program data, then a similar heuristic should allow the system to distinguish between the different types of files. A file system that does not separate user data from program data in any logical way, however, will be unable to use this or any other heuristic to make this distinction. Its users will be prevented from reusing their program definitions, though they will still be able to exploit automatic program hoarding to build profiles for individual tasks.

6.2 Advising about Hoard Contents

A second technique, incorporated to help reduce the burden of user-assisted hoarding, allows the system to provide users with advice about their task definitions. The presentation of this advice was introduced in Section 4.6.2. Here, I will focus on the heuristics used to identify those files and directories that should be brought to the user's attention.

The heuristics take one of two forms. The first type identifies files that the user is not hoarding, but probably should be. Heuristics of this first type identify files that are accessed infrequently but repeatedly, files that the user has successfully accessed during a disconnected session but has not hoarded, and files that the user could not access during a disconnected session because the file was not cached. The second type identifies files that are currently hoarded, but probably should not be. These heuristics identify files that have not been accessed during recent disconnected sessions or that have been accessed infrequently during disconnected sessions.

In this section, I will begin by presenting each heuristic and the intuition behind it. I will then briefly describe how the system collects the information necessary to identify these files.

6.2.1 Heuristics

I will present the five heuristics used to identify files that the user may want to consider adding or removing from the task definitions. These heuristics are all conceptually simple, but identify classes of files that users have been known to forget.

Infrequent, but Repeated, Access—This heurisitic looks for files that are accessed infrequently, but repeatedly, over long periods of time. A good example of such a file is the `.xinitrc` file. This file is used every time the user starts X. After starting X, the user is unlikely to access this file again for days, or even weeks. This usage pattern makes the `.xinitrc` file a prime target for replacement, unless the user has included it in a task definition. Unfortunately, because it is accessed indirectly by a program, it is also an easy file for users to forget. Thus, this heuristic analyzes the user's reference stream specifically looking for files that are accessed infrequently, but repeatedly, over a long period of time. It extends the horizon by maintaining information about files that are replaced from the cache. This heuristic may identify objects identified by the *unlucky reference* heuristic (below) if the object experiences a cache miss, but can identify such an object even if it does not. Further, this heuristic will identify files whose priority may need to be increased as well as files that simply need to be included in a definition.

Unlucky Reference—This heuristic identifies files that experienced a cache miss. Although this is an obvious heuristic, it serves to remind users of files that they may want to add to their task definitions. It relies on locality of reference: if the user attempts to access a file, she is likely to access it again in the future. Specifically, this heuristic identifies all files which experienced cache misses, while the CodaConsole was running, since the last time the interface offered the user advice.

Lucky Disconnected Reference—This heuristic analyzes the user's reference stream looking for files that were accessed during a disconnected session, but for which the user has not included in a task definition. Such files were successfully accessed during disconnected operation by simple good fortune. Had other references caused the file to be replaced, the system would not have known to refetch the file in preparation for disconnected operation. This heuristic identifies all cached, but unhoarded, files accessed during any disconnected session. As above, this heuristic assumes locality of reference.

Infrequent Disconnected Use—This heuristic identifies files that are used only infrequently during disconnected sessions. Specifically, it identifies files that have been used in fewer than 50% of disconnected sessions, though this value should eventually become a user-configurable parameter (see Section 9.2.2.1). These files are ones included in a task definition, but for which it appears that inclusion is unnecessary. Such files may have been heavily used for a period of time, but are now no longer needed and thus can be removed. Their inclusion may be taking cache space away from files that have a lower priority, but that are used more frequently. This heuristic brings this anomaly to the user's attention.

Ancient Disconnected Use—This heuristic identifies files that have not been used during the two most recent disconnected sessions, though this number will eventually become a user-configurable parameter (see Section 9.2.2.1). As in the previous heuristic, these files have an augmented priority, but they appear not to need the increased priority. These files may belong to a project that has been finished, but whose tasks have not been removed. Once again, their increased priority may take cache space away from lower priority objects that are used more frequently.

6.2.2 Instrumentation

Each of the heuristics described in the previous section depends upon statistics gathered during use of the system. In this section, I will briefly describe how I instrumented Venus and the Advice Monitor to gather these statistics.

6.2.2.1 Replacement Statistics

The instrumentation necessary to collect the statistics supporting the *infrequent, but repeated access* heuristic is complicated by the fact that these objects may not be referenced again before they are replaced (e.g., the `.xinitrc` file). Thus, the data necessary to identify this class of files must persist across cache replacements. The basic strategy, then, is to maintain a small amount of information about previously cached objects.

The intuitive design to support this data collection would be to augment the data structures internal to Venus to allow Venus to maintain ghost entries for previously cached objects. Unfortunately, this design suffers from a fundamental problem. If Venus were to maintain this information, it would need to store it in recoverable virtual memory (RVM). Even though the amount of information stored for each previously cached file may be quite small, the number of previously cached files is likely to be large enough that the overall storage requirement would overwhelm Venus' modest RVM resources.

The actual design, similar to that employed by automatic program hoarding above, requires Venus to log data about each file system object that it replaces. This data contains the Coda file identifier of the object as well as indications of whether the status or the data (or both) were replaced. Periodically, Venus starts a new log and then signals the Advice Monitor that the old log is available for post-processing. The Advice Monitor's ReplacementLogHandler thread post-processes this data, building a database containing the pathnames of replaced objects and counts of the number of times their status and data have been replaced. Once an object's status or data has been replaced 10 times, the object is brought to the user's attention.

This revised design places the responsibility for collecting the data on Venus and the responsibility for maintaining the data on the Advice Monitor. It minimizing the amount of effort required to collect the data, since that effort is likely to be on the critical path for servicing file system requests. In fact, the overhead of this data collection is small, as was

discussed previously in Section 6.1.2.1. The burden of maintaining the data falls to the Advice Monitor, which is the appropriate place from a software engineering perspective. The data is stored in a database that is stored on the local disk rather than in Venus' RVM space. Overall, the revised design is far superior to the intuitive design while remaining functionally equivalent.

6.2.2.2 Cache Miss Statistics

The instrumentation necessary to collect the statistics supporting the *unlucky reference* heuristic is straightforward. Each time Venus triggers a cache miss query during disconnected, weakly connected, or read disconnected operation (see Sections 4.6.1.1, 4.6.1.2, and 4.6.1.3, respectively), the interface keeps track of the pathname. On demand, the interface can generate a list of files that the user attempted to access, but for which Venus did not have cached data.

6.2.2.3 Disconnected Usage Statistics

The next three heuristics (*lucky disconnected reference*, *infrequent disconnected use*, and *ancient disconnected use*) rely on the user's disconnected behavior. To support these heuristics, I had to instrument Venus to collect disconnected usage statistics. Each time a volume transitions to disconnected operation, it records the current reference count in RVM. When a file is accessed during the disconnection, its last reference record is updated automatically. Upon reconnection, Venus first determines whether or not the disconnected session was *eligible*. Eligibility depends upon whether or not the user made any references during the period of time the volume was disconnected (i.e., if the current reference counter is higher than the disconnected reference counter). If the disconnected session is not eligible, then the user did not use any objects from this volume so the system does not collect statistics about this session. If the disconnected session is eligible, however, then Venus examines every cached object from the reconnected volume and compares its last reference record to the reference count at the point of disconnection. If the last reference count is higher than this value, the object was used during the disconnected session; otherwise it was not.

Informing the Advice Monitor of these statistics upon reconnection would involve a substantial amount of data. To avoid a substantial performance penalty, Venus produces these statistics only upon demand. When the Advice Monitor requests disconnected usage statistics, Venus generates a list of cached objects meeting certain disconnected usage criteria. These data are then used to identify objects for the *lucky disconnected reference, infrequent disconnected use*, and *ancient disconnected use* heuristics as described in Section 6.2.1.

6.3 Modeling User Patience

During the course of weakly connected operation, the system may encounter situations in which a file referenced by the user is not available in the cache. At times, that file may be important enough that the user is willing to wait a substantial amount of time for it to be fetched; at other times, the user may not be willing to wait long at all. Ideally, the system could distinguish those files for which a user is willing to wait from those for which she is not. I hypothesize that I can model this *user patience* and that this model depends upon just two things:

1. The importance of the file
2. The expected fetch time

If the system were able to predict the user's willingness to wait with reasonable confidence, it could reduce the frequency with which it requests advice during weak connectivity, thereby reducing the burden of translucent caching.

I call the maximum time that a user is willing to wait for a particular file the user's *patience threshold* (τ) for that file. This threshold depends critically upon how important the user perceives that file to be. For a very important file, the user is probably willing to wait many minutes. Because user perception of importance is the notion captured by the hoard priority, P, of an object, I posit that τ should be a function of P. At present, I am not aware of any data that could be the scientific basis for establishing the form of this relationship. Hence, I use a function based solely on intuition, but have performed some analysis to determine appropriate parameter values and have also structured the implementation to make it easy to substitute better functions as they are discovered.

6.3.1 Proposed Model

I conjecture that patience is similar to other human processes, such as vision, that exhibit logarithmic sensitivity [9]. This suggests a relationship of the form

$$\tau = \alpha + \beta e^{\gamma P}$$

where β and γ are scaling parameters and α represents a lower bound on patience. I chose parameter settings based on their ability to yield plausible patience values for files commonly found in the hoard profiles of Coda users. I use the values α=6, β=1, and γ=0.01, based upon the analysis presented in the following section.

Figure 6.2 illustrates the resulting model of user patience. Rather than expressing τ in terms of seconds, I have converted it into the size of the largest file that can be fetched in that time at a given bandwidth. For example, 60 seconds at a bandwidth of 64 Kb/s yields a maximum file size of 480 KB. Each curve in this figure shows τ as a function of P for a given bandwidth. In the region below this curve, cache misses are transparently handled and pre-approval is granted automatically during hoard walks.

This curve predicts that users would be willing to wait indefinitely for sufficiently large files. Clearly, this is not true. HCI researchers have found that delays beyond approximately 15 seconds are disruptive [48]. For this reason, any delay greater than 15 seconds is automatically presented to the user. Thus, the function employed by Venus is

$$\tau = \min(15, \alpha + \beta e^{\gamma P}) .$$

The user patience model is the source of adaptivity in cache miss handling. It maintains usability at all bandwidths by balancing two factors that intrude upon transparency. At very low bandwidths, the delays in fetching large files annoy users more than the need for interaction. As bandwidth rises, delays shrink and the interaction becomes more annoying. To preserve usability, the model must handle more cases transparently. In the limit, at strong connectivity, cache misses are fully transparent.

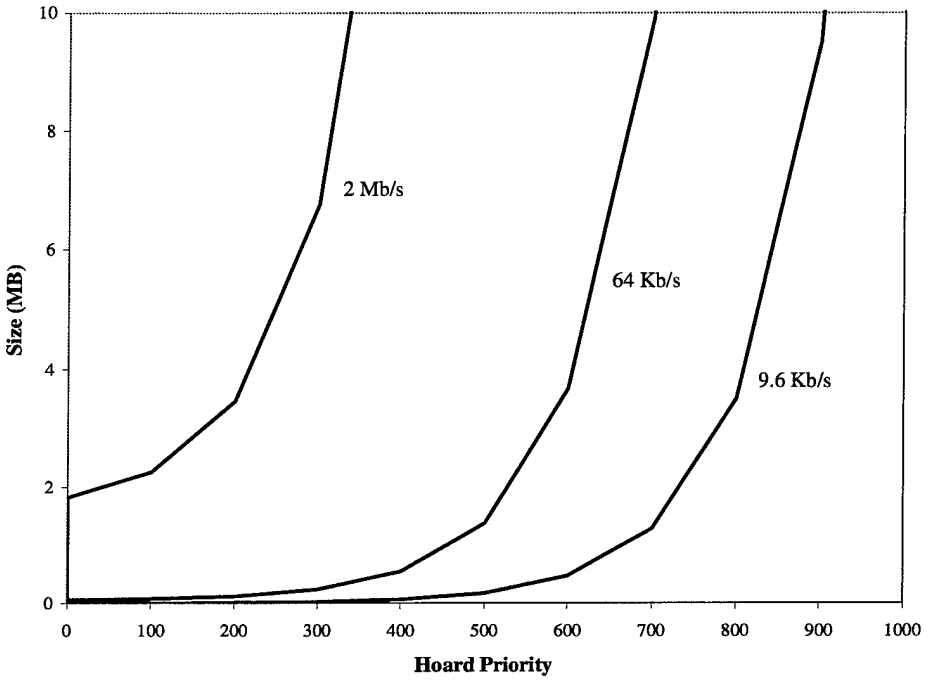


Figure 6.2: User Patience Model

This figure shows the user patience model. The X-axis shows the hoard priority of an object. The Y-axis shows the size of the object. Parameter values for this figure were $\alpha = 6$, $\beta = 1$, $\gamma = 0.01$. Each curve represents a different network bandwidth. The area below the curve represents those objects that will be fetched automatically; the area above the curve represents those objects that require user intervention before being fetched. For example, at 9.6 Kb/s, most high priority objects are fetched regardless of size, but only very small objects with low priority will be fetched automatically. At 2 Mb/s, however, even a 1-MB file with a priority of 0 will be fetched automatically.

6.3.2 Exploring the Parameters

The accuracy of the patience threshold depends upon its parameters (α, β, γ) and upon its argument (P). I begin this section with a brief discussion of the argument, and then examine each of the model's parameters.

6.3.2.1 Priority (P)

The argument to the function is the priority of the object to be fetched. The user indirectly assigns this priority when requesting that a task be hoarded using the window shown in Figure 4.15. These priorities range from 1 to N (the number of hoarded tasks), where 1 is the highest priority task. From these user-assigned priorities, the system must assign hoard priorities meaningful to Venus (in the range from 1 to 1000).

The obvious strategy for performing this translation would be to space out the hoard priorities evenly. This strategy has two important features: its ease of implementation and its ability to handle up to 1000 hoarded tasks. An unfortunate consequence, however, is that, as N increases, tasks become difficult for Venus to distinguish because the hoard priority of an object is but one component of its overall priority (see Section 2.2.2).

The strategy I used, instead, spreads high priority tasks at the cost of compressing low priority tasks. By spreading high priority tasks, I improve the odds that Venus can ultimately distinguish between files belonging to different high priority tasks, and thus allow Venus to make more intelligent caching decisions. This ability comes with a cost, however. By compressing low priority tasks, I decrease the odds that Venus can distinguish between files belonging to different low priority tasks.

In effect, this strategy increases Venus' ability to distinguish high priority tasks at the cost of decreasing Venus' ability to distinguish low priority tasks. This trade-off is important for two reasons. First, this trade-off assumes that the difference in importance between tasks prioritized 1st and 2nd by the user is generally higher than the difference in importance between tasks prioritized 31st and 32nd. Second, it increases the probability that all files belonging to a task with priority A are cached in preference to any files belonging to the task with priority A+1, for values of A near 1.[18] Because of Venus' prioritized cache management algorithm, objects with lower hoard priorities can, under certain circumstances, be cached in preference to objects with higher priorities.[19] Spreading the hoard priorities of the most important tasks makes this situation less likely to occur with these tasks.

Although this strategy reduces the number of unique tasks from 1000 to 88, this is still many more than a user is likely to need. The highest priority tasks are separated by a distance of 200; the lowest priority tasks are separated by a distance of only 1. To spread

[18] Because of the trade-off, this probability decreases as A increases.

[19] These circumstances require that the user has accessed the lower priority objects more recently. For this to happen, the reference priority component of the overall priority overwhelm the hoard priority component. Please refer to Section 2.2.2 for further details regarding Coda's prioritized cache management.

the assigned priorities, I employ the Fibonacci sequence. The number of elements spread by N points is Fib(i), as N decreases from 200 and i increases from 1. Thus, there is one gap of 200 priority units, one gap of 100 priority units, two gaps of 50 units, etc. If the user hoards more than 88 tasks, the additional tasks are all assigned the same low hoard priority and are, thus, indistinguishable from one another from the perspective of Venus. This distribution is summarized in Figure 6.3.

The use of the Fibonacci sequence is not critical to this strategy. The domain of the function (the user's priorities) must be the set of integers. The range of the function (Venus' hoard priorities) must be the set of integers from 0 to 1000. The mapping of user priorities to hoard priorities should be a nonincreasing, nonlinear function. The nonlinearity should spread lower values of the domain further apart than higher values of the domain, up to a point. Beyond that point, all values of the domain can map onto a single (low) value of the range. Any function with these characteristics should suffice. I use the Fibonacci sequence described above for its simplicity. Numerous alternatives, including exponential, inverse quadratic, and Zipf distributions, could work equally well.

6.3.2.2 Theoretical Lower Bound (α)

The first parameter I consider, α, represents a lower bound on patience. Even for an unimportant object, the user would prefer to tolerate a delay of up to α seconds rather than dealing with a cache miss; thus, this parameter specifies the amount of time the user is willing to wait for even the least important file. From a different perspective, α offers a way to prevent the interface from asking the user a question (e.g., the weak-miss query of Section 4.6.1.2) in situations where the system can fetch the object faster than the user can respond to the question. Thus, a candidate value for α is the minimum time required for a typical user to respond to the weak-miss query.

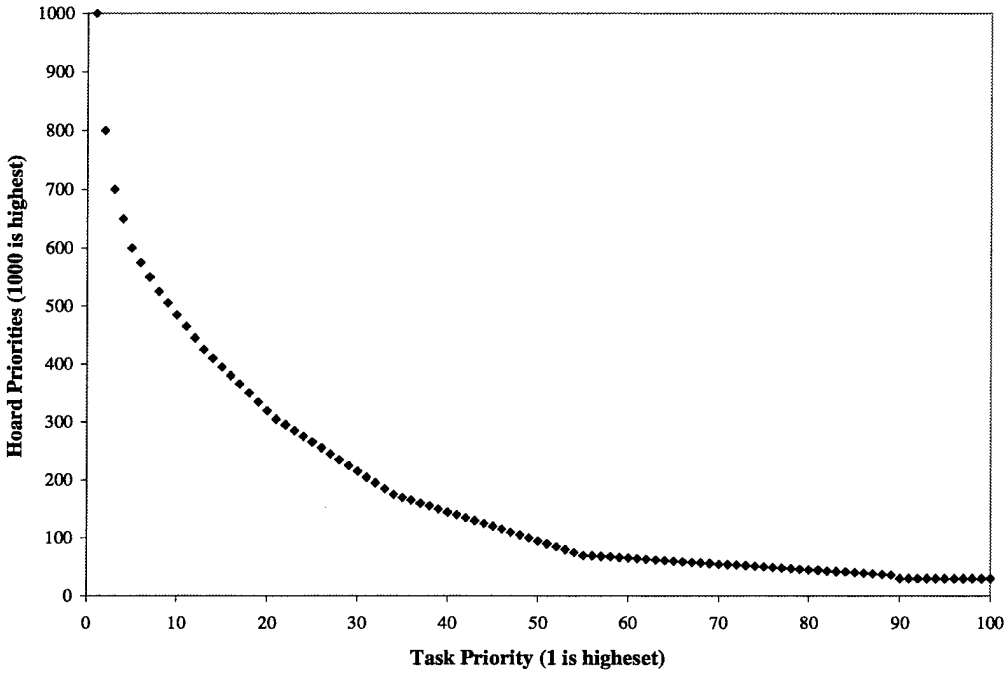
To determine the time required to respond to the weak-miss query, I use a Keystroke-Level Model (KLM), a form of GOMS analysis [7]. In GOMS modeling, one breaks down a task into a series of goals, operators, methods, and selection rules. For my models, I assume, among other things (see Figure 6.4), that the user is an expert. This assumption implies that the user has a method for responding to a weak-miss query, that the method is reasonably efficient, and that I need only define the actions specified in that method. The specific method used in my model is the one requiring the fewest operators. Once the operators necessary to complete the method have been identified, the time required to execute that method can be estimated by calculating the sum of the time required to apply each individual operator, as summarized in Figure 6.5.

For the model of the weak-miss query, I consider two pointing devices and two configuration options. The two pointing devices I consider are a mouse and a keyboard pointing device[20] common in laptop computers. The keyboard pointing device is of a small rubber cylinder (similar in size to the eraser on a pencil) that is situated between the

[20] IBM calls their device the TrackPoint.

Number of gaps	Units Separating Tasks
1	200
1	100
2	50
3	25
5	20
8	15
13	10
21	5
34	1
Total = 88	

(a)



(b)

Figure 6.3: Distribution of Task Priorities

This figure shows the distribution of task priorities. The table shown in view (a) defines how hoard priorities are assigned to hoarded tasks. The highest priority tasks are assigned the hoard priority 1000. The next most important task receives a priority of 800, or 200 units below the previous priority. The next task receives a priority of 700, or 100 units below the previous priority. The next two tasks are separated by 50 units and receive the priorities 650 and 600, and so on. More important tasks have more distinct hoard priorities. The chart shown in view (b) illustrates the shape of this distribution.

Assumptions
Expert user: knows how to use indicator lights knows where to look for *Advice Information* window to appear knows how to select event in *Advice Information* window
Weak-miss query is the only pending advice.
Weak Miss Event settings: no beeping no flashing
User is a touch typist.
User is an expert with the pointing device.
User must mentally prepare (**M**) to take action at beginning of method.

Figure 6.4: Assumptions Made in Building KLM Models.
This table shows the assumptions that I made in building the Keystroke-Level Models.

KLM Operator	Description	Time (ms)
H[mouse]	Home hand on device.	400
H[point]	Home finger on device.	75
B[button]	Press button (up or down).	100
M	Mentally prepare.	1200
P[location]	Point with mouse to target on display.	1100
R[type, machine]	System response time.	$SRT(t,m)$

Figure 6.5: Operators of the Keystroke-Level Model
The **H**, **M**, **P**, and **R** operators and their time estimates are defined in Chapter 8 of Card, Moran, and Newell [7], as modified by Kieras [21]. The **B** operator is defined by Kieras. Homing to a keyboard pointing device is assumed to be equivalent to moving a finger from one key to a neighboring key and is derived from Equation 2 of [16]. The time estimates for the **R** operator were measured from the interface mockup and appear in Figure 6.6, parameterized by the type of response time and the machine on which it was measured.

Type	Description	System Response Time (s) Laptop	Desktop
A-cold	*Advice* Indicator ⇒	1.21	1.14
A-warm	*Advice Information* Window	0.79	0.68
B-cold	*Advice Information* Window ⇒	0.19	0.24
B-warm	*Weak-miss Query* Window	0.16	0.17

Figure 6.6: Measured System Response Times

The time measurements were obtained via the `time` facility provided by Tcl/Tk [56]. I performed 7 trials under each condition on each machine. The measurement reported represents the mean of 5 trials (high and low were thrown out). The cold measurements represent the time necessary to bring up the window for the first time. The warm measurements represent the time necessary to bring up the window for the second time. The measurements for response time B required that I modify the code slightly to return immediately after displaying the query window to the user. (Otherwise, the measurement would have included the delay caused by the user answering the question.) The measurements for the Desktop were taken on a DECstation 5000/200 running Mach 2.5. The measurements for the Laptop were taken on an IBM Thinkpad 701c (75 MHz i486-based machine) also running Mach 2.5. The interface was otherwise idle. The machine was running multi-user, on the network, and with the standard complement of system daemons. A single user was logged in and running X windows; that user was inactive except for the actions necessary for obtaining the measurements.

'G', 'H', and 'B' keys of a QWERTY keyboard. Two buttons, located below the space bar, serve the same purpose as the left and right mouse buttons. The two configuration options I consider are whether or not the user has configured the weak-miss query to pop up a window on the monitor. The models for the four cases appear in Figures 6.7–6.10.

Consider Method #1, shown in Figure 6.7. For this method, I assume the user is using a mouse and has configured the interface not to pop up the weak-miss query. This method contains a total of twelve steps. It is invoked when the user notices that the interface has turned the *Advice* indicator red. The user decides to investigate and begins by reaching for the mouse (I assume the user was interrupted from a typing activity so that her hands are initially hovered over the keyboard), pointing to the *Advice* indicator, and double-clicking the left button. The two mental operators normally required to point the mouse and to double-click a button have been anticipated by the home-to-mouse operator and have thus been eliminated [7].

Once the system has displayed the *Advice Information* window, the user points to the *Weak Miss* entry and double-clicks the left button. The mental operator required to point at the entry is eliminated because it could be anticipated and is therefore assumed to have occurred concurrently with the system response time. Further, the mental operator required to double-click the mouse was anticipated by the previous point operator.

Once the system displays the request, the user must make a decision[21], point to her choice, and then click the left button. The first mental operator cannot be eliminated nor can it overlap with the system response time, because the information necessary to make the decision is not available until the window has been displayed. The mental operator preceding the button click, however, is fully anticipated by the previous point operation.

Once the user has completed the request, she returns the cursor to her work area and moves her hands back to the keyboard. The mental operator preceding this pointing action is not eliminated because I assume she verifies that she has, indeed, completed all necessary advice activity. Note that the model does not consider the time required to close the *Advice Information* window because such an action is not strictly necessary. If the model were to include this action, the time estimate would increase by 1.3 seconds (**MPBB** inserted prior to the "Point to Previous Window" action with the mental operator in the following action eliminated).

Because the other three models (shown in Figures 6.8-6.10) are similar, I will summarize their differences from this first model, but I will not discuss each of them in detail. The only difference between the two pointing device models is the amount of time required to home to the pointing device. In the case of a mouse, the user must move a hand from the keyboard to the mouse. In the case of the keyboard-pointing device, the user must move

User Action	KLM Operator
Notice Indicator Light	**M**
Reach for Mouse	**MH**[mouse]
Point to Indicator Lights	**MP**[indicator]
Double-click Left Button	**MBBBB**[left]
System Response Time A	**R**[A-cold, desktop]
Point to *Advice Information* Window	**MP**[advice]
Double-click Left Button	**MBBBB**[left]
System Response Time B	**R**[B-cold, desktop]
Point to Command	**MP**[command]
Click Left Button	**MBB**[left]
Point to Previous Work Window	**MP**[work]
Reach for Keyboard	**MH**[keyboard]
$T_{execute} = 3 \cdot t_M + 10 \cdot t_B + 2 \cdot t_{H_M} + 4 \cdot t_P + t_{R_1} + t_{R_2} = 11.18 \text{ seconds}$	

Figure 6.7: Method #1—Desktop without Popup Query

Operators shown in light gray were eliminated by Rule #1 of the Keystroke-Level Model. The one shown in dark gray occurred concurrently with the preceding system response time. (Although this **M** operator should theoretically take longer than the system response time, the system response time takes longer in practice.)

[21] I do not include the time necessary for the user to make this decision in my models because I assume that it is identical in each of the four cases.

User Action	KLM Operator
Notice Indicator Light	**M**
Reach for Mouse	**MH**[point]
Point to Indicator Lights	**MP**[indicator]
Double-click Left Button	**MBBBB**[left]
System Response Time A	**R**[A-cold, laptop]
Point to *Advice Information* Window	**MP**[advice]
Double-click Left Button	**MBBBB**[left]
System Response Time B	**R**[B-cold, laptop]
Point to Command	**MP**[command]
Click Left Button	**MBB**[left]
Point to Previous Work Window	**MP**[work]
Reach for Keyboard	**MH**[keyboard]
$T_{execute} = 3 \cdot t_M + 10 \cdot t_B + 2 \cdot t_{H_P} + 4 \cdot t_P + t_{R_1} + t_{R_2} = 10.46 \text{ seconds}$	

Figure 6.8: Method #2—Laptop without Popup Query

User Action	KLM Operator
Task Interruption	**M**
Reach for Mouse	**MH**[mouse]
Point to Command	**MP**[command]
Click Left Button	**MBB**[left]
Point to Previous Work Window	**MP**[work]
Reach for Keyboard	**MH**[keyboard]
$T_{execute} = 3 \cdot t_M + 2 \cdot t_B + 2 \cdot t_{H_M} + 2 \cdot t_P = 6.8 \text{ seconds}$	

Figure 6.9: Method #3—Desktop with Popup Query

User Action	KLM Operator
Task Interruption	**M**
Reach for Mouse	**MH**[point]
Point to Command	**MP**[command]
Click Left Button	**MBB**[left]
Point to Previous Work Window	**MP**[work]
Reach for Keyboard	**MH**[keyboard]
$T_{execute} = 3 \cdot t_M + 2 \cdot t_{H_P} + 2 \cdot t_B + 2 \cdot t_P = 6.08 \text{ seconds}$	

Figure 6.10: Method #4—Laptop with Popup Query

her index finger from a keyboard key to the pointing device.[22] Homing a finger to the keyboard-pointing device is assumed to be equivalent to moving a finger from one key to a nearby one. The total time difference between these two conditions is approximately 0.7 seconds.

Note, however, that the assumption that the only difference between these pointing devices is their homing time is a simplification. Researchers comparing the performance of people using these devices to performance using mice have reported mixed results. An early study showed that, for pure pointing tasks, the pointing stick is 20-25% slower than a mouse [4, 43]. Research has also shown that, for intermixed pointing and keyboard tasks (such as the one under discussion), the keyboard pointing stick is actually faster than a mouse [43]. Unpublished studies show that, for more recent keyboard-pointing devices, experienced users come within 5% of mouse performance. My assumption that the main difference between the keyboard-pointing device and the mouse is in its homing time reflects the fact that the task in question is a mixed pointing and keyboard task, and that the user is experienced with the device in question.

The differences between the two configuration options are more substantial. In the case of the no-popup option, the user must first open the *Advice Information* window and then open the advice request window. Neither of these activities is required of the popup option since the system automatically opens this window for the user. The models comparing these two options predict that this change in event configuration makes a difference of more than 4 seconds in the time required to complete the questionnaire.

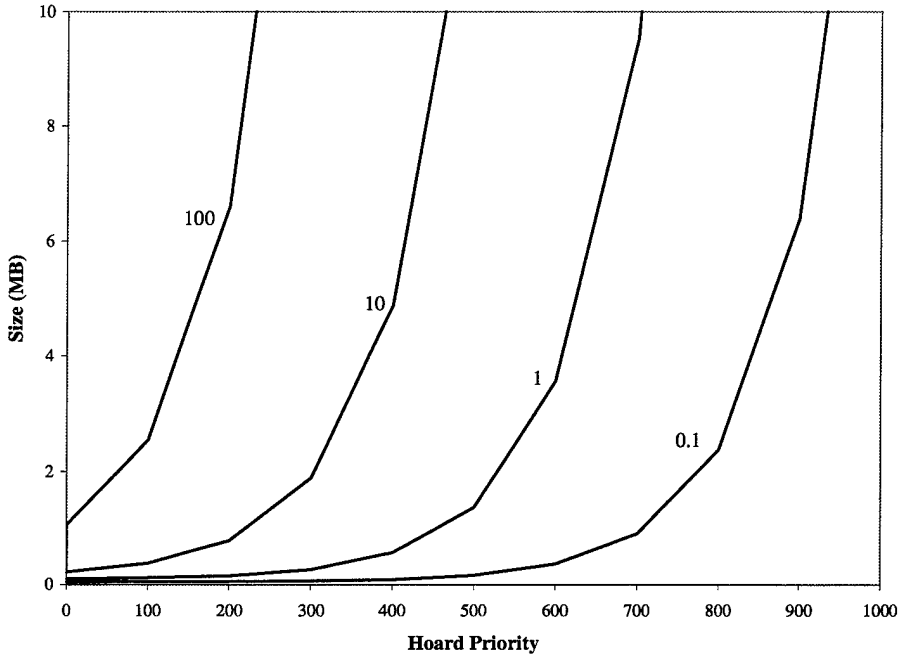
The time required to answer the weak-miss query, depending upon pointing device and configuration options, ranges from 6 to 11 seconds. The models show that the time required changes only minimally based upon the choice of pointing device, but quite substantially based upon the configuration of events. Because of this disparity, the default value for α is 6 seconds, but the CodaConsole increases this value to 11 seconds when the user's event configuration is set to not pop up this query.

6.3.2.3 Sensitivity Analysis of β and γ

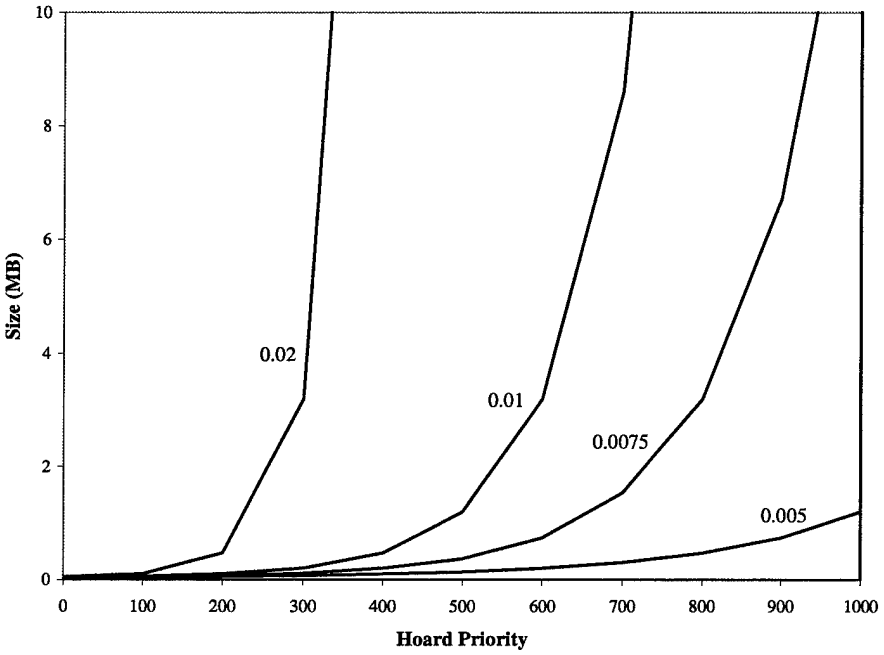
The other two parameters to the user patience model, β and γ, control the shape of the curve. Figure 6.11 shows how changes to these parameters affect the model. As is evident in these figures, decreasing the value of β from its default of 1 or decreasing the value of γ from its default of 0.01 flattens the curve. Similarly, increasing their values sharpens the curve. The default values for each of these parameters were chosen so that the model would yield plausible patience values for files and priorities commonly found in the hoard profiles of Coda users. To determine better estimates for those parameters, one would need to verify the model and collect data regarding the patience of users in practice (see Section 9.2.1).

---

[22] The user must also move her thumb to the primary button (e.g. the equivalent of the left mouse button). I assume this operation overlaps with the home to the pointing device and the subsequent movement of the cursor.



(a) Sensitivity to β



(b) Sensitivity to γ

Figure 6.11: Sensitivity to Changes in β and γ

These figures show the effect of changing the value of the β and γ parameters to the user patience model. The value of β in view (a) varies from 0.1 to 100; the values of α and γ remain constant at 6 and 0.01, respectively. The value of γ in view (b) varies from 0.005 to 0.02; the values of α and β remain constant at 6 and 1, respectively.

6.4 Summary

In this chapter, I have presented three techniques intended to reduce the burdens of translucent caching. The first technique, hoarding programs automatically, reduces the amount of effort users must expend building and maintaining hoard profiles for the programs they use. The second, advising about hoard contents, reduces the likelihood that users will experience failures of availability by identifying file system objects that the user may want to consider adding to a task definition. This technique also identifies objects that the user may want to consider removing from task definitions, thereby reducing the time and memory users must invest in hoarding. Finally, the third technique, modelling user patience, reduces the number of advice requests initiated by Venus. Reducing this number directly reduces the time and cognitive requirements placed on the user.

7 Evaluating the Interface

The previous chapters discussed the design and implementation of the CodaConsole interface as well as the techniques employed by it. This chapter presents an evaluation of the usability of that interface using a technique developed by the human-computer interaction (HCI) community.

For the benefit of those readers not familiar with HCI evaluation techniques, I begin this chapter by briefly summarizing the range of usability evaluation techniques available, focusing on the technique used in my evaluation. For further information on these and similar techniques, Nielsen [37] provides an excellent overview.

I then present the usability test, detailing the goals, method, results, analysis, and findings of the study. The test considers two perspectives, that of expert Coda users and that of novice Coda users. Briefly, the goal of the test was to determine the ease with which users were able to master the interface, their ability to understand the feedback provided by the interface, and their ability to operate the interface.

After presenting the results of the usability test, I then reflect on the usability test itself, focusing on the lessons I learned about the process. I conclude the chapter with a summary of the main results.

7.1 HCI Evaluation Techniques

The HCI community has developed numerous evaluation methods. These range from simple (and quick) techniques like heuristic evaluation [36] and cognitive walkthroughs [57] to ones that are more involved like usability testing [10] and field trials. *Heuristic evaluation* aims to find usability problems by having a small set of evaluators judge the interface's compliance with a set of accepted usability principles. *Cognitive walkthroughs* judge the ease of learning, particularly exploratory learning, of an interface. Both of these methods are *discount* usability engineering methods[23] because they can be performed in little time and at low cost [37]. *Usability testing* is an evaluation method whose goal is to discover usability problems by observing users interacting with the software. It requires a working interface (though that interface can be a mock-up) as well as more time and money, but offers more realistic and detailed data. *Field trials* evaluate software in "the real world". By that, I mean the users are the actual people who use the software every

[23] Nielsen categorizes heuristic evaluation as a discount usability method. Though he does not classify cognitive walkthroughs, this method also fits his definition.

day and the tasks used in the evaluation are the actual tasks those people do during the period of observation. Field trials require significant amounts of time and money as well as a production-quality system, but offer the most realistic data.

Usability testing is, in some sense, a compromise between the discount usability methods and field trials. It offers significant advantages over discount methods, but at significantly lower cost than a field trial. Furthermore, it can be performed relatively early in the life cycle of the software. For these reasons, I chose this method for my usability evaluation. In the rest of this section, I discuss the purpose of usability tests and describe two important issues to consider while designing such a test.

7.1.1 Purpose

The purpose of a usability test is to discover *the most serious usability problems* in a piece of software by observing users interacting with it. The goal is neither to prove that the software is usable nor to obtain statistically significant results. The goal is to discover usability problems so that they can be addressed and corrected. Thus, qualitative data, such as comments made by users as they work with an interface, are often as enlightening as quantitative data, such as the time it took them to complete a task. Data sources, then, include timings, responses to questions, counts of the number of actions taken and even comments made about the interface.

7.1.2 Selecting Participants

In order for the results of a usability test to be meaningful, the participants need to be representative of the people who are expected to use the software. Figure 7.1, often used by Randy Pausch [40], illustrates why this is important. The pie chart on the left shows that the percentage of all computer users who are computer scientists is extremely small[24]. The fundamental problem is that the intuitions of computer scientists do not necessarily apply to the other 99.99% of computer users; in fact, those intuitions hinder their ability to make systems usable to non–computer scientists. If we were to assume other computer scientists were representative of all users, the likelihood of serious usability problems would be high, even though we had performed usability testing.

By extension, this same argument also applies to developers of a system. Because developers know more about the interface's domain than computer scientists in other areas, their intuitions do not necessarily apply to other computer scientists. They simply have too much domain knowledge. Thus, recruiting participants who are representative of the expected user population is a crucial part of usability testing.

---

[24] Not drawn to scale because if it were, "our" slice of the pie would not be visible.

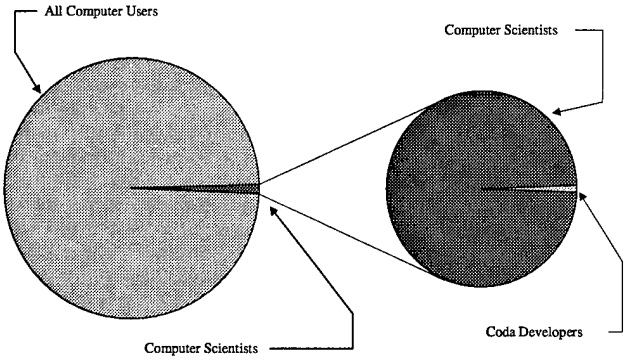


Figure 7.1: Computer Scientists are Special People
This figure illustrates that computer scientists represent a very small percentage of all computer users. It further illustrates that Coda developers represent a very small percentage of computer scientists. Coda developers have background knowledge not available to other computer scientists. Likewise, computer scientists have background knowledge that is not available to other computer users.

7.1.3 Selecting Tasks

A similar argument shows the importance of presenting representative tasks during a usability test. The critical observation here is that usability testing can only discover problems in the areas of the interface under study, though problems may generalize to other areas of the interface that were not studied. If the tasks presented to participants are not representative of actual usage patterns, serious usability problems may go undetected until the interface is released for general use.

7.2 Usability Test: Goals

The first objective of my usability test was to identify areas of the interface and tutorial that were particularly problematic. My goal was to create a list of usability problems, categorized by their severity [10]. The method with which I decided to identify these problems was through observations of people using the interface, and analysis of the qualitative and quantitative data collected during that observation.

The second objective of my test was to gauge whether or not the interface was meeting the learning and performance goals I had set. In particular, my usability test focused on three major issues: the ease of mastering the interface, users' ability to understand the feedback provided, and their ability to operate the interface.

Ease of mastering the interface:

- Can users learn to use the interface in approximately one hour?

Ability to understand feedback:

- Can users understand problems indicated by the interface?
- Can they determine what actions might resolve those problems, if any?

Ability to operate:

- Are users able to define and hoard tasks?
- Are users able to provide advice to the system?
- Are users able to configure individual events?

In answering these questions, my goal was to obtain an indication of how well people with backgrounds similar to those of Coda users would be able to perform activities similar to those required of Coda users. Providing conclusive answers to these questions would obviously require a study of actual Coda users interacting with the production-quality interface and is beyond the scope of this thesis.

7.3 Usability Test: Method

To answer these questions, I performed a usability test[25] in which I observed a total of 14 people interacting with the interface. The first five participants were pilot users. The interface, materials, and/or procedures used for these users differ somewhat from those described in this chapter and contained in Appendix B. The remaining nine participants all used the same interface, materials and procedures. In this section, I discuss the demographics of the participants, the procedure used during the test, and the measurements I made.

7.3.1 Participants

Because my goal was not to release Coda and this interface to the general public, but rather the more modest goal of making Coda more easily accessible to other computer scientists, my user population was other computer scientists. The users for my test were recruited from among computer science graduate students at Carnegie Mellon. Because, at the time, Coda ran only on Unix-based operating systems, all participants were required to have substantial Unix experience and, in fact, all used Unix systems on a daily basis. In addition, I screened users for their experience with the Andrew File System. All participants were required to have substantial experience with AFS, using it on a daily basis. All novices and all but one expert stored either their e-mail directory or their home directory in AFS. Because AFS is well entrenched in the CMU community, this was an

[25] This test was reviewed and approved by the Carnegie Mellon University Provost's office in accordance with the requirements of Public Law 99-158 as implemented by Part 46 of Title 45 of the Code of Federal Regulations and General Assurance No. M-1462-XM.

easy requirement to meet. I did not screen users based upon their HCI experience.

I initially defined a *novice* user as a person who knew nothing about Coda prior to participating in the test. However, due to the difficulty of finding qualified users that met this stringent requirement[26], I was forced to relax it somewhat. The novice users knew very little about Coda. Most of them knew that it was a file system, and that it was related in some way to AFS. None of them had any direct experience with Coda. I defined an *expert* user as a person who had substantial experience using Coda. All of the expert users had Coda laptops and had operated disconnected for substantial periods of time over a period of at least a year.

Although I posted a request for participants for the test on both an electronic bulletin board and on a department-wide, electronic chat system, most participants heard about the test through word of mouth. I recruited a total of 14 users to participate in the test, offering them Coda shirts for doing so. All but two of those users were computer science graduate students. Of the two who were not graduate students, one (C1) was a faculty member and one (S1) was a nontechnical member of the support staff with some undergraduate-level computer science background. Neither of these users could be considered either experts or novices. One pilot user (P5) and the three expert users (E1, E2, and E3) had used Coda. No other user had done so. Three pilot novices (P2, P3, and P4) had some HCI experience. Two novices (N1 and N2) and the faculty member (C1) had extensive HCI experience. No other users had HCI experience. One user (T1) was originally considered a novice, but his data was eventually thrown out because it became clear, during the Tutorial, that he knew more about Coda than I had originally thought.[27]

7.3.2 Treatment

I began each session of the experiment by asking the participant to sign a consent form. Next, the participant filled out a brief background questionnaire, providing some basic demographic information. Once that was completed, the experimenter demonstrated a verbal protocol[28] using a computer game (Klondike). The participant then played Othello briefly to practice thinking aloud. During this practice verbal protocol, the experimenter positively reinforced those comments (both positive and negative) made by the user that were specific in nature, and provided concrete examples of positive and/or negative feedback if none were forthcoming. After a few minutes of practice, the participant began a detailed, interactive tutorial of the CodaConsole interface. The content of the Tutorial is shown in Screens 1-62 of Appendix B. This Tutorial was expected to require approximately one hour to complete. After a break, the participants performed a number

---

[26] At least one Coda paper is now covered in a class required of all computer science graduate students.

[27] The specific comments that led me to this conclusion are documented in Appendix C.

[28] A *verbal protocol* is a technique used to understand what users are thinking and where users run into problems as they work with an interface. Users are simply asked to think aloud as they work. Although this is awkward for users at first, it has been shown to be extremely effective once users become accustomed to talking as they work [11]. One participant commented to me that he continued talking aloud for awhile after completing the test.

of exercises ranging from identifying problems to preparing for a disconnected session. These Exercises are contained in Screens 63-83 of Appendix B. The Exercises were grouped into three phases. During Phase One (Screens 65-74), the participant answered basic questions about the state of the system. During Phase Two (Screens 75-80), he defined and prioritized three tasks. During Phase Three (Screens 81-83), he reacted to problems indicated by the interface. The Exercises were also expected to require approximately one hour to complete. After completing the Exercises, the participant answered a brief survey evaluating the interface. Finally, the experimenter debriefed the participant, allowing the participant the opportunity to ask questions. The materials used in this test can be found in Appendix B. Both the Tutorial and Exercise segments of the test were videotaped, and the audio from all three segments was also recorded.

7.3.3 Measurements

The first measurement I collected was the time (in seconds) required to teach users about the details of the indicator light interface. I measured these times for each screen of the on-line Tutorial. In addition, I measured some higher-level times such as the time the user spent learning about each indicator as well as the time the user spent on the entire Tutorial. Note that this is not quite the same as the sum of the times of the specific screens because those times do not account for the time necessary for the Tutorial to trigger events or display the individual screens.

The second measurement I collected was the time (in seconds) required for the user to complete each exercise. I measured the time for each complete exercise, not at the granularity of individual questions within an exercise. As with the Tutorial times above, I also measured some higher-level times such as the time needed for each of the three phases as well as the time spent on the entire Exercise session. Again, because of the time needed to trigger events or display individual screens, the summary times may not equal the sum of the individual times.

The third measurement captured the user's ability to use the interface to answer questions or perform some action. In addition, it measured the user's ability to describe a problem that had been indicated through the interface, as well as the steps necessary to resolve it, if appropriate. For this measure, I collected the user's response to each exercise. A response could be a written answer to a question or a change to one of the configuration files. I then scored the user's responses according to the key contained in Appendix B.

The final measurement captured the user's efficiency in answering the questions or performing the required actions. For this measure, I counted the number of steps the user took to complete each exercise. A step corresponds to double-clicking an indicator light, clicking a button, or typing a pathname, as well as similar activities. The baseline measurement was that of the author performing the same actions. The specific actions performed during the baseline measurement are contained as the Par Score of Appendix B. It is important to note that each par score is likely to represent a lower bound on the number of steps necessary to answer the question because the author was not only an expert using the interface, but also had foreknowledge of the expected answers to the

exercises. Although an expert without foreknowledge of the exercises would have provided a better standard of comparison, such an expert did not exist.

All but the first participant were videotaped during their sessions. The first participant, P1, informally piloted the Tutorial and Exercises. No video or audio recordings were made because the experiment did not occur in the User Studies Lab. From the transcripts of the recorded sessions, I identified comments made by each user. These comments are recorded in Appendix E in two forms: a numerical listing by comment identifier number and a table summarizing which users made each comment during the three segments (Tutorial, Exercises, Debriefing) of the test.

7.4 Usability Test: Results

Overall, the results of the usability test were highly positive. Novice Coda users were able to use the system almost as well as expert Coda users. They learned about Coda in approximately one hour and they were able to successfully complete exercises typical of those required of users of the new interface. In this section, I revisit the original objectives of the test, examining whether or not users were able to master, understand, and operate the interface.

7.4.1 Learnability

The first set of questions relates to the ease of learning how to use the interface. The first objective of the test was to determine whether or not users could learn to use Coda and the interface in approximately one hour. Indeed, they could. The expert and novice participants in the test each took approximately one hour. The fastest user, N4, took just under 42 minutes; the slowest, N2, took just under 63 minutes.

Another goal of the usability test was to identify areas of the Tutorial that users found difficult. All but two screens that users found difficult were directly related to one or more usability problems in the interface. Improving the Tutorial may have helped users, but would not have fundamentally solved the problem. The other two screens (those numbered 23 and 28) were probably falsely identified as difficult by limitations in my analysis. The analysis presented in Section 7.5.1 below examines this issue in detail.

In order to confirm this result, I examined the qualitative data collected as part of the questionnaire administered after completion of the Exercises (see Appendix B). From these data, I learned that users felt the Tutorial was basically complete. On a scale of one to five (where 5 is "very complete"), the Tutorial received a score of 3.8. Four of the six users commented that reintegration was not covered, though it was used during the Exercises. This omission was a deliberate attempt to see how users would react to "learning on the fly". Based upon their reactions, it appears that they did not appreciate the experiment. Users thought the Tutorial was easy to understand (4.2 out of 5). Users unanimously felt that the Tutorial allowed them to grasp the scope of the interface.

Finally, five of six users felt it was presented at an appropriate level of detail. The sixth user, an expert, felt there was too much detail, specifically too much Coda background. Because the Tutorial was written for a novice audience, this is not surprising.

Based upon the quantitative and qualitative data I collected, the Tutorial achieved its objects of teaching users about Coda and the interface in about an hour. Further, no serious difficulties directly attributable to the Tutorial were uncovered.

7.4.2 Understandability

The second set of questions relates to the user's ability to understand the information presented by the interface. Eighteen of the twenty four exercises address this issue. These exercises include most of Phase One (those asking users to describe the current network connectivity, to identify which tasks are currently hoarded, etc.) and all of the exercises in Phase Three (which asked users to explain events notified via the indicator lights). Isolating the users' scores on these exercises from Figure 7.2, I find that users did extremely well. All but one user scored 97% correct or better, and that user scored 90% correct. Furthermore, the data show that, with the exception of just one of these exercises (SpaceStatus), no two users answered the same exercise incorrectly. These users convincingly demonstrated their understanding of the information presented by the interface and of what, if any, actions might resolve those problems.

Further, these users silenced any concerns over whether or not users could understand the feedback provided during event notifications. Phase Three of the Exercises triggered 12 different events. For each event, users were asked to fill out an "incident report", consisting of the urgency of the event, a description of the problem, and a description of any actions that might resolve the problem. The six expert and novice users filled out 71 incident reports describing these events—one user failed to report an event that transitioned from a "bad" state to a "good" state. Of these 71 incident reports, only three contained incorrect information. Users clearly understood the error conditions and what, if any, steps were possible to resolve those problems.

7.4.3 Operability

The third set of questions relates to the user's ability to operate the interface. Eight exercises address this issue. These exercises include three from Phase One (which involved configuring events and providing advice, and which were excluded from the discussion above) and all of Phase Two (which involved defining and prioritizing tasks). Although users scored well on these exercises, their scores declined somewhat (see Figure 7.2). Five of six users scored 90% or better, and the sixth user scored 84%. Furthermore, the data show that, with the exception of two of these exercises (ProvideHoardWalkAdvice and DefineBugsTasks), no two users answered the same exercise incorrectly. Although these scores are very encouraging, users clearly had more difficulty operating the interface than they did understanding the feedback it provided.

Problem Identifier	Pts	Novice Users				Expert Users				Overall	
		1	2	4	Avg	1	2	3	Avg	Avg	StdDev
Describe current network conditions	3	3	3	3	3.0	3	3	3	3.0	3.0	0
Describe "Tasks unavailable" event & configuration	6	5	6	6	5.7	6	6	6	6.0	5.8	0.4
Fix configuration for this event	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Determine token expiration time	1	1	1	1	1.0	1	1	1	1.0	1.0	0
Describe hoarding details	12	12	12	12	12.0	12	12	12	12.0	12.0	0
Provide hoard walk advice	2	2	1	0*	1.0	1	1	1	1.0	1.0	0.6
Describe current space status	3	2	3	3	2.7	1	2	3	2.0	2.3	0.8
Describe "Weakly Connected Cache Miss" event & "Disconnected Cache Miss" event	8	6	8	8	7.3	8	8	8	8.0	7.7	0.8
Fix configuration for above events	4	4	4	4	4.0	4	4	4	4.0	4.0	0
Define "Fixing Bugs" task	9	7	8	7	7.3	9	8	7	8.0	7.7	0.8
Define "Header Files" task	5	5	5	5	5.0	5	5	5	5.0	5.0	0
Define "CV" task	6	5	6	6	5.7	6	6	6	6.0	5.8	0.4
Prioritize newly defined tasks	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Request immediate hoard walk	1	1	1	0*	0.7	1	1	1	1.0	0.8	0.4
Describe "Filling RVM" problem	3	3	3	3	3.0	3	3	3	3.0	3.0	0
Describe "Fixed RVM" problem	2	Ø	2	2	1.3	2	2	2	2.0	1.7	0.8
Describe "Task Unavailable" problem	3	3	3	3	3.0	3	3	3	3.0	3.0	0
Describe "Hoard Walk Begin" event	2	2	2	1	1.7	2	2	2	2.0	1.8	0.4
Describe "Hoard Walk End/Task Available" event	2	2	2	2	2.0	2	2	1	1.7	1.8	0.4
Describe "Operating Weakly Connected" problem	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Weak Miss" event	2	2	1	2	1.7	2	2	2	2.0	1.8	0.4
Describe "Operating Disconnected" problem	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Disconnected Cache Miss" event	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Operating Strongly Connected" problem	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Tokens Expiry Event"	3	3	3	3	3.0	3	3	3	3.0	3.0	0
Describe "Reintegration Pending Tokens" event	3	3	3	3	3.0	3	3	3	3.0	3.0	0
Total Points	92	83	89	86	86.0	89	89	88	88.7	87.3	2.4
Percentage Correct	100	90	97	93	93.3	97	97	96	96.7	95.0	2.9

Figure 7.2: Correctness Ratings

This table shows the score each user received on each of the exercises as well as their total and percent correct. The empty set notation (Ø) indicates the user did not perform this exercise. User N4 explained precisely which files she would have requested to have fetched during the "Provide hoard walk advice" exercises; unfortunately, she failed to actually select those objects to be fetched in the interface. Had she completed the exercise as she described, her score would have been a 1. Because of her failure to complete this exercise, she was also unable to complete the "Request immediate hoard walk" exercise—though she did the most reasonable thing given the situation.

A different metric, which also gauges the user's ability to operate the interface, compares the number of actions users need to perform each exercise with the number needed by an expert. Figure 7.3 shows this data. In general, users need approximately twice as many actions as a user who is experienced with the interface and who has foreknowledge of the expected answers. Users clearly found the hoard walk advice and task definition aspects of the interface more challenging than other aspects of the interface. This trend is not surprising because these aspects of the interface require more problem solving. One interesting observation is that expert users required 30 steps more than novice users, on average. Largely, this appears to have been the result of two anomalous data points (E1's performance on defining the "Fixing Bugs" task and E3's performance on defining "Header Files" task). This observation should not be taken as an indication that expert users experienced more difficulty than novice users.

The test set out to answer three specific questions related to operability. I wanted to learn whether or not users could define and hoard tasks, provide advice to the system, and configure individual events. Users were largely able to define and hoard three realistic tasks (see Figure 7.2). The most disturbing trend I observed was the high number of users who failed to hoard the `compress`-related programs. Users were able to provide advice to the system, both with respect to weakly connected demand fetches and with respect to hoard walks. Users were also able to configure events, largely without difficulty. The problems users experienced are discussed in detail in the next section.

7.5 Usability Test: Analysis

As the previous section documents, the participants successfully met all of the objectives set forth for the test. As with any user interface, however, there is always room for improvement. One of the important goals of the test was to identify areas of the interface on which to concentrate our future efforts. Because users performed so well, I had to examine the data more closely to identify these areas.

This section presents this analysis in detail. The analysis is based upon the data collected during the course of the test. The raw data is contained in Appendices C and D. The details of the analysis are contained in Appendix E. The results of this analysis, the findings, are contained in Appendix F and summarized in Figure 7.4.

These findings are categorized by their *scope* and *severity* [10]. The scope indicates how widespread a problem is. A problem with *local* scope is limited to a single window whereas a problem with *global* scope applies to multiple windows. A problem with global scope is generally considered more critical than one with local scope. The severity of a problem indicates how critical the problem is. Dumas and Redish define four different severity levels. A problem with severity level 1 prevents the completion of a task. One with severity level 2 causes users significant delay or frustration. One with severity level 3 presents only a minor effect to usability. One with severity level 4 represents a more subtle problem and may point to potential enhancements to the interface.

Problem Identifier	Par	Novice Users 1	2	4	Avg	Expert Users 1	2	3	Avg	Overall Avg	StdDev
Describe current network conditions	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Task Unavailable" event & configuration	3	5	3	6	4.7	6	3	3	4.0	4.3	1.5
Fix configuration for this event	4	3	4	3	3.3	4	4	4	4.0	3.7	0.5
Determine token expiration time	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe hoarding details	4	6	2*	4	4.0	6	4	3	4.3	4.2	1.6
Provide hoard walk advice	10	35*	46	31	37.3	20	36	26	27.3	32.3	9.0
Describe current space status	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Weakly Connected Cache Miss" event & "Disconnected Cache Miss" event	3	7	5	3	5.0	3	3	15	7.0	6.0	4.7
Fix configuration for above events	8	12*	9	8	9.7	9	17	8	11.3	10.5	3.5
Define "Fixing Bugs" task	36	62	47	40	49.7	96	44	58	66.0	57.8	20.5
Define "Header Files" task	16	26	26	25	25.7	47	37	100	61.3	43.5	29.0
Define "CV" task	21	73	46	47	55.3	57	31	37	41.7	48.5	15.0
Prioritize newly defined tasks	14	5*	33*	38	25.3	27	24	37	29.3	27.3	12.2
Request immediate hoard walk	3	3	3	7	4.3	3	5	7	5.0	4.7	2.0
Describe "Filling RVM" problem	2	3	6	6	5.0	4	6	6	5.3	5.2	1.3
Describe "Fixed RVM" problem	0	Ø	0	0	0	0	2	2	1.3	0.8	1.1
Describe "Task Unavailable" problem	3	7	10	5	7.3	7	2	3	4.0	5.7	2.9
Describe "Hoard Walk Begin" event	0	1	0	0	0.3	0	0	1	0.3	0.3	0.5
Describe "Hoard Walk End/Task Available" event	0	3	5	8	5.3	7	4	4	5.0	5.2	1.9
Describe "Operating Weakly Connected" problem	2	2	4	2	2.7	3	1	6	3.3	3.0	1.8
Describe "Weak Miss" event	1	3	3	2	2.7	3	1	2	2.0	2.3	0.8
Describe "Operating Disconnected" problem	1	5	4	3	4.0	0*	0	7	2.3	3.2	2.8
Describe "Disconnected Cache Miss" event	2	2	2	2	2.0	2	4	2	2.7	2.3	0.8
Describe "Operating Strongly Connected" problem	0	2	0	2	1.3	2	1	0	1.0	1.2	1.0
Describe "Tokens Expiry Event"	0	1	3	2	2.0	2	3	2	2.3	2.2	0.8
Describe "Reintegration Pending Tokens" event	4	1	2	10	4.3	4	2	4	3.3	3.8	3.3
Total	143	273	269	260	267	318	240	343	300	284	39
Efficiency (Score/Par)	1.0	1.9	1.9	1.8	1.9	2.2	1.7	2.4	2.1	2.0	0.3

Figure 7.3: Efficiency Ratings

This table shows the number of steps each user took to perform each of these required tasks. The empty set notation (Ø) indicates that the user did not perform the task. Numbers which are underlined represent exercises for which this user was more efficient than the interface expert. Numbers marked with an asterisk are explained in the paragraph below.

User N1 requested significantly less data be fetched during the "Provide Hoard Walk Advice" exercise though he required more steps to provide that advice. User N1 hit certain buttons multiple times in a row (as many as eight). Consecutive events such as this are counted as two actions for the purposes of measuring the user's efficiency. Primarily this affects the two "event configuration" exercises and the three "task definition" exercises. User N1 prioritized tasks as he defined them so his efficiency on the "Prioritize newly defined tasks" exercise is artificially low. User N2 has an artificially high rating on this same task because he went back to reorganize the "Define 'Header Files' Task" while performing this task. User N2 has an artificially low efficiency score for the "Describe hoarding details" exercises because he resized the window and avoided scrolling. User E1 already had the "Network Information" window visible when he performed the "Describe 'Operating Disconnected' problem" exercise.

ID	Description	Scope	Severity	Page
1	Users confused by event notification	Global	1	414
2	Some help windows aren't helpful	Local	1	414
3	Users fail to include needed programs in definition	Local	1	415
4	Event Configuration Tab of Control Panel layout inappropriate to task	Local	1	415
5	Certain characters, if used in task names, can crash the interface	Local	1	415
6	Password visible on the Authentication Information window	Local	1	416
7	Users do not recognize the hoard walk advice request as a request for advice	Local	1	416
8	User does not recognize that the Hoard Walk Advice offers more details	Local	1	416
9	Indicator changes to Normal urgency level before appropriate to do so	Local	1	417
10	Users unsure whether hoard walk is proceeding during request for advice	Local	1	417
11	User prioritized tasks inappropriately	Local	1	417
12	Users want undo facility	Global	2	418
13	Users find the behavior of "Save" and "Commit" confusing	Global	2	418
14	Users are frustrated by the placement of windows on the screen	Global	2	419
15	Interface loses definitions under certain conditions	Global	2	419
16	Users experienced difficulty while prioritizing tasks	Local	2	419
17	Users have difficulty entering data into "Data Definition" window	Local	2	420
18	Interface loses pathnames	Local	2	420
19	Users experience difficulty finding an element in predefined list	Local	2	421
20	User has difficulty unselecting element from the contains list	Local	2	421
21	Submitting a password takes excessively long	Local	2	421
22	User believes colors mean too many different things	Global	3	422
23	Interface offers no feedback to indicate a window is already visible	Global	3	422
24	Windows do not resize gracefully	Global	3	422
25	Users confused when help windows grab and maintain focus	Global	3	422
26	Users confused that changing name and saving creates a new definition	Global	3	423
27	Users dislike pathnames entry	Global	3	423
28	Hoard Walk and Advice indicators confused	Global	3	423
29	Hoard Walk and Task indicators confused	Global	3	423
30	User experiences window overload	Global	3	424
31	Interface mixes upper- and lower-case letters	Global	3	424
32	User confused by terminology	Global	3	424
33	Meters shown at 1% have no visible color	Global	3	425
34	<Enter> does not make the help window disappear	Global	3	425
35	Numbers are sometimes unreadable on meters	Global	3	425
36	Users confused by addition of empty entry widget	Global	3	425
37	User surprised indicator lights are dynamic	Local	3	426
38	User confused by the behavior of the Hoard Walk indicator	Local	3	426
39	Users confused by two-color blinking	Local	3	426
40	Users somewhat confused because the indicator stops blinking	Local	3	427
41	User disagrees with default notification for certain events	Local	3	427
42	Users search for a way to configure events	Local	3	427
43	Users unsure how an event will be notified	Local	3	428
44	User assumes system uses a 24-hour clock	Local	3	428
45	Users confused by the meters in the "Space Information" window	Local	3	428

(continued on next page)

	(continued from previous page)			
46	User confused by "questionnaire already running" dialog	Local	3	429
47	The Tix tree widget requires three clicks to expanding/contracting nodes	Local	3	429
48	User dislikes having fetch estimates expressed in seconds	Local	3	429
49	User confused by "unselect" dialog	Local	3	429
50	Task definitions unnatural	Local	3	430
51	Users confused by the cache meter in the "Task Information" window	Local	3	430
52	Newly hoarded task may not be visible in list of hoarded tasks	Local	3	430
53	Users confused by the task meters	Local	3	430
54	User finds it strange that a single click selects an item for the contains list	Local	3	431
55	Interface double-selected elements under certain circumstances	Local	3	431
56	Extraneous underlining appears in lists of the "Task Definition" window	Local	3	431
57	Priorities need to be labeled	Local	3	431
58	Meter may hide long task names	Local	3	431
59	User types a pathname into the name of a program definition	Local	3	432
60	User doesn't believe system verifies existence of executable programs	Local	3	432
61	Meters do not cover orders of magnitude or relative information very well	Global	4	432
62	User requests balloon help	Global	4	433
63	User believes task interface to be overly complicated	Global	4	433
64	Interface is cognizant of vertical screen real estate, but not horizontal	Local	4	433
65	Users would like help on the indicator lights window	Local	4	434
66	User concerned about missing an event notification	Local	4	434
67	User would like feedback regarding the "health" of Venus	Local	4	434
68	User would like different configuration settings	Local	4	434
69	User would like a set of system defaults	Local	4	435
70	User wants to modify all event notifications at once	Local	4	435
71	User would like to be able to stop long transfers over weak networks	Local	4	435
72	User would like a count-down timer on advice requests	Local	4	435
73	User wants expert help for Space problems	Local	4	436
74	The Network help screen should explain why the network is degraded	Local	4	436
75	User wants the Task help screen to explain why a task is unavailable	Local	4	436
76	System should predict if hoarding a task would cause space problems	Local	4	436
77	User would like a tree view of tasks	Local	4	437
78	The "Data Definition" window should indicate the size of subtrees	Local	4	437
79	User wants more from program definitions	Local	4	437
80	User expects interface to track and automatically hoard missing objects	Local	4	438
81	Bogus fetch statistics confuse user	Local	4	438
82	Users confused when a file becomes unavailable	Local	4	438
83	User uncertain whether or not a program is available	Local	4	438
84	Users express concern over difficulty knowing what data is required	Local	4	439
85	User would like a graphical representation of what is in the cache	Local	4	439
86	User would like a visual representation of Venus' on-going activities	Local	4	439
87	Users may feel limited by constraint that only tasks can be hoarded	Local	4	440

Figure 7.4: Summary of Findings

This table summarizes the findings that appear in Appendix F. For each, I give the finding identifier number, a brief description of the finding, and the page number of Appendix F on which the complete description can be found.

This section begins by presenting the analysis of quantitative data collected during the Tutorial and the Exercises from the six expert and novice users. It then presents a summary of the qualitative data collected from a questionnaire administered immediately after these participants completed the Exercises. It continues by presenting an analysis of the comments elicited from users during the course of the test. The analysis of the comments includes data from all participants, including the expert, novice, pilot, and miscellaneous users.

7.5.1 Tutorial

In this first section, I use the timings collected as the expert and novice users learned about the interface to identify individual screens of the Tutorial that caused these users difficulty. I then reanalyze the data, isolating novice data from expert data to identify screens that presented difficulties to only one of the two groups.

7.5.1.1 Identifying Difficulty Screens

To identify difficult screens, I use a simple heuristic. I begin by calculating each user's reading rate on each screen of the Tutorial. I then calculate each user's average reading rate on this material as well as the standard deviation. I hypothesize that the distribution of each user's individual screen reading rates is approximately normal, making the average and standard deviation both meaningful metrics. The heuristic then defines a *difficult* screen for a particular user to be one for which that user's reading rate on that screen is more than one standard deviation below that user's average reading rate. A *very difficult* screen is more than two standard deviations below the user's average. I define *easy* and *very easy* in a similar manner. By examining the number of users who found each screen very easy, easy, neutral, difficult, or very difficult, I can estimate the difficulty of each screen of the Tutorial.

The key to this metric is calculating a user's reading rate on each screen of the Tutorial, given the following pieces of information:

- *Dwell time*—the time a user spent on the screen
- *Word count*—the number of words users were required to read
- *Operator count*—the number of operations (e.g., pointing the mouse, clicking a button, or typing a pathname) required to complete the screen

The operator count is derived from a Keystroke-Level Model [7] of each screen of the Tutorial (see Appendix E). The reading rate is calculated by the following equation:

$$ReadingRate = \frac{WordCount}{DwellTime - \sum_{k \in operators} N_k * T_k}$$

Basically, the time spent reading is calculated by subtracting the time spent performing actions from the time spent on the screen.

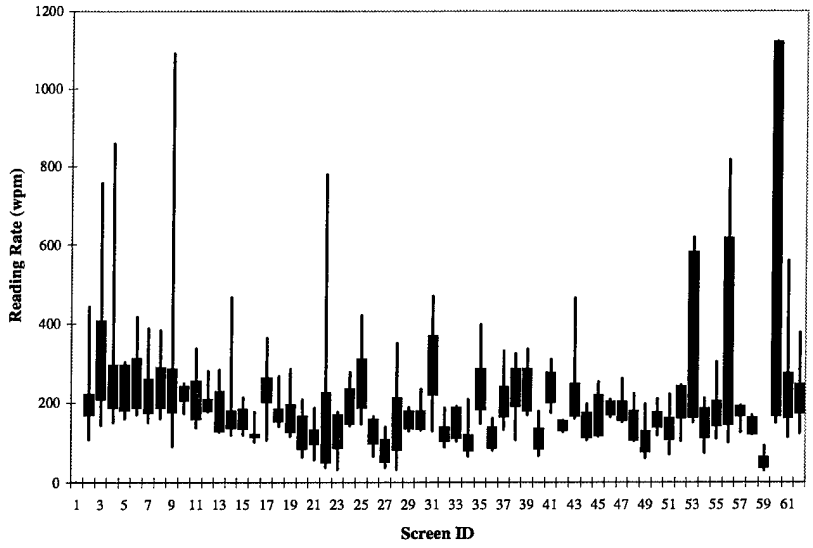
Figure 7.5(a) shows the range of reading rates observed over each screen of the Tutorial. The reading rates generally fall in the range of 100-300 words per minute. Given the educational background of the participants, I would expect their mean reading rate to be about 275 wpm for nontechnical topics [6]. The figure shows that our participants fall well below this expected rate on a number of screens—an indication that they were experiencing difficulty understanding the content of those screens. Perhaps the most striking observation is the number of extremely high reading rates observed on a few of the screens, some of which exceed 1000 wpm! The highest reasonable reading rate is about 650 wpm [7] so users who exceed this rate probably did not read every word on that particular screen. For this reason, I threw out any reading rates exceeding 650 wpm (a total of seven from four different users: N4, E1, E2 and E3). Figure 7.5(b) shows the data reanalyzed.

Using these data, I calculated the average reading rates over all screens of the Tutorial as well as the standard deviations for each of the expert and novice users. For each screen and each user, the heuristic identifies whether the given user found the particular screen very easy, easy, difficult, very difficult, or neither easy nor difficult. Figure 7.6 shows the Tutorial screen analysis visually.

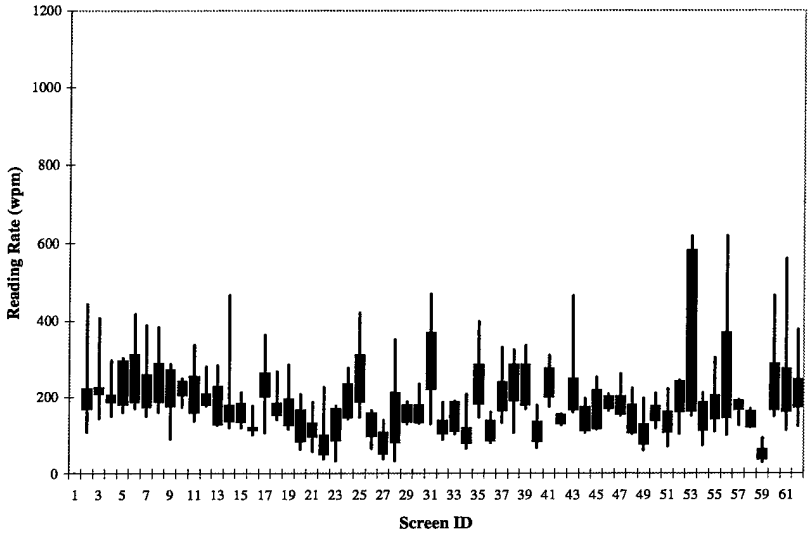
7.5.1.2 Examining Difficult Screens

Although I will not present every piece of evidence this analysis uncovered, I would like to highlight some of the more compelling ones. The analysis identified two screens (those numbered 27 and 59) that were difficult for five out of six of our novice and expert users! It also identified five screens (those numbered 22, 23, 28, 34 and 49) that were difficult for at least three users. In the paragraphs that follow, I will discuss each of these seven screens. The first one will be presented in detail and the remaining ones will be summarized (the complete information for all of them is available in the Appendices).

Screen 27 was identified as difficult for five out of the six novice and expert users! As reproduced in Figure 7.7, Screen 27 of Appendix B presents the way in which meta-information can be entered into the *Data Definition* window. This evidence, combined with numerous other pieces of evidence gleaned from other segments of the test, resulted in Finding 17. The detailed report of this finding appears in Appendix F, but is reproduced as Figure 7.8 for convenience. Each finding of Appendix F contains four sections: a brief description, a listing of evidence, a detailed explanation, and recommendations for addressing the problem. The evidence for the problems comes from five sources: violations of known heuristics, results from the Tutorial or Exercises, comments made on the Evaluation Questionnaire administered at the end of the test, and comments made spontaneously during the course of the test. A brief description of each piece of evidence is available in Appendix E. For this finding, every segment of the test contributed one or more pieces of evidence. The finding that resulted from the Tutorial,



(a) Reading Rates (original analysis)



(b) Reading Rates (sans skimming)

Figure 7.5: Range of Reading Rates

This graph shows the range of reading rates observed from the six novice and expert users. Each screen of the Tutorial is shown along the X-axis (Screen ID numbers correspond to those shown in Appendix B). The reading rate, in words per minute, is shown along the Y-axis. The thin bar shown for each screen represents the full range of reading rates; the thick bar represents the range after excluding both the fastest and slowest reader. Note that the initial screen (Screen 1) was not timed and that two users both read Screen 60 at the same, very high, rate. View (a) shows the range of reading rates from an initial analysis. View (b) shows the range after throwing out those data resulting from users skimming (rate > 650 wpm) the given screen.

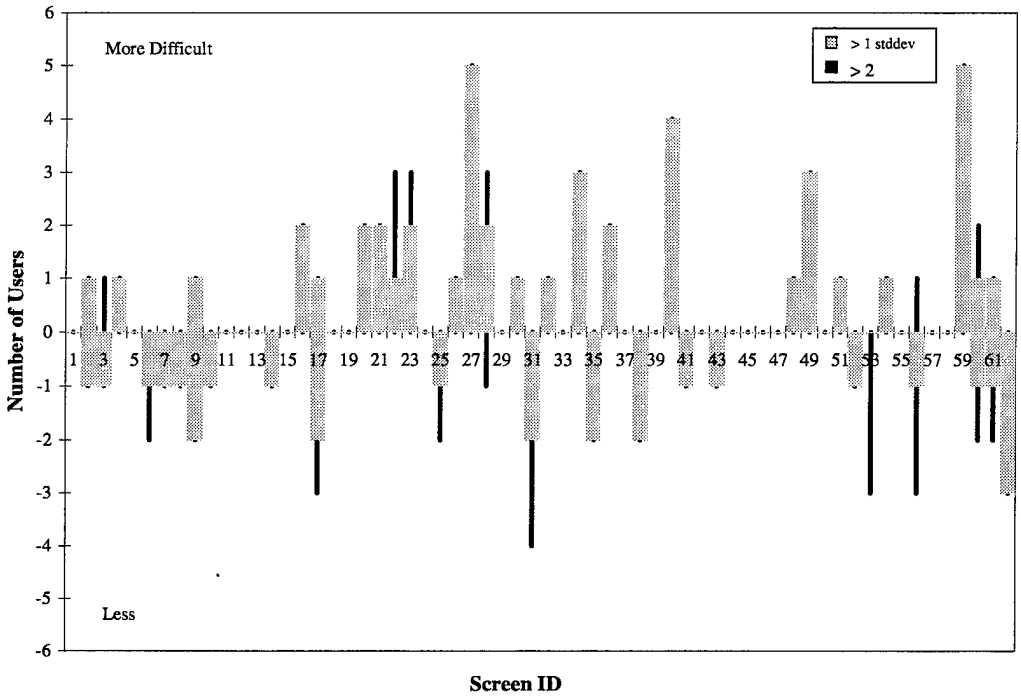


Figure 7.6: Screen Difficulty

This chart shows estimates of the difficulty of each screen of the Tutorial. The numbers on the X-axis represent Screen ID numbers and correspond to the numbered screens of Appendix B. The Y-axis shows the number of users. Each bar represents the number of novice and expert users who found the given screen easy or difficult (depending upon whether the bar appears above or below the X-axis). Each of the gray bars represent the number of users whose reading rate on the given screen was greater or less than one standard deviation above or below their average reading rate. Each black line represents the number of users above or below two standard deviations. Screens without bars were neither easy nor difficult for these users.

→ Please type in pathnames (feel free to make some up) that should be included in the new definition.

If the pathname you type is a file, then, when you hit <Enter>, an empty entry widget will appear, allowing you to enter the next pathname. If the pathname you type is a directory (yes, the system does check), then, when you hit either <Tab> or <Enter>, the meta-information area will become enabled, allowing you to choose the appropriate meta-information for this entry. If you used <Enter>, an empty entry widget will also appear and your cursor will be moved down. You'll need to use the mouse to select the appropriate meta-information. If you used <Tab>, you'll be able to use <Tab> and <Enter> to move around and select the meta-information respectively. If you continue to tab after selecting the meta-information, a new entry widget will appear and your cursor will also be moved down.

→ When you have completed the definition, click the "Save" button.

Figure 7.7: Screen 27

This figure shows the content of Screen 27 of the Tutorial. It is a duplicate of Screen 27 shown in Appendix B, and is reproduced here for convenience.

Finding #17: Users have difficulty entering data into "Data Definition" window
Evidence:
Heuristic Violation: Feedback
Error Messages
Prevent Errors (modal behavior)
Tutorial Results: 6
Exercise Results: 6
Qualitative Results: 2, 7, 9
Comment IDs: 3, 20, 36, 56, 101, 132, 196
Explanation: Users experienced two problems while defining user data elements. The first of these problems occurred when users entered a non-existent directory into the pathname entry area of the window. When this happened, users expected the meta-information area to become enabled (because they had entered the name of a directory). Because the system could not verify that the pathname entered was a directory, it did not enable the directory. There was no other feedback as to the problem. Indeed, the interface didn't even recognize the problem because it assumed the pathname to be a file. The second problem occurred when users tried to specify meta-information using <Tab> and <Enter>. Users were confused about the effect of these keys and were unsure about when each was appropriate to use.
Recommendations: The first problem is easy to remedy. The interface must check that the input pathname exists before checking to see if the input pathname is a directory. If the pathname does not exist, the interface must indicate this information to the user. The indication should take the form of a pop-up dialog box initially, but the interface might allow users to customize the indication to a beep or something similar. The second problem is more perplexing. The cause of the confusion is unclear to me. The <Tab> key advances the highlighting to the next area (as it does in many other windows and in other systems). The <Enter> key selects the current area. The solution may be to abandon these accelerators as "problem causing" and find different ones that cause less confusion. Users made a number of suggestions (indirectly by saying that they "expected" a certain key to have some effect). These suggestions included:

- Using <Back-Tab> to go backwards[1]
- Using the arrow keys to move around the meta-information[2]

One user also expected ^U to delete an entire pathname. Users also wanted to be able to click in the disabled meta-information area, though this desire may have been because they expected the meta-information to have been enabled for non-existent directories.

[1] Note that this expectation suggests that at least one user found the <Tab> intuitive.
[2] The right arrow key, then, would work like the <Tab> key does in the current implementation; the left arrow, then, like <Back-Tab> would.

Figure 7.8: Finding 17

This figure shows Finding 17 from Appendix F. There are four main pieces of information. The first section summarizes the problem. The second section gives a listing of the evidence supporting the finding. I examine five types of evidence: violations of known heuristics, results from the Tutorial or Exercises, comments made on the Evaluation Questionnaire administered at the end of the test, and comments made spontaneously during the course of the test. The third section provides a detailed explanation of the problem and the final section provides recommendations for solving the problem.

Tutorial Result 6, documents that "Five users had difficulty entering data into the 'Data Definition' window [Screen 27]." (The other pieces of evidence for this finding will be discussed below.) This evidence contributes directly to the second half of this finding, regarding the use of <Tab> and <Enter> as keyboard accelerators. Given the large number of users experiencing difficulty with this method of input, the recommendation is either to abandon or to supplement these accelerators. The arrow keys provided an obvious alternative set of keyboard accelerators.

Screen 59, reproduced in Figure 7.9, was also identified as difficult for five out of six of the expert and novice users. This screen asks users to examine two event configurations. Not surprisingly, this piece of evidence was supported by numerous other segments of the test. The basic problem is that in order to configure a specific event, users must know on which indicator that event is notified. The layout of this tab of the *Control Panel* is simply inappropriate to the task. It was redesigned during pilot testing based upon a suggestion made by the initial pilot user. The redesign introduced this usability problem. A detailed history of the problem, including the original complaint, appears in Finding 4.

Screen 22, reproduced in Figure 7.10, asks users to double-click on an element in one of two different lists. The KLM model [7] of this screen (see Appendix E for details) assumes that the user picked one of the lists and then double-clicked on the element in that list. If the user actually looked for the element under both lists before double-clicking on one of them, the additional time would not have been accounted for and the screen may have been mistakenly identified as difficult. Because it is impossible to model the screen for both of these behaviors, I choose to model the minimal actions required to complete the instructions. This decision leaves the analysis open to the possibility of falsely identifying this screen as difficult. As it turned out, however, this evidence was supported by three different comments (Comment IDs 18, 110 and 123) made by a total of six of the participants. Together, this evidence resulted in Finding 19, a Level 2 usability problem.

Screen 23, reproduced in Figure 7.11, introduced users to the *Data Definition* window by describing its contents verbally. Users apparently had difficulty orienting to this window. For example, when the screen contained a statement like "you should see X", users then searched for X in the window. The analysis has likely resulted in a false-positive identification of this screen as difficult because the models do not account for the time needed to verify information between the instructions and the interface. This particular screen contained four pieces of information that users may have matched to the window. No evidence from other segments of the test supports this piece of evidence; therefore, no findings resulted from it.

Screen 28 asked users to close any open *Data Definition* windows. It contained a total of 19 words: "We have now completed our look into defining user data. Please close all windows titled 'User Data Definition'." This window lacks technical content and users had recently closed other windows (in Screens 14 and 17) so the explanation for their difficulty must lie elsewhere. Other similar screens (Screens 17, 53, 56) contained extremely similar content and directions, but did not present difficulties to users. The main difference between Screen 28 and these other screens is the wording "Please close

→ Under the "Advice" indicator, please select the "Read Disconnected Cache Miss" event.

Notice that this event is not configured to notify the user; therefore, the notification options are all disabled.

Please examine how the "Operating Disconnected" event (under the "Network" indicator) is configured. How would you describe this event and its configuration? The next screen gives the answer for you to verify your understanding of event configurations.

Figure 7.9: Screen 59

This figure shows the content of Screen 59 of the Tutorial. It is a duplicate of Screen 59 shown in Appendix B, and is reproduced here for convenience.

Now, let's look at the user data for the "writing thesis" task in more detail.

→ Please double-click on the "thesis" element in either the predefined user data list or the "'writing thesis' contains" list.

Figure 7.10: Screen 22

This figure shows the content of Screen 22 of the Tutorial. It is a duplicate of Screen 22 shown in Appendix B, and is reproduced here for convenience.

Double-clicking on the name of a user data set pops up a "User Data Definition" window showing the definition of the selected user data. The user data definition contains pathnames to files and/or directories. In the "thesis" definition, you should see three directories:

```
/coda/usr/bovik/thesis/
/coda/usr/bovik/thesis/dissertation/
/coda/usr/bovik/bib/
```

The first of these is the top-level of Harry Bovik's thesis area. The second is the directory in which he stores his dissertation. The third is the top-level of his bibliography directory.

You'll notice an empty entry widget after the bibliography directory. This empty entry is not part of the definition -- it just allows you to continue adding more paths to the definition.

Figure 7.11: Screen 23

This figure shows the content of Screen 23 of the Tutorial. It is a duplicate of Screen 23 shown in Appendix B, and is reproduced here for convenience.

all windows...". If users had not deviated from the directions, they would have had just a single window titled 'User Data Definition' opened on their monitor. One possible explanation for their difficulty then is that they were confused by the wording and expected to see more than one open window. In fact, this explanation is supported by comments (Comment ID 34) made by one participant (T1). Another explanation is simply that they had more windows open and had to select the correct one to close (though that window should have been the one on top). The KLM models do not account for this selection. Because this screen contains just 19 words, any modeling errors, any approximation errors inherent in the KLM technique or any unrelated pause during reading (e.g., sneezing or yawning), will be pronounced in the reading rate. The screen simply contains too few words to amortize the effect of those errors. I conclude that this screen represents a false-positive identification.

Screen 34, reproduced in Figure 7.12, lead users through the process of modifying a task definition. Ten users commented on many aspects of this process during the course of the test (see Comment IDs 5, 6, 7, 23, 62, 123 and 181). Together, this evidence resulted in no fewer than five separate usability findings (see Findings 19, 20, 26, 54, 56) related to the *Task Definition* window. Any or all of these findings could have contributed to the difficulty users experienced on Screen 34. The recommendations for addressing these findings include a redesign to present the definitions in a tree format.

Screen 49, reproduced in Figure 7.13, asked users to complete the request for hoard walk advice, to notice that the indicator turned yellow, and to close the *Hoard Walk Information* window. Two primary problems caused users difficulty. Participant E1 noticed some bugs in the Tix widget library that required three clicks to expand a node of the tree; he also had problems with the small active area of the expansion/contraction widget. The second problem was caused by confusion over the *Task Available* event notification. A number of findings contributed to the difficulty users had with this screen; these are Findings 1, 10, 39, 40 and 47. Recommendations for addressing these problems include redesigning the way in which flashing event notifications occur and limiting the indicator from which this request for advice is accessible.

7.5.1.3 Comparing Performance of Novice and Expert Users

The next question I considered was whether or not these data showed a difference between novice and expert performance. Figure 7.14 represents a revised version of Figure 7.6 in which no distinction is made between very difficult and difficult or between very easy and easy. The different colored bars shown for each screen in this graph represent the contributions of novice and expert users. This graph shows that novice and expert users were fairly consistent in their performance, but the results of three screens were particularly interesting.

Screens 20 and 21 are interesting because two of the three novices found them difficult whereas no expert users did. Screen 20, reproduced in Figure 7.15, introduces the *Task Definition* window to users. One possible explanation for why the expert users did not

Your next goal would be to define the "writing with scribe" task. Now that you've saved the "writing with latex" task, you can modify it to create this new task. (Or you can go up, click the black down arrow, select the task named "New...", and start from scratch.) We'll assume that you'll be modifying this task.

First, we should change the name of this task.

→ Click in the "Task Names:" entry area, fix the name, and hit <Enter>.

Next, we should remove LaTeX and xdvi since we can no longer use those tools.

→ Click on "latex" in the Programs area under the contains list.
→ Hit either the <Backspace> or <Delete> key.
→ Repeat for the "xdvi" entry.

Then, we will want to select the newly defined ScribeTools program definition.

→ Click on the "ScribeTools" in the Programs predefined list.

Finally, we want to save this definition.

→ Click the "Save" button.

Figure 7.12: Screen 34

This figure shows the content of Screen 34 of the Tutorial. It is a duplicate of Screen 34 shown in Appendix B, and is reproduced here for convenience.

→ Please request that all the tasks be fetched by using the "Fetch?" button.
→ Then, click on the "Finish Hoard Walk" button in the lower right corner.

Notice that the "Hoard Walk" indicator turns yellow (the warning color) briefly while the hoard walk finishes its simulated fetching of the requested files.

This completes our overview of the "Hoard Walk" indicator.

→ Please close the window labelled "Hoard Walk Information".

Figure 7.13: Screen 49

This figure shows the content of Screen 49 of the Tutorial. It is a duplicate of Screen 49 shown in Appendix B, and is reproduced here for convenience.

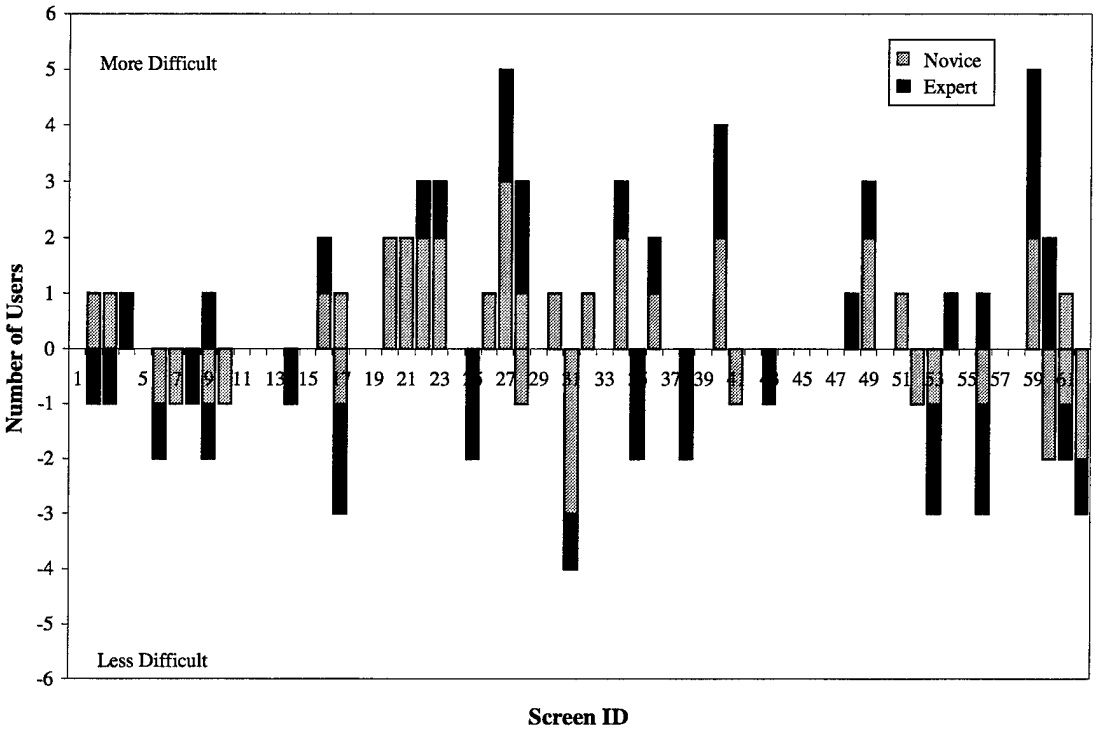


Figure 7.14: Comparison of Expert and Novice Contributions

This chart shows the contributions of expert and novice users to the difficulty rating of each screen of the Tutorial. The numbers of the X-axis represent screen numbers and correspond to the numbered screens of Appendix B. The Y-axis shows the number of users. Each bar represents the number of users who found the screen easy or difficult. The light-colored segments of each bar represent novice users; the dark-colored bars represent expert users. No distinction is made between difficult and very difficult or between easy and very easy in this chart.

find this screen difficult is because they may have heard about on-going research into "task-based hoarding" so the concept might have been seeded, allowing them to grasp the details more readily. Screen 21, reproduced in Figure 7.16, informs users that they can open task definitions by double-clicking in either of two places. For this screen, expert users should not have had any advantage.

Screen 60, reproduced in Figure 7.17, is also interesting because two of the three experts found it difficult while two of the three novices found it easy. Screen 60 contains the "answer" to a question asked on the previous screen (Screen 59 of Figure 7.9). I have no explanation for why experts found reading the "answer" difficult.

In general, the analysis does not show obvious evidence that novice performance and expert performance differs tremendously. Both novices and experts contribute to most of the screens categorized as easy and as difficult. The few screens where their performance differs do not suggest any problems that need to be addressed.

7.5.2 Exercises

In this section, I use timings as well as correctness and efficiency ratings to identify individual exercises that caused users difficulty. I examine the number of users receiving incorrect scores on each exercise. I then examine the number of users who required an excessive number of steps to complete each exercise. Finally, I look at the relative amount of time each user spent on each exercise as compared to the total time that user spent on the Exercises. The data presented in this section were collected from the six expert and novice users.

7.5.2.1 Number of Incorrect Responses

The first heuristic for identifying troublesome exercises simply counts the number of users receiving less than full credit for a given exercise. Figure 7.18 shows this metric visually. The X-axis represents exercise identification strings. The numbers on the Y-axis represent the number of users receiving less than full credit. Each bar represents the number of users who answered the given exercise incorrectly.

As seen in this chart, only three exercises caused more than one user to respond with incorrect answers. These exercises were ProvideHoardWalkAdvice, SpaceStatus, and FixingBugsTasks. I examine each of these exercises in detail.

The first exercise, ProvideHoardWalkAdvice (Screen 71 as reproduced in Figure 7.19), appears to have been poorly designed. This exercise set up a scenario and asked users to provide hoard walk advice in preparation for disconnected operation. The advice request showed three tasks needing data, two of which were relevant to the work described in the background scenario. In addition, those two tasks both needed to fetch data for the editing subtask. Users were expected to deduce that a previous (but still usable) version of the editor was available and that the data needing to be fetched represented a new (and

Double-clicking on the name of a task pops up a "Task Definition" window showing the definition of the selected task. This window has three main sections: one for subtasks, one for programs, and one for user data. Each of these three sections is split into two parts. The list on the left side shows a complete listing of all currently defined tasks, programs, or user data respectively. The list on the right side shows those definitions that are included in this task.

Let's look at the last main section---the one labelled "User Data". The list on the left shows the name of every set of user data that has been defined to date (including "advice sources", and "thesis"). The list on the right shows the subset of user data sets that have been included in the "writing thesis" task definition (just "thesis"). The other two sections, the one labelled "Subtasks" and the one labelled "Programs", are similar.

By looking at the three lists on the right, we see that the definition for "writing thesis" contains the "writing" subtask, the programs for "ScreenCapture", and the "thesis" user data.

Figure 7.15: Screen 20

This figure shows the content of Screen 20 of the Tutorial. It is a duplicate of Screen 20 shown in Appendix B, and is reproduced here for convenience.

In order to view another task definition, double-click on its name in either the "Task Information" window (as you just did) or even in the "Task Definition" window (often more convenient).

Figure 7.16: Screen 21

This figure shows the content of Screen 21 of the Tutorial. It is a duplicate of Screen 21 shown in Appendix B, and is reproduced here for convenience.

The "Operating Disconnected" event is configured to notify the user with a critical urgency level by flashing the "Network" indicator light but not beeping or popping up a window.

Figure 7.17: Screen 60

This figure shows the content of Screen 60 of the Tutorial. It is a duplicate of Screen 60 shown in Appendix B, and is reproduced here for convenience.

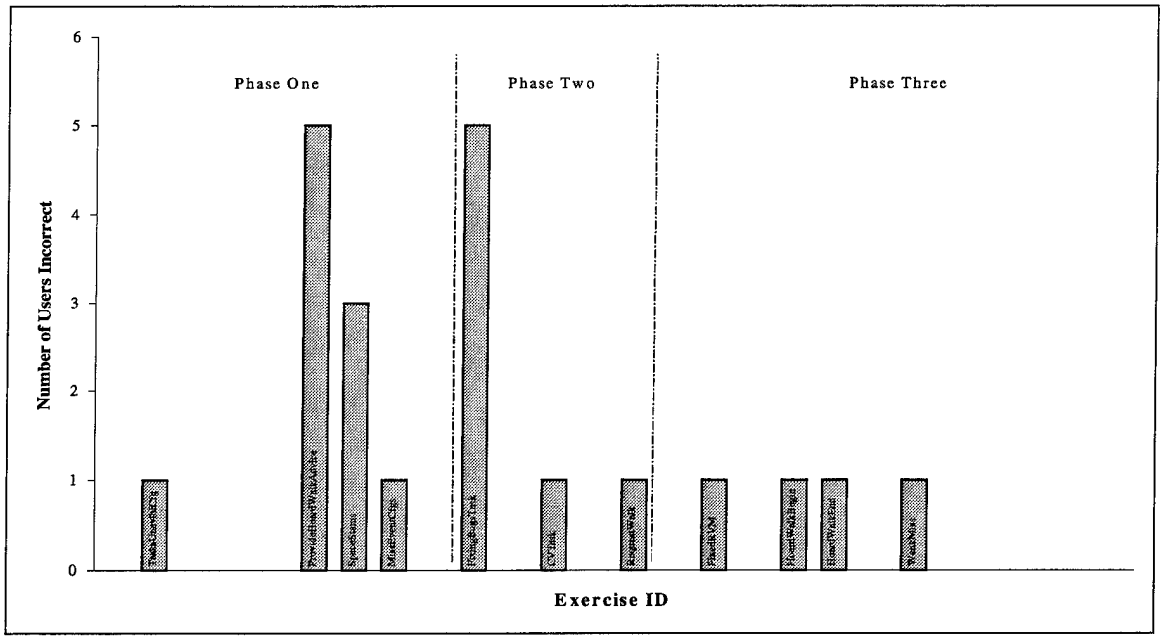


Figure 7.18: Number of Incorrect Responses

This chart shows the number of users answering each exercise incorrectly. The X-axis shows each exercise identifier; the Y-axis shows the number of users. The height of the bar represents the number of users who answered the given question incorrectly. Those exercises without a bar were answered correctly by all six novice and expert users. Only three exercises caused more than one user problems.

Now, suppose you are attending a PI Workshop (attendance required to continue to receive grants from this particular funding agency) in Nowheresville, IA. You had to arrive a day and a half early to get cheap tickets and now you're bored to tears. Thankfully, you have plenty of work to do while you wait, but unfortunately, you were unable to hoard prior to leaving for the airport.

You need to hoard two tasks at this time. First, a colleague asked you to preview a submission for SOSP16. You need to pick up the latest version and read it so that you can send comments back to the authors. If you have time, you'd also like to work on a few letters of recommendation. These two tasks have already been defined ("SOSP16 Paper" and "Letters of Recommendation").

You have selected two tasks to be hoarded and have recently made a connection via an (expensive) long-distance telephone line. Just a minute ago, you requested that a hoard walk be performed. Now, you notice that the hoard daemon has requested advice. Please examine the request, decide what objects should be fetched at this time, and describe your reasoning below.

Figure 7.19: Screen 71

This figure shows the content of Screen 71 of the Exercises. It is a duplicate of Screen 71 shown in Appendix B, and is reproduced here for convenience.

presumably unnecessary) version. Information vital to this conclusion was available under the *Task* indicator. Users had just completed an exercise in which they looked at the relevant information, but few of them made the connection.

Users received one point for not selecting the irrelevant task and a second point for not selecting editing. All six of the novice and expert participants realized[29] that the third task was irrelevant to the scenario. Five of the six, however, fetched the editing subtask. Reconsidering the exercise, the level of understanding needed to complete it correctly is very high indeed. The real goal of this aspect of the exercise was to observe whether or not users could give detailed hoard advice. Examining the advice given by each user in this light, I find that all six users explored the details and that four of them specified more detailed advice than just "fetch the two relevant tasks". Overall, the user's ability to understand what the hoard daemon was planning to fetch is encouraging. From my experience with this exercise however, I must conclude that if we expect users to avoid fetching new versions (upgrades) of binaries, we must provide them with more explicit information. Findings 75, 82 and 83 all address this basic issue.

The second exercise, SpaceStatus (Screen 72 as reproduced in Figure 7.20), asked users about the status of the cache, the local disk, and the RVM. Three users performed one or more components of this exercise incorrectly. It appears that the questions were poorly worded. Two users misinterpreted the first question of this exercise, regarding the current space status, and responded that the cache was two thirds full (which, incidentally, was correct). Two users also answered the second question, regarding how much disk space is dedicated to the Coda cache, incorrectly. One of these users answered in the wrong units, percent rather than MB, but provided the correct value for those units. The other user gave the size of the local disk, perhaps reading the value from the wrong meter or perhaps not reading the entire question. Although three users answered these questions incorrectly, their mistakes suggest that the two questions involved were easily misinterpreted. I conclude that the interface is not the primary cause of trouble.

The final exercise, FixingBugTasks (Screen 76 as reproduced in Figure 7.21), asked users to hoard data related to fixing Coda bugs. The results of this exercise are quite disturbing because only a single user completed it successfully. All of the remaining users failed to hoard the `compress`-related programs (the bug reports were stored in compressed format). Two of them also failed to include the editing task as a subtask, though they

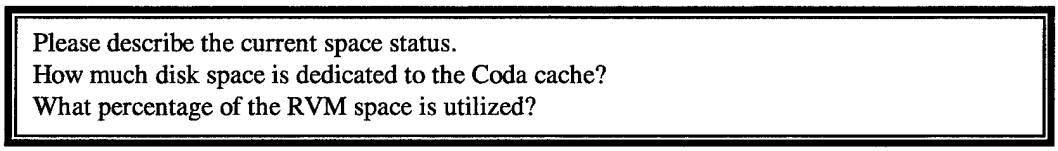


Figure 7.20: Screen 72

This figure shows the content of Screen 72 of the Exercises. It is a duplicate of Screen 72 shown in Appendix B, and is reproduced here for convenience.

[29] One user (N4) identified those objects she intended to select, but failed to complete the actions required to carry out those selections. My analysis is based upon her reasoning rather than her actions.

You are behind on addressing recent bugs reported by users. You'd like to take this occasion to catch up. You'll need the Coda sources, the bug reports, and the build environment.

The sources to the code that you must modify are located in three places:

```
/coda/usr/bovik/src/venus/
/coda/usr/bovik/src/advice/
/coda/usr/bovik/src/console/
```

You've squirreled the original reports away in

```
/coda/usr/bovik/src/bugs/
```

The programs you need to build Coda have been defined as ``building coda".
The last time you hacked disconnected, you realized that you had forgotten to include gdb in your building Coda definition. It can be found in

```
/coda/misc/gnu-comp/i386_mach/omega/bin/gdb
```

Please create a task for fixing Coda bugs.

Figure 7.21: Screen 76

This figure shows the content of Screen 76 of the Exercises. It is a duplicate of Screen 76 shown in Appendix B, and is reproduced here for convenience.

each hoarded it separately at a lower priority. The failure of so many users to completely specify task definitions highlights a serious deficiency in the interface (see Finding 3), one that must be addressed. The failure of two users to hoard the editing task as a subset of their definition or even as a high priority top-level task suggests users may not fully understand the consequences of their chosen priorities (see Finding 11).

7.5.2.2 Number of Inefficient Responses

The second heuristic for identifying troublesome exercises counts the number of users requiring an excessive amount of time to complete a given exercise. I consider users to have taken an excessive amount of time if the number of steps they required to complete an exercise exceeded twice that required by an expert user of the interface with knowledge of the correct answers (2*par). Figure 7.22 presents this metric visually. The X-axis shows exercise identification strings. The numbers on the Y-axis represent the numbers of users requiring excessive amounts of time. The dark gray segments of each bar in the chart represent users exceeding 2*par. The light gray segments represent those exceeding 3*par.

As seen in this chart, a number of exercises required excessive numbers of steps for multiple users. In this section, I examine the causes for five of these exercises: ProvideHoardWalkAdvice (Screen 71), MissEventCfgs (Screen 73), HeaderFilesTask (Screen 77), CVTask (Screen 78), and PrioritizeTasks (Screen 79). The other exercises identified were either only troublesome for a single user or had par values under 5 with no individual user's score exceeding 10. In either of those cases, I did not consider the exercise problematic.

The first exercise identified by this metric, HoardWalkAdvice (Screen 71 as reproduced in Figure 7.19), had a par of 10 and a range of user scores from 20 to 46. This exercise asked users to provide detailed advice to the hoard daemon. Users spent time exploring the details of what objects needed to be fetched. Five to ten node-expansion/contraction pairs would account for 10 to 20 points. Add to that some allowance for users to select some elements and deselect them during the normal process of problem solving, and the fact that so many users exceeded par by 10 to 36 points is not unreasonable. From these data alone, I would not conclude that this exercise presents a significant problem to users, though it could play a supporting role when combined with other evidence of a problem. No other evidence of such a problem appeared during the course of the user test.

The second one identified was the MissEventCfgs (Screen 73 as reproduced in Figure 7.23) exercise. In this exercise, users were asked to describe the configuration of the Weakly Connected Cache Miss Advice event and the Disconnected Cache Miss Advice event. Two users took excessively many steps to complete this exercise. In both cases, their activities reveal that they had difficulty finding these events in the *Event*

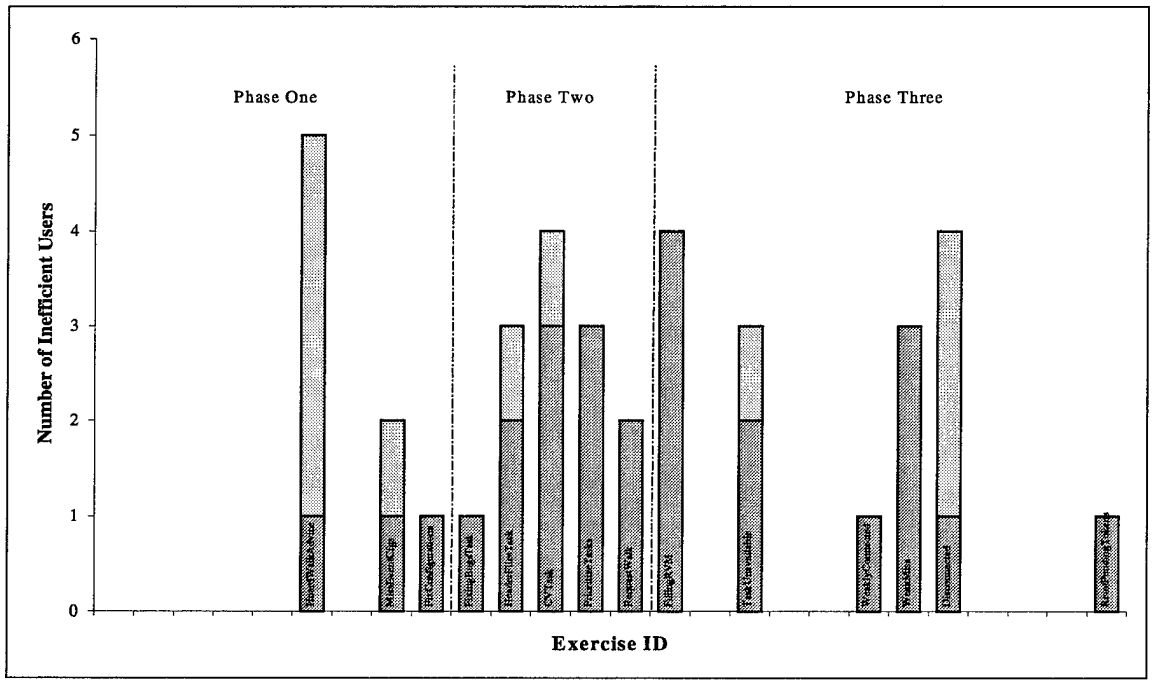


Figure 7.22: Number of Inefficient Responses

Each bar in this chart shows the number of users whose actions were inefficient, meaning the number of actions required to complete an exercise exceeded par substantially. The X-axis shows each exercise identifier; the Y-axis shows the number of users. Each bar represents the number of users who performed inefficiently on a given exercise. The dark gray segment of the bar represent those users exceeding 2*par; the light gray segments represent those exceeding 3*par. Users experienced the most inefficiency during Phase Two and while providing hoard walk advice. The tasks in Phase Three could be completed with a small number of actions. Any additional exploration could cause a user's efficiency to exceed two or three par.

Configuration tab of the *Control Panel*. The fundamental problem is that to find an event on this window, the user must know on which indicator that event is notified. Often it is not too difficult to guess because the name of one of the indicator lights appears in the name of the event (e.g., One or More Tasks Unavailable). In the case of these two miss events, however, three indicators are possible—*Network*, *Space*, and *Advice*. This exercise provides another piece of evidence for Finding 4.

Users who experienced numerous minor inefficiencies caused this metric to identify the third and fourth of these exercises, HeaderFilesTask (Screen 77 as reproduced in Figure 7.24) and CVTask (Screen 78 as reproduced in Figure 7.25). Some users encountered bugs that caused the system to lose data, forcing them to reconstruct it. Some explored the definitions to see what was available for them to reuse (as the instruction, "Don't forget—you can reuse data, program, and task definitions!", in Screen 75 told them to do). These minor inefficiencies account for those users exceeding 2*par. The user (E3, an expert) exceeding 3*par on the HeaderFilesTask exercise used `@sys` in a pathname, failing to heed the instruction of Screen 75 to "pretend that the machine you are running on is an i386_mach box". Although the other two experts also failed to heed this directive[30], they each recognized their deviation from the directions relatively quickly. User E3, however, struggled for quite some time until he finally realized the problem. The interface contributed substantially to his confusion because of the lack of feedback when a directory does not exist (see Finding 17). The instructions also contributed to the confusion in not specifically mentioning the type of machine actually in use. The user exceeding 3*par on the CVTask exercise (N1) experienced many minor inefficiencies, one of which was also the result of the lack of feedback for nonexistent directories (this time caused by a typographical error). Although no single problem accounts for all of the inefficiencies experienced by users during these two exercises, the combination of a small number of serious bugs and lack of appropriate feedback accounts for much of it. Future versions of the interface must address these problems (discussed in Findings 15 and 17).

The final exercise identified by this metric was the PrioritizeTasks (Screen 79 as reproduced in Figure 7.26) exercise. In this exercise, users were asked to prioritize the tasks they defined in preparation for disconnection. Three users exceeded 2*par on this exercise. Two of them experienced difficulty with the interface while prioritizing their tasks. Two of them (one of those two users and the third user) were forced to reorganize their task definitions before prioritizing them to make the HeaderFilesTask a top-level task rather than a data definition within the BugFixingTask. They were forced to reorganize their definitions because only tasks can be prioritized and because the instructions asked them to give the HeaderFilesTask a lower priority. The instructions strongly suggested the appropriate organization, yet the intuition of these users led them to a different organization. This behavior suggests that users may feel limited by the constraint that only tasks may be prioritized and that this constraint may need to be reconsidered (see Finding 87).

---

[30] It's interesting to note that all three expert users attempted to use `@sys` whereas not a single novice user did so. Because `@sys` is a feature of not only Coda but also AFS, this difference illustrates a point made early in this chapter that developers (and their friends) are not representative of all computer scientists.

Please describe the "Weakly Connected Cache Miss Advice" and the "Disconnected Cache Miss Advice" events and their configurations in the space below.

Figure 7.23: Screen 73

This figure shows the content of Screen 73 of the Exercises. It is a duplicate of Screen 73 shown in Appendix B, and is reproduced here for convenience.

Oh, and the last time you hacked from home, you wanted to take a look at some header files that you had forgotten to hoard. They aren't really important, but it'd be nice to have them around. Unfortunatley, even though each individual header file is relatively small, together they take up quite a lot of space -- so you don't want to just add them to your ``building coda" programs since then they might be hoarded in preference to really important source files. They are located in several places:

```
/coda/project/coda/alpha/include/
/coda/project/coda/alpha/i386\_mach/include-special/
/coda/misc/c++/i386\_mach/alpha/include/
/coda/misc/tcl/common/beta/include/
```

Figure 7.24: Screen 77

This figure shows the content of Screen 77 of the Exercises. It is a duplicate of Screen 77 shown in Appendix B, and is reproduced here for convenience.

You need to write your CV before Tuesday when you will be talking to a representative from AT\&T Labs (assuming, of course, that the visitor can get here and that you can get to school). Since you haven't yet written your CV, you've borrowed the file for your advisor's CV to use as an example. Unfortunately, he still uses Scribe! Since you're time-pressured, you've decided to stick with Scribe. Please create a task for writing your CV. The necessary files are located in:

```
/coda/usr/bovik/personal/cv/
```

Unlike LaTex, Scribe outputs PostScript directly. To run Scribe, type ``scribe cv.mss"; it will produce a ``cv.ps" file that you can preview with ``gv". You may assume that the binaries for both scribe and gv are located in /coda/misc/bin.

Figure 7.25: Screen 78

This figure shows the content of Screen 78 of the Exercises. It is a duplicate of Screen 78 shown in Appendix B, and is reproduced here for convenience.

Finally, you'll need to prioritize these tasks. Recognize that the data needed for all these tasks may or may not fit into your cache -- the priorities you choose now will decide what (if anything) gets left behind. To help you in prioritizing these tasks, we've listed all of the tasks you would like to work on during the blizzard, as well as what they are intended for. (The list below is in no particular priority order.) If you haven't already hoarded these tasks, please do so now. When you have finished, please use the space below to describe your reasons for choosing the priorities you did.

Fix bugs reported by users.
Various header files to support bug fixing.
Write your CV to give to this visitor on Tuesday.

Figure 7.26: Screen 79

This figure shows the content of Screen 79 of the Exercises. It is a duplicate of Screen 79 shown in Appendix B, and is reproduced here for convenience.

7.5.2.3 Relative Exercise Dwell Time

The third heuristic for analyzing user performance during the Exercises looks at the relative amount of time each user spent on each exercise as compared to the amount of time that user spent on the entire set of exercises. Figure 7.27 shows the data visually. Along the X-axis, I list exercise identification strings. Each bar along the Y-axis represents the percentage of time one user spent on the given exercise as compared to their total time spent on the Exercises. Each exercise shows six bars, one per user.

The chart shows that this metric confirms my previous analysis. Users spent a relatively large amount of time on six of the seven exercises previously identified as difficult. The fact that the seventh exercise, SpaceStatus, is not identified here supports my conclusion that the interface did not cause users difficulty. This metric shows that users did not spend a large amount of time on this exercise. This fact suggests that neither the window nor the questions confused users, supporting my earlier conclusion that the interface was not the primary cause of trouble with this exercise.

One obvious feature of this chart is the large percentage of time participant E1 spent on the DefineBugsTask exercise. This user explored the definitions during the process of defining this task, as he was instructed to do in Screen 75. This exploration is simply reflected here.

This chart does make two interesting points. First, expert users seem to spend less time than novice users providing hoard walk advice, as evidenced by the fact that the slowest expert completed this exercise more quickly than the fastest novice. My explanation for why this might be is that this idea was familiar to each of them prior to participating in the test. Second, with that one exception, the novice users and the expert users spend similar relative amounts of time on each exercise.

7.5.3 Evaluation Questionnaire

In this section, I present a summary of data collected from the questionnaire administered immediately following completion of the Exercises. The data presented were collected from the six expert and novice users. I highlight those responses that address the interface.

Users found the interface easy to use. On a five point scale, where five was "very easy" and 1 was "very difficult", users rated the interface a 4.5. One expert participant commented that he was somewhat confused regarding which indicator to open. The Debriefing revealed that the user was confused regarding the distinction between the *Task* and *Hoard Walk* indicators (see Finding 29).

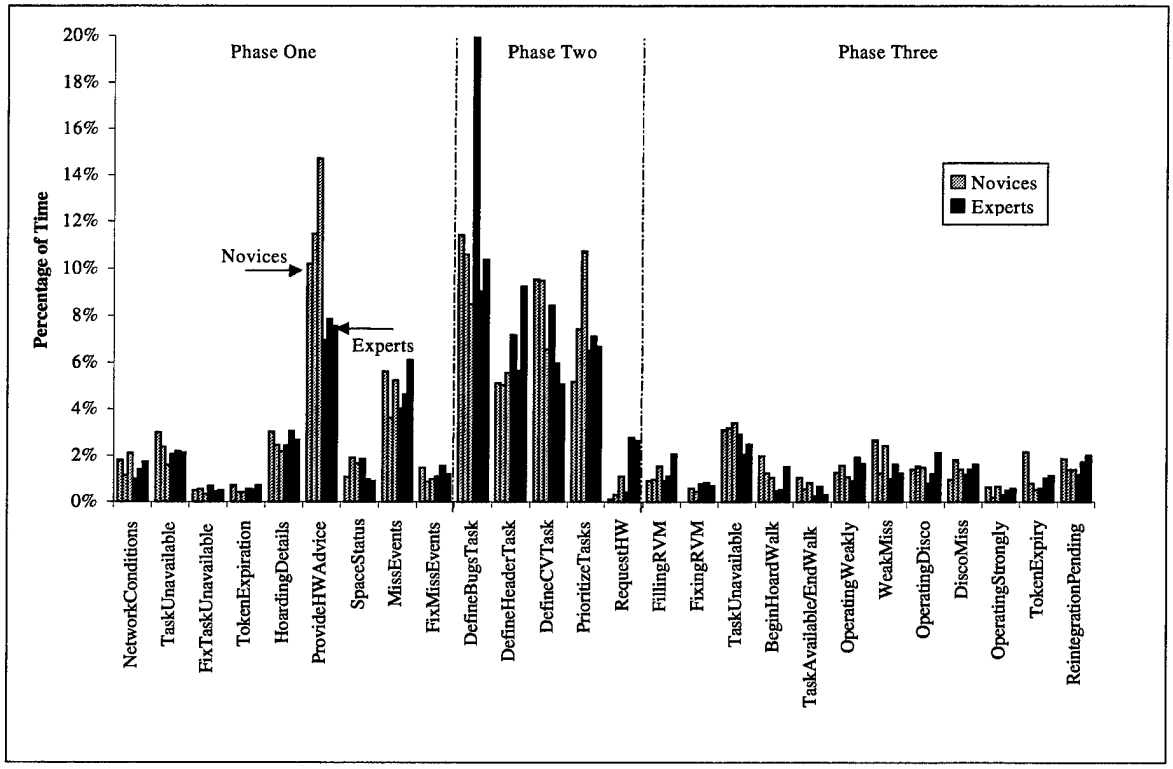


Figure 7.27: Relative Dwell Time

This chart shows the percentage of time each user spent on each exercise. The X-axis identifies the exercises. The Y-axis shows percentage of time relative to the total time the user spent on the Exercises segment of the test. For each exercise, the six bars represent participants N1, N2, N4, E1, E2, and E3 respectively. The light-colored bars represent the novice users; the dark-colored bars represent the experts. The maximum percentage of time spent on any individual exercise was just under 20%. The interesting observation about this graph is that it is not particularly interesting. The expert and novice users both spend about the same relative amount of time on each exercise. The only notable difference is that the novices tend to spend a larger percentage of their time providing hoard walk advice than the experts do. In addition, user E1 spent an excessively large percentage of his time defining the "BugsTask". This user explored the definitions extensively during the process of defining the task. He was complying with the suggestion to reuse definitions.

Users ranked their ability to find the features they wanted in the indicator lights a 4.2 (again, on a five point scale where 5 meant "very easy"). The two users who ranked this question as "neither easy nor difficult" both commented that it was difficult to find events in the *Event Configuration* tab of the *Control Panel.* These comments provide an additional piece of evidence for Finding 4.

Users reported that they were likely to keep the indicators lights visible on their monitor all (or almost all) the time. Their average ranking was a 4.3 out of 5 (where 5 meant "very likely"). This ranking is meaningful because users highly value "screen real estate".

Four users felt that the interface provided an appropriate amount of detail, but two felt that it provided too little. In particular, the two users commented that help was not helpful (see Finding 2) and that the "no such file" indication was too subtle in the *Data Definition* window (see Finding 17).

Users found the concept of hoarding easy to understand; their average ranking was a 4.5 out of 5. They also found that preparing for disconnected operation was easy; their average ranking was a 3.8 out of 5. Furthermore, they were highly confident in their preparation; their average ranking was 4 out of 5. These rankings suggest that users were completely unaware of the subtle mistakes they had made in creating their task definitions.

When asked which aspects of the interface they liked the most, users responded with one or more of the following:

- The ability to define tasks (4 users)
- The ability to see the status of the system (3 user)
- The ability to configure event notifications (2 users)

When asked which aspects of the interface they liked the least, users generally responded with unique answers. In fact, the only answer to be reported more than once was a complaint regarding the way in which meta-information was specified in the *Data Definition* window (see Finding 17).

7.5.4 Verbal Protocol

As part of the analysis, I transcribed portions of the verbal protocol collected from all thirteen users for whom I have recordings, including pilot and miscellaneous users and the novice user whose quantitative data was thrown out (see Section 7.3.1). Afterall, just because a user was a pilot, does not mean that their comments are not valid. My goal in transcribing these tapes was not to have a complete record of each session, but to have a record of the critical incidents. In particular, I did not transcribe segments during which the user read verbatim from either the Tutorial or the Exercises except when necessary to provide context. Appendix D contains the transcripts of each session. Figure 7.28 contains an excerpt from the transcripts of participant E1. As seen in this figure, the

Time	Transcript	ID
	So, walls...I'm guessing walls is this red outline (around the xterm).	285
7:29	[CodaConsole started] Okay, something just popped up. A little box. Control Panel, Tokens, Space, Network, Advice, everything is green. I'll bet that's good.	
8:47	[Color blind– other cultures] OK. So, I can do those colors. I wonder...so, traffic lights have that, th– the ordering and I wonder if something like that might work here too.	
12:55	One of what servers? Had the color been red instead of yellow, the indicator would have been showing that our connection to at least one server had been severed. OK. Double-click on the network indicator for more detail regarding our connection to individual servers. Aha! So that will tell me. I see. OK. Cool.	
13:29	So what if we have 100 Mb/s? Can that be scaled I wonder.	50
14:15	Soon the bandwidth to scarlatti will drop to 0, which it just did, and Oh! It's blinking yellow and red? Ah, I thought it would have just changed red. This indicator flashes red and the network information window automatically updates to show the change in scarlatti's estimated bandwidth. OK. Oh, so it stops flashing... So maybe that's just to draw my attention.	9 33
16:00	[First impressions of Task Information window.] So, cache space. Alright. So the task, I'll read, instead of trying to figure it out. Although, let's see, so it looks like there are 97 MBs available and that 70% of it is either available, 70% of that 97 is available or 97 MB is available. Let's read.	269
	It's too bad this window manager seems to pop everything up up there.	1
22:10	[First impression of Data Definition.] So it looks like a hoard database shown in widget form.	
22:15	So I wonder if there's they have the old hoard now and forever or just now notation as well, I wonder, I'll bet it's just now and forever.	
22:30	So I wonder if they gray out if it's a file.	
23:00	Meta-information is a weird name here.	209
23:30	Ah, yes, so they are grayed out when nothing's there.	
	Interesting that it's not <Tab> [after naming the data definition, you hit <Enter> to move the cursor down into the definition area.]	3
24:35	[Task10: <Tab>/<Enter> explanation for data definition meta-information.] Alright, I'm confused. "If the pathname you type is a directory, then, when you hit either <Tab> or <Enter>, the meta-information area will be enabled." Alright, so let's go see if I can rename this. Ah... Better not try that. <Tab> just selects the window and another <Tab>. OK. Hit either <Tab> or <Enter> the information window "will become enabled, allowing you to choose the appropriate meta-information for this entry. If you used <Enter>, an entry empty entry." OK. So <Tab>, <Tab> sends me to the meta-information entry as does <Enter> and <Enter> additionally adds an empty entry widget. OK. "You'll need to use the mouse to select the appropriate meta-information. If you used <Tab>", mouse, no keyboard shortcuts, "you'll be able to use <Tab> and <Enter> to move around and select the meta-information." Ah! OK. "If you continue to <Tab> after selecting the meta-information, a new entry widget will appear." Well, let's see.	3

(continues)

Figure 7.28: Excerpt from E1 Transcripts

This figure shows an excerpt from the transcripts of the verbal protocol produced by participant E1 during the Tutorial. The complete transcripts appear in Appendix D. The left-hand column of the table gives the timestamp relative to the beginning of the Tutorial, when known. The middle column shows the comments made by the participant, though any comment appearing within square brackets is a note from the transcriber. The right-most column provides indices into the comment listing of Appendix E.

transcripts contain timestamps relative to the beginning of the segment (when those timestamps are known). In addition, they code individual comments made by the user, providing the ID of the comment listed in Appendix E.

Appendix E contains a complete listing of the comments coded from these transcripts in numerical order. This appendix also provides a table documenting which users made each comment during which segments (Tutorial, Exercises, Debriefing) of the test, an excerpt of which is shown in Figure 7.29.

The metric I used to analyze this data was the frequency with which comments were reported. Figure 7.30 shows this information. The X-axis shows the number of users. Each bar represents the percentage of all comments reported by a given number of users. From this chart, I can identify compelling comments as those made by many users. The chart shows, visually, what percentage of comments was made by a given number of users. As is typical in usability testing, most comments were made by just a single user—in this case 58% of them. In fact, of the remaining 42% of comments, more than half were made by just two users. Less than a quarter of the comments were made by three or more users, yet five comments were made by more than half of the users! For the purposes of identifying evidence of usability problems, my analysis examined any comment made by three or more users. Because of space limitations, I discuss only those comments made by five or more users.

As seen in Figure 7.29, ten users found two-color blinking (Comment 9) confusing. When an event configured to flash arrived, the appropriate indicator light would blink between the old urgency color and the new urgency color for about ten seconds. So, for example, if the current state of the *Task* indicator is green, then when the *Task Unavailable* event arrives, the *Task* indicator will flash green-red-green-red-etc. for ten seconds before settling on red. The idea was to show the previous state as well as the new state. Unfortunately, users found it confusing because, unless they remembered the old state, they could not determine what the new state was until the indicator settled on the final color. This evidence lead to Finding 39.

Nine users found that they had to guess which indicator a particular event was associated with when trying to configure an event (Comment 17). This comment, as well as two related comments (those numbered 162 and 206) provides the final piece of evidence for Finding 4. For more information regarding this problem, please refer to Section 7.5.2.2 and Appendix F.

Eight users commented that they wanted the interface to tell them the size of the subtree in the *Data Definition* window so that they could select meta-information more intelligently (Comment 57). Without a doubt, this information would improve the usability of the interface. Unfortunately, obtaining the information efficiently could prove problematic (see Finding 78).

Eight users also commented that they did not know anything about reintegration (Comment 239). The lesson here is that participants did not appreciate being asked to go beyond the lessons covered in the Tutorial. I did so to see how well users would generalize what they had been taught and apply it to a similar situation. They did very well, but clearly indicated that the topic had not been covered.

	Pilots				Novices			Experts			Misc.			Number	Unique
ID	2	3	4	5	1	2	4	1	2	3	C	S	T	Reporting	User
1				T	T			M		A	T		T	6	
2	(See Comments #80 and 209-243)														
3				T	D	T		A	E	F		T		7	
4						T		F		E			T	4	
5					T	T		T	T	T			T	6	
6					F	F		A			E			4	
7			E					T		E				3	
8								T		F			F	3	
9	T		E	T	T		T	T	F	T	E		A	10	
10						T		T						2	

(continues)

Figure 7.29: Excerpt from Table Summarizing Comments

This figure shows an excerpt from a table in Appendix E that summarizes the comments made by participants during the course of the usability test. The leftmost column shows the Comment ID number (as listed in Appendix E). The next 13 columns indicate whether a given user made the comment, and, if so, during which portion of the test. A T indicates that the comment was made during the Tutorial, an E indicates Exercises, a D indicates Debriefing, an F indicates both Tutorial and Exercise, an M indicates both Tutorial and Debriefing, an L indicates both Exercises and Debriefing, and an A indicates Tutorial, Exercises, and Debriefing. The final two columns show the total number of users making the comment and user who made a unique comment (a comment made by only one user).

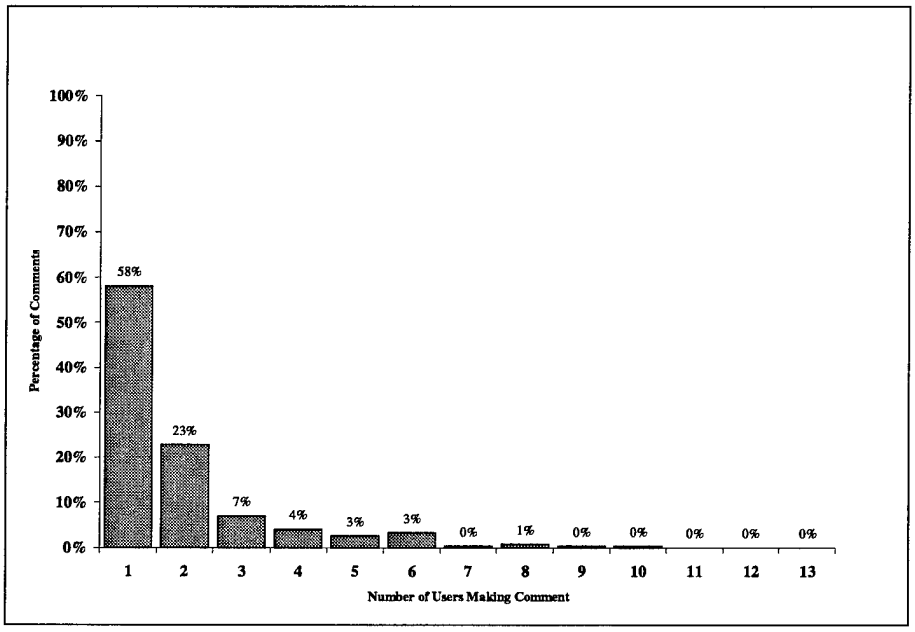


Figure 7.30: Comment Frequency

This chart shows the frequency with which verbal comments were made. Each bar represents the percentage of comments made by the given number of users. Thus, 23% of comments were reported by two users while only 1% of comments were reported by eight users. No comments were made by more than ten users.

Seven users had difficulty entering data into the *Data Definition* window (Comment 3). In particular, they had problems understanding when to use <Tab> and when to use <Enter>. This comment, combined with evidence from every other segment of the test, lead to Finding 17. For a detailed explanation of this problem, please refer to Section 7.5.1.2.

7.6 Usability Test: Findings

The results of the test show that the interface allowed the novice Coda users to use Coda with approximately the proficiency of the experienced Coda users. Novices scored 90% or better on exercises typical of those required of Coda users. They successfully diagnosed problems and possible solutions, defined tasks, and provided advice to the system about hoard walks and demand fetch requests. The interface clearly met each of its objectives.

Not surprisingly, the test also provided a wealth of information about how users interact with the interface and where they encountered problems. Analysis of this evidence reveals areas in which the interface could be improved. This analysis resulted in 87 usability findings. I classified each finding according to its scope and severity (see Section 7.5). Figure 7.31 shows the number of findings in each classification. 24% of the findings had severity level 1 or 2, and 26% had global scope.

Appendix F contains a complete listing of the findings, organized by their severity and scope. Because of space limitations, I cannot present each of these findings here. Instead, I present a different classification of these findings, one based upon the type of problem uncovered:

- Issues dealing with the approach
- Bugs in the interface
- Usability problems
- Omissions from the interface (either accidental or intentional)
- Confusing behaviors
- Suggestions for enhancement

For each classification, I highlight a number of findings. The specific ones highlighted were chosen for a variety of reasons: they might be representative of the entire class of problems; they might be particularly important; or they might have an interesting history. The findings presented in this section should be considered a sample of the type of problems I uncovered, *not* a "short list" of problems that need to be addressed.

7.6.1 Issues with Approach

The test uncovered a small number of usability problems that briefly caused me to question the wisdom of translucent caching. The first of these is user confusion over

event notifications (see Finding 1). Although users were able to determine the cause of problems during Phase Three of the Exercises, they expressed confusion over what had happened at other points during the test. These observations were not limited to the Tutorial so they cannot be categorized as a learning effect. It is certainly reasonable to believe that user understanding will continue to improve with further exposure to the interface, but that does not solve the fundamental issue that certain event notifications were confusing to users. We must ask whether or not there is something the interface could do to reduce this confusion. In fact, these problems suggest that the interface needs to offer a central place where users can go for a written explanation of the current, or most recent, problems. In essence, just as keyboard accelerators help the user tell the interface what to do more quickly, the indicator lights offer the interface an accelerator to tell the user what has happened. They do not provide complete information, however, and so a more descriptive explanation is warranted.

A second usability problem that caused serious concern was the difficulty users had identifying which programs to include in task definitions (see Finding 3). Although users were highly successful in defining task definitions, five out of six of them failed to hoard a program necessary for one of their definitions. This mistake was one of very few that were made by any of the users, yet it was extremely common (even among the pilots and miscellaneous users). It has the potential to make user-assisted hoarding unusable, but it can be addressed with the addition of a new type of advice that warns users about missing programs. Effectively, the interface would have to issue a "watch" on each hoarded file. When a program accessed an object under watch, Venus would inform the interface. If a program accessing hoarded data is not also hoarded, the interface could indicate a potential hoarding problem to the user through the advice monitor at a reasonable time, perhaps at the end of a voluntary hoard walk.

Severity	Scope	Number of Findings
1	Global	1
	Local	10
2	Global	4
	Local	6
3	Global	15
	Local	24
4	Global	3
	Local	24

Figure 7.31: Classification of Findings

Each finding is categorized by its scope and severity. Scope is a measure of how widespread a problem is. A problem with local scope is limited to a single window; one with global scope applies to multiple windows. Problems with global scope are generally considered more critical than problems with local scope. Severity is a measure of how critical a problem is considered. A problem with severity 1 prevented a user from completing a task. Severity 2 problems caused significant delay or frustration. Severity 3 problems had a minimal impact on usability. Severity 4 problems represent more subtle issues and may suggest potential enhancements to the interface.

7.6.2 Bugs

No matter how well tested an interface seems, there is always another bug lurking in the shadows. During the course of the usability test, I found a total of 24 interface bugs (listed in Appendix E). Six of these bugs are critical: one crashes the interface (see Finding 3) and five cause the interface to lose data (see Findings 15 and 18). A number of others are urgent in that the system simply behaves incorrectly. For instance, in certain situations selecting an element in a task definition results in the element appearing in the list twice (see Finding 55). Similarly, when the user types a password into the interface to obtain tokens, the password is echoed in the window using a text color that is almost, but not quite, identical to the background color of the entry widget (see Finding 6). The monitor on which the test was performed allows the user to read the password, whereas the monitor on which the interface was developed did not. A number of other reported bugs were cosmetic in nature, though none of them were reported frequently enough to warrant an official Finding. For example, the labels associated with the radiobuttons on the *Event Configuration* tab of the *Control Panel* do not line up correctly (see Comment ID 79). Similarly, the widgets on that same window visibly shift when a new event is chosen from the listing (see Comment ID 135).

7.6.3 Usability Problems

The data showed that users experienced significant difficulty finding particular events in the *Event Configuration* tab of the *Control Panel* (see Finding 4). Two of the six participants completed the MissEventCfgs exercise inefficiently. During the course of the test, a few users resorted to brute-force search to find an event and one user failed to complete the exercise correctly because of this difficulty. The fact that nine users commented on the difficulty of finding events provides further evidence of this usability problem. This window of the interface must be redesigned such that users are not required to select the indicator on which an event is notified while configuring that event. The redesign must, however, show on which indicator the event will be notified.

Another important discovery was the difficulty users had entering data definitions (see Finding 17). In particular, seven users commented that they were confused by the distinction between <Tab> and <Enter>. Although the test clearly demonstrated that the means of entering these definitions was confusing for more than half of the users, it offered little in the way of explanation for their confusion or suggestions for improvement. The behavior of <Tab> and <Enter> is consistent with other uses of those keyboard accelerators. Thus, the recommended solution is to allow additional keyboard accelerators, such as the arrow keys, for those users confused by the behavior of <Tab> and <Enter>.

Suspiciously enough, a second important discovery also relates to data definitions. That is, when users enter the pathname of a directory that does not exist, the system provides only indirect feedback of the error, by not enabling the meta-information (see Finding 17). Six users commented about this lack of feedback. Furthermore, two users exceeded 3*par

on a task definition exercise largely because of this lack of feedback. I hypothesize that this problem may be at least partially related to the previous one because if the meta-information is not enabled, <Tab> does not allow the user to move the cursor around the meta-information options. After entering a nonexistent directory, users may have been confused because <Tab> did not then work. The recommended solution to this problem is to add a dialog error message to inform users when a pathname that has been entered does not exist. A higher cost solution to this problem would be to add pathname browsing.

The data also show that a number of users had difficulty prioritizing tasks (see Finding 16). Three users exceeded 2*par on the relevant exercise. Two of those users' inefficiency on this task can be directly attributed to problems related to using the prioritization mechanism. Further support for this problem comes from comments made by those two users plus two others who did not exceed 2*par on the relevant exercise. The interface either needs to use traditional drag-n-drop rather than the substitute version, or the *Task Definition* window needs to be revised to avoid the necessity of drag-n-drop.

7.6.4 Omissions

Another class of problems resulted from features left out of the interface: some purposely and some inadvertently. Perhaps the most critical omission in the interface is the lack of an undo facility (see Finding 12). Numerous users accidentally deleted definitions and could not undo their action. Instead, they were required to reconstruct the data they had deleted. A production interface absolutely must have at least a one-level undo facility.

Users also discovered some minor omissions in the interface. They encountered two help windows that were overlooked and provided no useful information, and one that was never updated to reflect a change in the interface (see Finding 2). I also noticed[31] that, if the Space help window is visible on the monitor, it does not dynamically update when its parent window gets updated (see Finding 2). For instance, if it was showing that the cache space is at a warning level and a critical cache space event arrives, the help window does not update to show the messages related to the new space status. Finally, users encountered a few windows that did not increase the size of their widgets in proportion to an increase in their own size. These windows had been overlooked when I added the resize feature to other windows in the interface (see Finding 24).

7.6.5 Confusing Behaviors

Perhaps the most confusing behavior exhibited by the interface is the way in which it flashes the indicator lights to indicate the arrival of an event (see Finding 39). Two-color flashing, from the previous urgency color to the new urgency color, does not allow users to determine the current state of the indicator light until the flashing stops. Two possible solutions are to flash one color longer than the other (e.g., illuminate the new urgency

[31] No users reported this problem.

color for twice as long as the old urgency color) and to abandon two-color flashing altogether (e.g., flash from dark gray to the new urgency color).

Users discovered that the interface offers no feedback regarding the completion of saves or commits (see Finding 13). One of the users was so discombobulated by this behavior that he clicked the save button eight or ten times in a row every time he saved a definition. One common way to provide such feedback, progress meters, is probably inappropriate for application to this problem because the save occurs so quickly that users would be unlikely to notice the meter's brief appearance. Another way to provide the same information, in a way more appropriate to the task at hand, is using the emacs asterisk. This approach uses the asterisk to indicate that the document needs to be saved; once saved, the asterisk disappears.

Although the lack of feedback after completing a save or commit was mildly confusing, it was just the tip of an iceberg (see Finding 13). The interface follows suggestions for maintaining a "unified file model" [8]. In particular, the interface

- Automatically saves definitions when the user closes the definition window
- Does not close a window after the user saves a definition
- Does not pop up the SAVE dialog when the user closes the definition
- Allows the user to abandon changes since opening or creating the definition
- Provides an optimal, manual SAVE command

These behaviors are substantially different from the behavior of traditional interfaces. The suggestions above, however, encompass just those portions of Cooper's complete set of recommendations that seemed to apply to a simple "document" like a definition. Those recommendations not implemented include the ability to snapshot, milestone, rename, place & reposition, specify the stored format, and undo specific changes.

Cooper's recommendations appeal to me because they step back and look at the functionality required of an interface from the perspective of the user rather than from the perspective of the computer. Although his recommendations are still appealing, I now believe that I should have stuck more closely to his model. Keeping some aspects of the traditional file model while adopting some aspects of Cooper's unified file model produces an inconsistent interface that confuses users. The implementation should also include snapshop and milestone capabilities as well as a specialized undo facility. It should not offer a SAVE AS capability, and it should replace in-place naming with in-place renaming. The interface does not need the ability to place & reposition definitions or specify the stored definition format because it manipulates such a specialized and small class of documents.

7.6.6 Enhancements

Users also made numerous suggestions for future enhancements, either directly or indirectly. Some of the suggestions offer minor improvements over the current interface. Other suggestions would require a major redesign of the interface. Four of these suggestions are listed here:

- One user commented that the interface was cognizant of vertical screen real estate, but not of horizontal screen real estate (see Finding 64). This implies that the user might prefer to have the indicator lights appear horizontally across the top or bottom of the monitor rather than vertically along a side.
- More than half of the users commented that they would like to compare the amount of space necessary to hoard the immediate children of a directory as opposed to all the directory's descendants without having to resort to using a shell (see Finding 78).
- Another user suggested a tree view of the task definitions, similar to that shown in the hoard walk advice request (see Finding 77). One difficulty with implementing this view of task definitions is in dealing with the meta-information included in data definitions. Another is that task reuse implies that we do not have a proper tree structure.
- Another user suggested that the interface provide users with a visual display (perhaps a basic `cache-ls`, akin to `ls`, or a graphical tree view) of the contents of the cache (see Finding 85). Undoubtedly, such a feature would be useful for determining whether or not an individual file or directory is available. It also has a high potential for confusing users, however, because large portions of the file system space would necessarily be unavailable in the cache. If users accidentally confused this view of the file system with the full view of the file system, they might become unnecessarily concerned about the "disappearance" of uncached file and directories.

7.7 Reflections

During the course of performing this usability test, I learned much about the process. I experienced difficulty recruiting novice users, largely because I could not find users who fit my definition of novice. In modifying the interface during piloting testing, I introduced new problems and even made one of the windows less usable. I found the verbal transcripts of the sessions to be the most useful set of data. Finally, I found that predictions for the number of usability problem uncovered with increasing users were less than satisfactory, though a slight modification to one of those predictions holds promise and deserves further study. In this section, I will discuss each of these lessons.

7.7.1 Recruiting Users

Recruiting novice users for this test proved to be somewhat difficult.[32] Perhaps my expectations were unrealistic. A friend of mine had recently performed a usability test looking at an interface for remote control of an autonomous vehicle. She had had no problems recruiting users. Based upon her experience, I did not anticipate any problems either. There were, however, a few differences between her test and mine. She could include anybody with a driver's license. My novice users had to be computer science graduate students with expertise in Unix and AFS, but no knowledge of Coda. Little did I know that Coda is now covered in one of the courses required of all graduate students. Her test was highly appealing—remote control of a HMMWV! Mine was using an interface to a file system. Though I had to compromise somewhat on their background knowledge of Coda[33], I eventually found novice users.

Based upon this experience, I would have used a different procedure for recruiting novice users. Just as before, my first step would have been the same—define the "ideal novice user". I would then, however, determine the size of the population of people who fit these constraints. If the size were too small (as in my case), I would reconsider my definition, looking to see if there were an equally good population of "ideal novice users" that might lead to a larger population from which to draw participants. In my case, I could have drawn users from the ECE department (for instance). Although the graduate students in this department may have been less likely to have had Unix and AFS experience, they would also have been less likely to have knowledge of Coda. Because I did not do this, I cannot say that recruiting users would have been any easier had I done this. I do believe that redefining my ideal novice user in this way would have lead to equally valid conclusions—assuming, of course, that the users met the other constraints of my definition, namely extensive Unix and AFS experience.

7.7.2 Pilot Testing

With the exception of P1[34], the pilot users experienced conditions identical to those the novice and expert users experienced. After each pilot user completed the study, I scored the responses as I planned to do for the novice and expert users. I also addressed any problems that I observed as they were performing the test before continuing with the next pilot user. These problems included "bugs" in the Tutorial and Exercises as well as bugs in the interface.

The original design of the *Event Configuration* tab of the *Control Panel* had a single listbox that contained a listing of all events. The first pilot user suggested that this

---

[32] Recruiting expert users was significantly less difficult because they were already involved with the project and stood to benefit from the interface.

[33] Novice users were allowed to know that Coda was a file system and was a descendant of AFS, but not that it supported disconnected operation.

[34] Recall that P1 tested the prose of the Tutorial and Exercises. Her answers to the Exercises were not recorded.

window should show the indicator on which events would be notified. Based on this suggestion, I redesigned the window as shown in Chapter 4. As piloting continued, the problems with this window (see Finding 4) became obvious.

Another bug that I addressed was the alphabetizing of definitions. The interface originally alphabetized the "New..." definition in the middle of the list. No one had noticed this behavior before participant P2, probably because they had all used lower-case names. During pilot testing, I modified the interface such that "New..." always appeared first. Toward the end of the pilot testing, it became clear that I had introduced a bug (see Finding 31) related to these modifications.

After these two experiences, I took the attitude that piloting was intended to solve problems with the testing procedures and materials, not with the interface. Thus, a number of the usability problems that the novice and expert users experienced came as no surprise to me. I had observed similar problems with the pilot users. I did not address these problems prior to running the novice and expert users, however, for three reasons. First, rushing to address them could have introduced worse problems (as was the case with the *Event Configuration* tab). Second, I did not fully understand the severity or frequency of some problems. Third, the user study lab is a shared resource, one that my experimental setup was monopolizing. I could not address the more substantial problems without halting the test and leaving the lab unused.

Even if I had taken the attitude that pilot testing should be used to solve usability problems, however, it is not clear that I would have noticed some of the problems. Because the materials used during the test changed after almost every pilot user (sometimes substantially), performing the kind of analysis I did in Section 7.4 would have been extremely difficult. Without that analysis, I may not have noticed some of the trends. For instance, if you recall the problems with the ProvideHoardWalkAdvice exercise (as discussed in Section 7.5.2.1), the results of this exercise for the pilot users were extremely mixed. P2 made the same mistake that most of the novice and expert users made. P3 did not recognize the advice request as a request for advice and caused me to change the interface somewhat. P4 answered the question correctly. P5 also made the standard mistake. Looking back at this data, the writing was on the wall. Users were not making the connection. At the time though, it was not as clear especially because the "last" pilot answered the question correctly. I say P4 was the "last" pilot because the changes had come to the point of diminishing returns. I ran P5 only to make sure that nothing in the test was particularly confusing for expert users.

Had I taken this attitude from the very beginning of pilot testing, I could have run one or two fewer pilots. It would have saved a large amount of time and money. Each user required a time investment of at least 15 hours[35] and a financial investment of approximately $100 for lab time, media, and Coda shirt. Of course, had I run two fewer users, I may have also missed some findings.

---

[35] Running each user required 4-5 hours. Post-processing the raw data required approximately 3-4 hours. Transcribing the audiotapes required another 6-8 hours. Analyzing the resulting data was not done on a per-user basis so is difficult to estimate.

7.7.3 Benefit of Transcripts

While analyzing the data, it became clear to me that the transcripts provided the bulk of the evidence. This observation led me to ask two questions:

- How lop-sided was the evidence?
- What was the balance in terms of the severity of the findings?

In order to explore these questions, I analyzed the origins of the evidence for each of the 87 findings. Figure 7.32 and Figure 7.33 show this analysis visually. Figure 7.32 shows the number of findings revealed by data from particular segments of the test. It classifies those findings according to their scope and severity. Figure 7.33 shows a tree structure of the findings. The nodes of the tree, labeled T (Tutorial), E (Exercises), Q (Questionnaire), and V (Verbal Transcripts), represent the origins of evidence for the findings. The leaves of the tree each represent a single finding and are color coded to represent the severity of the finding represented. Each arc indicates that at least one piece of evidence for that finding originated from that segment of the test. These data clearly show that the transcripts provide the bulk of the evidence.

Although the Tutorial, Exercises, and Questionnaire contributed disproportionately to the more critical (level 1 and 2) findings, the transcripts doubled the number of level 1 findings identified by the test. Furthermore, they quadrupled the number of level 3 and level 4 findings! In fact, had I only looked at the transcripts and ignored all other analysis, I would have missed just a single finding (Finding 87). Although transcribing and analyzing the verbal protocol, even just the critical portions of it, was time consuming, the benefits are clear.

	Classification							
	Severity Level 1		Severity Level 2		Severity Level 3		Severity Level 4	
Segment	Global	Local	Global	Local	Global	Local	Global	Local
Tutorial	1	1	0	4	1	1	0	0
Exercises	1	4	1	2	0	2	0	1
Questionanire	0	1	2	1	3	1	0	5
Transcripts	1	10	4	6	15	24	3	23
Total	1	10	4	6	15	24	3	24

Figure 7.32: Findings Identified by Classification and Segment

This table shows the number of findings of each classification identified by each segment of the test and by the test as a whole. For example, using data from the Exercises alone, I would have identified one problem with severity level 1 that was global in its scope and four problems with severity level 1 that were local in their scope.

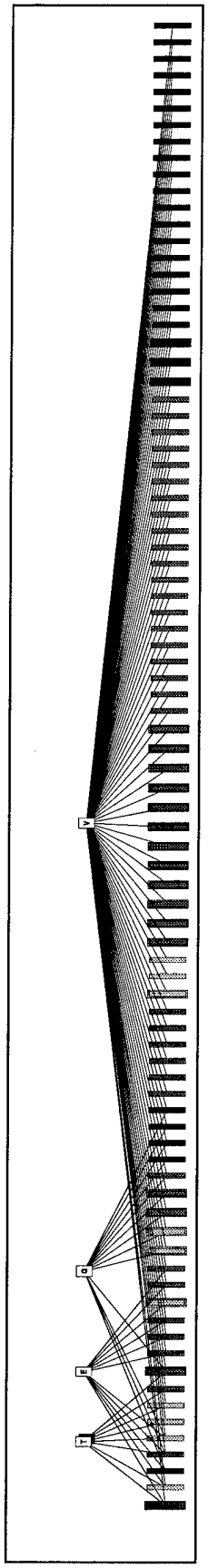


Figure 7.33: Finding Originations
This figure shows where evidence for each finding of Appendix F originated. The four top-level nodes correspond to the tutorial (T), exercises (E), questionnaire (Q), and transcripts of verbal protocol (V). Each leaf in the tree represents a single finding. The color of the leaves represents the severity rating of the given finding. Red indicates a Level 1 severity rating; yellow represents a Level 2 finding; green represents a Level 3 finding; and, blue represents a Level 4 finding. The size of the leaf represents the scope of the finding. Wide leaves correspond to global findings; narrow leaves correspond to local findings.

7.7.4 Optimal Number of Participants

Given the amount of effort needed to observe and analyze the data from each participant in a usability test, it comes as no surprise that the HCI community would like to know the optimal number of participants needed. I analyzed the evidence from each of the findings to determine which users contributed any piece of evidence, no matter how minor. This analysis is shown graphically in Figure 7.34. Each of the thirteen users is represented by a column in this matrix and each of the findings is represented by a row. Each colored box in this figure indicates that the given user contributed evidence for the given finding. More informative users appear toward the right. More frequently reported problems appear toward the top. The figure shows that the most informative user contributed to more than 50% of the findings while the least informative user contributed to fewer than 10% of the findings. The color of the box indicates the severity of the finding. This figure shows that many users found evidence for most, but not all, severe problems. In fact, six of the most severe problems were reported by just two users apiece. It also shows that most of the Level 4 problems were reported by just a few users.

Assuming that any piece of evidence would be sufficient to identify a problem, I graphed the percentage of findings uncovered as my test progressed (i.e., with my users in the order they appeared in the test). I examined the findings at each individual level, as well as the most critical problems (those with severity levels 1 or 2) and the findings as a whole. These data are shown in Figure 7.35(a-f).

An interesting characteristic of these graphs is that all but one of them is best approximated by a log-based function. The surprise is that one of them, the severity level 4 findings, is best approximated by a linear function. Evidently each additional user saw different enhancements. I hypothesize that this curve might also be logarithmic in the limit; in fact, you can see what might be a flattening trend near the end.

With just three users, I would have uncovered more than 50% of the problems at severity levels 1 and 3, but only 30% of the problems at the other levels. To uncover 80% of the problems, I would have had to run approximately six users. It is clear that beyond about six users I quickly reached the point of diminishing returns.

My next goal was to determine what, if any, effect the ordering of participants had on these results. To make this determination, I calculated the number of findings uncovered by every subset of my users. I then calculated the average number of findings reported by subsets of a given size. For instance, a set of thirteen users contains 78 subsets of size two. For each of these subsets, I calculated the number of findings they uncovered. I then averaged these values over all 78 subsets. By performing this same calculation for the other subset sizes, I could determine the average number of findings uncovered with increasing numbers of participants. On average, I find that six users were required to uncover 80% or more of the findings, but that only three users were required to uncover 50% or more of the findings.

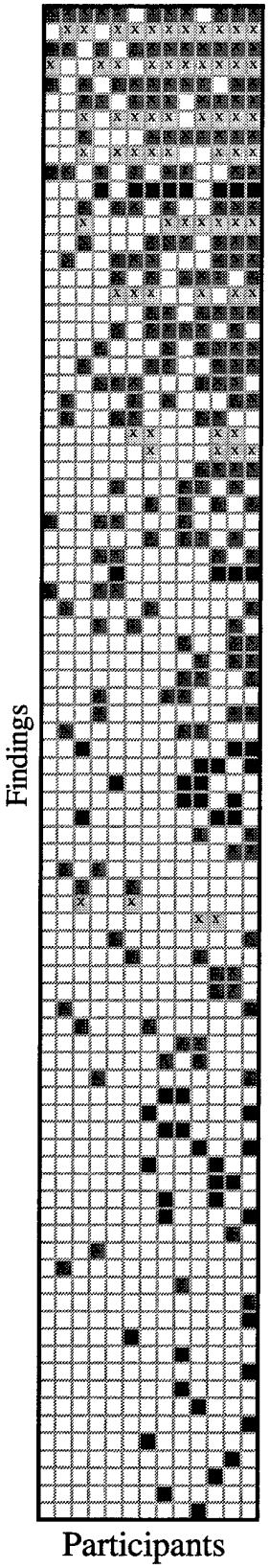


Figure 7.34: Frequency of Reporting
This matrix indicates which users contributed evidence to which findings. Each box indicates that the given user contributed evidence to the given finding. The color of the box indicates the severity of the finding: red represents level 1 problems, yellow represents level 2, green represents level 3, and blue represents level 4. The matrix has been sorted so that more productive users appear in the rightmost columns and more frequently reported problems appear towards the top. The order of users from left to right is: P2, S1, N4, P3, P5, C1, N1, E2, P4, T1, N2, E1, E3.

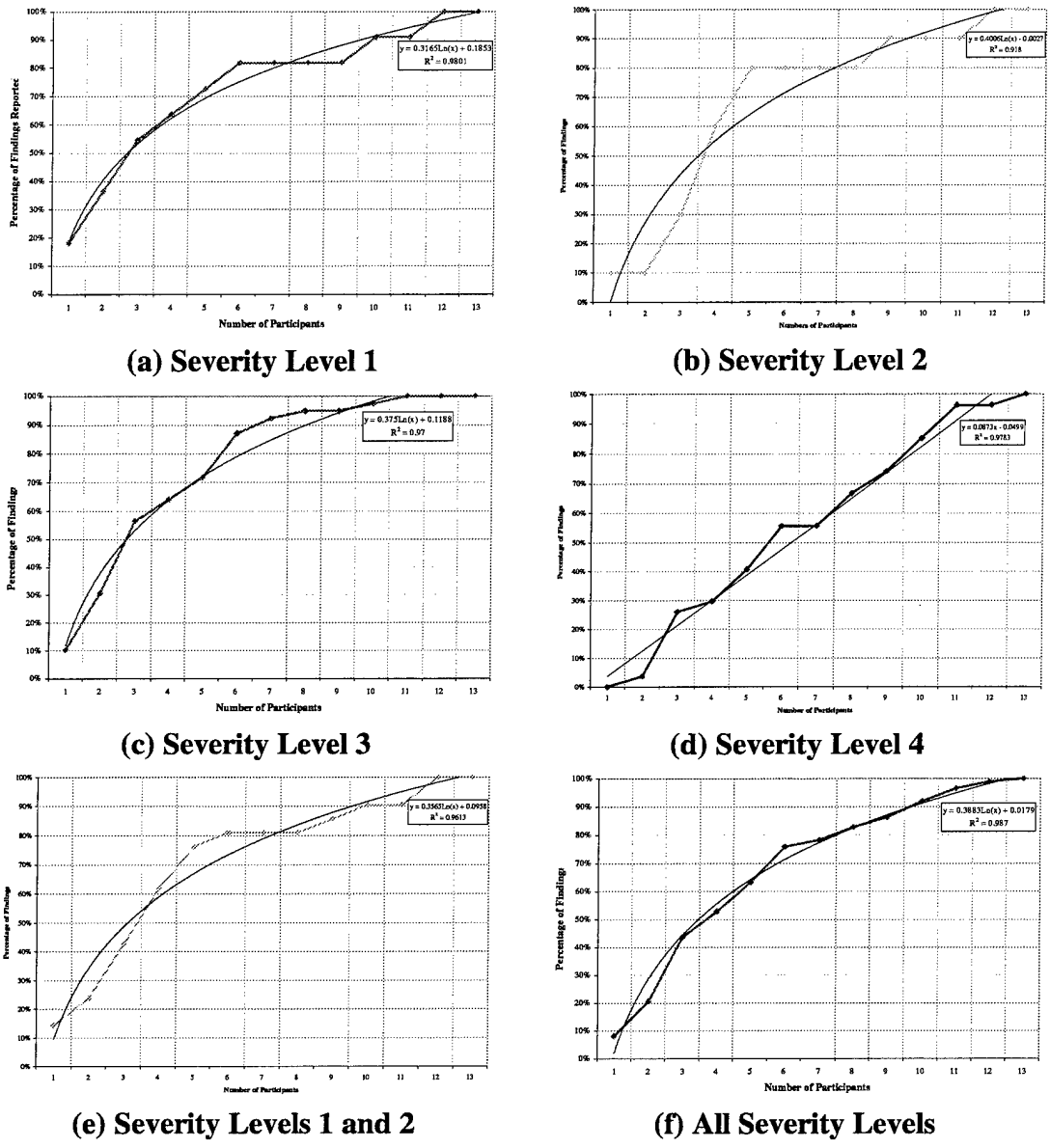


Figure 7.35: Number Users Required to Cover Most Severe Findings

This figure shows the number of participants necessary to cover the usability findings. In Views (a)-(d), I show just the findings with severity levels 1-4 respectively. In View (e), I show the most critical findings, those with a severity level of 1 or 2. In View (f), I show all findings. In each view, I show the closest function that approximates the data. All but one graph is most closely approximated by a log function. View (d), however, is more closely approximated by a linear function.

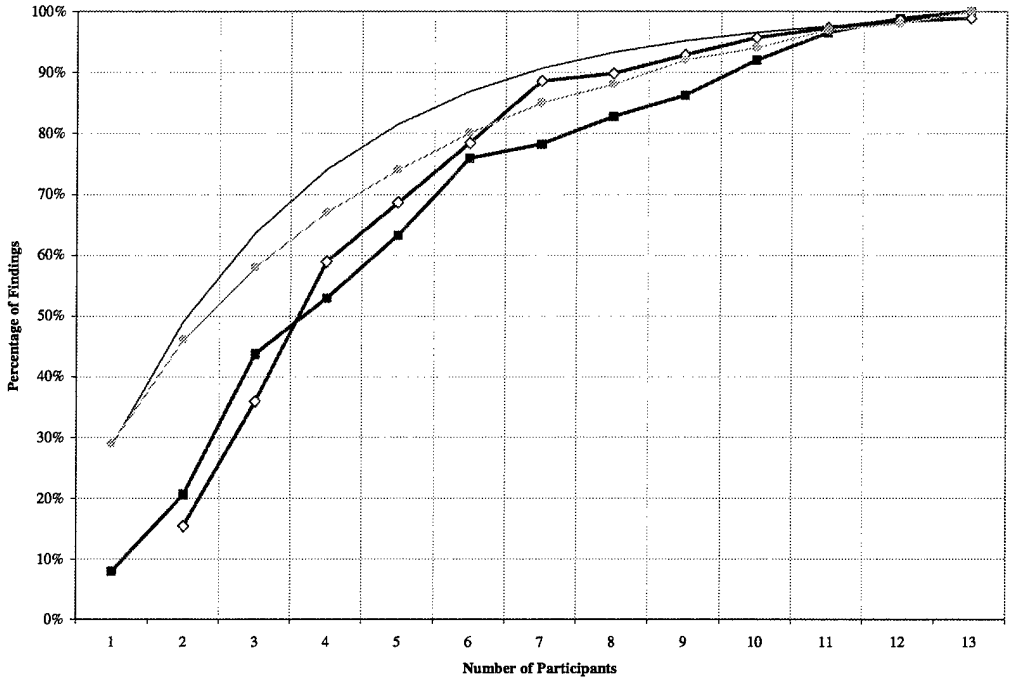


Figure 7.36: Proportion of Findings Found and Expected

This graph shows the actual proportion of findings found by the thirteen users in my test in chronological order as well as three predictions for those proportions. The line marked with filled squares represents the actual values. The curve (without markers) represents the baseline prediction using the average probability of a user uncovering a finding as calculated over all thirteen users. The line marked with open diamonds represents a modification of the baseline prediction using a running average. The line marked with filled circles represents the average number of findings uncovered by the given number of participants; it removes any effect caused by the chronological ordering of users.

The final point I wanted to explore was to examine the success of predicting the number of users required to uncover a certain percentage of findings. Virzi points out that probability theory indicates that the proportion of problems one would expect a given number of users to identify is given by the formula $1-(1-p)^n$, where p is the probability of detecting a problem (nearly 29% for this test) and n is the number of participants (13 for this test) [54]. I compared the prediction based upon this formula, which I refer to as the *baseline prediction*, with the data I collected; the results are shown in Figure 7.36.

This figure actually shows four curves. The line with filled square markers represents the actual number of findings uncovered as the test progressed (users are ordered chronologically). The curve with filled circle markers represents the average number of findings uncovered by subsets of various numbers of participants (as discussed above). The curve without markers represents the predicted values based upon the average probability of a user finding evidence supporting a finding (just under 29% for this test). The curve with open diamond markers represents a modified prediction based upon a running total average.

From these graphs, it is clear that the baseline prediction models the shape of the curve quite well, but that it overestimates the number of findings uncovered more than expected. In particular, it overestimated the expected percentage of findings by 20-30% until the number of participants reached six and by approximately 10% until the number of participants reached ten. It is also clear that the subset average predicts somewhat better, but it overestimates the number of findings uncovered by more than 20% for small numbers of participants. Neither of these models is particularly satisfying because both are very inaccurate for small values of participants, which happens to coincide with the area of the curve most interesting to experimenters trying to determine the necessity of running another experiment.

Because the baseline prediction depends heavily on the accuracy of p, I reanalyzed the data using a running average. For each point on the line with open diamond markers, I used the average probability "to date". So, to calculate the prediction for the fifth user, I averaged the probabilities collected from the first four users. The beauty of this prediction function is that it uses only data that an experimenter would have available to determine whether or not to continue running the experiment. Although I expected the running average prediction to be less accurate than the baseline prediction (because it had fewer data points to consider and was likely to be less accurate than the overall average), it actually proved to be a more accurate predictor for these data. For this reason and because it represents a more practical method for use in the field, it warrants further validation. One area of particular concern is its accuracy "on average" over multiple studies, ideally on the same studies included in [38, 54, 55].

It is interesting to note that the data from this test differ substantially from the baseline prediction, a result that apparently contradicts Virzi. Virzi claimed "In practice, this difference has been small, and neglecting it will lead to predictions within a few percentage points of those obtained" [54]. In my experience, Virzi's prediction was off by 20-30% in precisely the range of participants that would be most critical to evaluators working in the field (six or fewer users). At least part of the difference between Virzi's results and mine is simply a matter of statistics. Virzi's evaluation of the accuracy of his predictor function is based upon a Monte Carlo Procedure, examining samples of the appropriate size from the set of all users for each prediction [55]. My evaluation is based upon the chronological ordering of the users in my test. His evaluation shows us that, *on average*, the predictor function is accurate to within a few percentage points. My data show that, *in practice*, it can be quite inaccurate, particularly for small values of n. In fact, in comparing Virzi's prediction with my subset average prediction, I find that his prediction comes within approximately 7% of mine. Still, for evaluators in the field, it appears that these predictions should be used as a rough guide, and that analyzing the data "on the fly" to confirm that the test has indeed reached the point of diminishing returns continues to be the prudent course of action.

7.8 Summary

The goals of this usability test include two facets. The first was to identify areas in which the interface and the tutorial could be improved. The second was to observe a handful of people actually using the interface, looking at some specific issues regarding the ease with which these users learned the interface, their ability to understand the feedback provided by the interface, and their ability to operate the interface.

The usability test identified specific areas in which both the tutorial and interface could be improved. These findings included concerns regarding the approach, bugs in the interface, usability problems, omissions in the interface, confusing behaviors, and future enhancements. One of these findings suggests an extension to the interface that could further improve the user's ability to define tasks.

Regarding the specific issues of concern, this usability test suggests that the interface met each of its objectives. These users were able to learn how to use the interface in approximately one hour. They were able to understand problems indicated by the interface. They were able to determine what actions might resolve those problems. They were largely able to define and hoard tasks. They were able to provide advice to the system. And, finally, they were able to configure event notifications.

Perhaps the most important result of this test, however, is the fact that these novice Coda users performed almost as well as these expert Coda users. This result is extremely surprising. Although I certainly expected the interface to help novice users, I did not anticipate the extent to which it would do so. Historically, novice Coda users experience a long and arduous learning curve. A few of them have even given up before reaching the top. The interface apparently reduces this learning curve substantially, making the climb less strenuous.

Many interesting questions remain unanswered. These include questions related to the performance of the system, questions related to the effectiveness of the heuristics, as well as questions related to users' ability to operate and understand the system in the field. Although these questions are all interesting, discovering the answers to them is beyond the scope of this thesis and must be left for future work.

8 Related Work

This chapter presents work related to this thesis. It is divided into three sections. The first section presents systems that also address what has become known as the hoarding problem. The second section discusses computer interfaces that use a dashboard metaphor. The third section presents work on open implementation because this interface can be considered an example of such a system.

8.1 Hoarding

A number of other systems address the hoarding problem. In addition to Coda, these systems include Little Work, FACE, Briefcase, Transparent Analytical Spying, Seer, and Laputa. The last of these systems, Laputa, describes an application-specific solution to the hoarding problem. The remaining systems each describe their approach for addressing the general hoarding problem.

To aid in understanding these different systems, Figure 8.1 presents a taxonomy for these different approaches. The taxonomy includes a number of different categories.

Granularity of objects: Describes the granularity with which each system specifies the files and directories that need to be hoarded in preparation for disconnected operation

Prioritization: Specifies whether users can prioritize hoarded objects and, if so, how many distinct priorities they can use

Identification of objects: Describes how each system determines which objects need to be hoarded in preparation for disconnected operation

Distinguish application data from user data: Specifies whether the system makes a distinction between application and user data

User Interface: Describes the interface to the system that is presented to the user

Characteristic	Mechanisms						
	Little Work	**FACE**	**Briefcase**	**Coda (original)**	**TAS**	**Seer**	**Coda (revised)**
Granularity of Objects							
File-based	✓	✓	✓		(✓)		✓
File-based + Meta				✓			✓
Command-based				(✓)	✓		✓
Task-based					(✓)	✓	✓
Prioritization of Objects	None	None	None	1000	None	None	88
Identification of Objects							
User Responsibility		✓	✓	✓	✓		✓
History Horizon	LRU			LRU			Extended
Spying				Manual	Automatic		Automatic
Semantic Distance						✓	
Distinguish (application vs. user)							
Doesn't	✓	✓	✓	✓		✓	
User Responsibility				(✓)			
Heuristics					✓		✓
User Interface	None	Command	Graphical	Command	Graphical	Command	Graphical
Extent		Basic	Basic	Basic	Basic	Minimal	Extensive
Feedback		None	Minimal	None	None	None	Extensive

Figure 8.1: Taxonomy of Caching for Availability Mechanisms

This table summarizes the characteristics of different approaches to the hoarding problem. The original Coda system refers to that described by Kistler [23]; the revised Coda system refers to that described in this thesis. The taxonomy begins by examining the granularity with which objects must be specified in each system. Some systems require the user to specify individual files and directories, sometimes augmented with meta-information (e.g. similar to that described in Section 2.3.2). Other systems allow the user to specify programs while still others offer support for tasks. It continues by specifying whether the system allows objects to be prioritized, and if so, how many distinct priorities are supported. It then examines how objects are identified for hoarding: by the user, by a history-based mechanism (perhaps LRU or an extended history horizon), by spying on file references, or by semantic distance. It then indicates whether the system distinguishes between application and user data, either internally using heuristics or externally with the help of the user. Finally, it describes the user interface provided by the system, including the type of interface (command-line or graphical), the extent of the interface (*minimal* allows the user to warn the system of an impending disconnection; *basic* supports user-assisted hoarding; *extensive* supports further interaction), and the amount of feedback provided. Checkmarks shown in parentheses indicate that users can do this if they choose, but that the system does not explicitly support or require this activity.

Let me make this taxonomy concrete by reviewing where the original Coda system, as described in Chapter 2, fits into this scheme. The granularity with which Coda specifies objects to be hoarded is at the level of file system objects, but it allows users to provide meta-information for directories. The original Coda system supported up to 1000 distinct priorities for hoarded objects. The user was responsible for hoarding objects, although, because the hoarded data was stored directly in the client's cache, the system also cached objects that had been accessed recently. The system did not distinguish between user and application data, except insofar as such a distinction was made by users within their hoard profiles. It presented a basic command-line interface to the user with no feedback about the state of the system.

8.1.1 Little Work

Little Work [14, 15] represents the baseline solution to the hoarding problem. This system offers absolutely no explicit support for users of disconnected operation. Because the system uses an LRU caching algorithm, users simply prime the cache before disconnecting by running the programs they expect to use during the disconnected session. Users evidently do this using their shell as the system provides no user interface. The authors have also added a *fetch-only* mode of operation that allows them to fetch any missing objects while connected over a weak link.

8.1.2 FACE

FACE [3] is the first system that considered exploiting user assistance to increase data availability. They implement a secondary cache, which they call a *stash*. While using a disconnected client, users access data out of this stash. In contrast, Coda uses a unified cache for both performance and availability.

Alonso and his colleagues suggest a number of techniques to identify files that need to be stashed. The first of these is a set of permanently stashed files, a list of which is kept in a well-known location such as `~/.stashrc`. Their second suggestion requires writing a tool that parses Makefiles to identify files that a given Makefile requires to run. Their third suggestion is approximately equivalent to Coda's spy tool. Their fourth suggestion requires instrumenting applications.

Their prototype appears to have implemented just two commands supporting user-assisted hoarding, neither of which was listed in their original suggestions. The first of these adds a file to the stash; the second removes a file from the stash. The authors offer no indications as to which of their suggested interfaces were implemented, nor do they offer any indication as to the success of user assistance in improving availability.

8.1.3 Briefcase

Briefcase [41, 50], a feature of Microsoft's Windows95, allows users to put copies of their documents into an electronic briefcase. In essence, Briefcase is simply a user-friendly term for what is, effectively, a cache. Users explicitly place documents in this cache before disconnecting. During disconnected operation, users modify cached documents. Upon reconnection, the user can request that the cache be synchronized with the local copy of the data. During synchronization, the system uses the timestamp associated with both the primary copy of the file and the cached copy of the file to determine which one to keep (the most recently updated file wins). Users can override this choice manually.

Briefcase assumes users are synchronizing a laptop with a desktop computer. It assumes that all necessary applications are resident on both machines. Thus, the problem is one of synchronizing user data only. It offers no support for indirect references to files. For instance, suppose a Word document includes a link to an Excel spreadsheet. If the user places that Word document into her Briefcase, the system is not smart enough to also include the Excel spreadsheet. To make matters worse, updates to the Excel spreadsheet are not reflected as updates to the Word document that includes a link to them. Thus, Briefcase is not smart enough to realize the primary copy of the Word document changed. On a simple test, I managed to get my Word document to become inconsistent with my Excel spreadsheet even after repeated attempts to force Word to update the imbedded object. My use of Briefcase produced not a single error or warning message about the actions I had taken. Briefcase is a simple-minded solution that addresses only a limited form of the hoarding problem, and is not yet ready for naïve users.

8.1.4 Transparent Analytical Spying

Transparent Analytical Spying[36] [52] applies prefetching research [51] to the problem of hoarding. This work effectively extends Coda's spy utility by providing a simple user interface to control spying, as well as a hoard selection interface to control what gets cached prior to disconnecting.

This system works by observing file accesses made by executing programs. For each program, it builds a set of access trees. Each access tree corresponds to a single execution of a given program (and its descendant processes). Because these access trees are maintained over the long term, they must be coalesced to save space.

Automatic program hoarding, described in Section 6.1 is also motivated by Tait and Duchamp's prefetching research. The primary difference between my subsystem and their Transparent Analytical Spying work is in their use of access trees. The access trees, which were critical to the success of the original prefetching application, offer no advantage to their system or my subsystem presently. Such trees could, in theory, help to prune the set

---

[36] I call this system *Transparent Analytical Spying* because that is the closest the authors come to naming this system. Kuenning [26] calls this system the *Spy Utility*. I chose not to use this term because of the conflict with Kistler's *spy* utility [22].

of file objects hoarded, but would require further information to do so without significant user input. For this reason, I chose to simplify the design by maintaining simple access lists.

This system also introduces the notion of *generalized bookends*. The user specifies when the system should start and stop spying. In essence, this system offers a graphical interface to a more sophisticated version of the spy tool (see Section 2.3.3) available in the original Coda system. At that time, she can specify a name, called a bookend, for the set of references made during the spy session. These bookends allow the system to group top-level activities. Bookends serve a purpose similar to that of tasks in my system.

Before disconnecting, the user requests that the system hoard her data. The system then displays a hoard selection window that allows the user to select the bookends that should be hoarded. In the same window, the user can select individual programs and specify additional files and directories.

Transparent Analytical Spying is the first system to present a graphical interface to the user of disconnected operation, though the operations are very basic. The interface provides no feedback to the user regarding the availability of the selected data or anything else.

The level of user assistance varies depending upon the interest of the user. The authors' goal as far as user assistance is concerned seems to be that of allowing optional assistance. Uninterested users would be able to accept the defaults and be reasonably happy. Interested users would be able to customize the hoarding selections and improve their disconnected availability. The authors have published no studies showing whether they met this goal, or examining the usability of their graphical interface.

8.1.5 Seer

Seer [25, 26, 27] applies multidimensional clustering to the problem of hoarding. This is the first system that requires almost no intervention by the user. The results are quite impressive.

This system identifies related files and directories by exploiting temporal locality of reference. The authors observe that related files and directories are generally accessed close together in time. They define the *semantic distance* between two file system objects, A and B, based on the number of opens that occur between the open of A and that of B. Using the semantic distance between files, Seer clusters related files into groups.

Before disconnecting from the network, the user must warn the system—this warning is the only user interaction required by Seer. Upon receiving this warning, the system assigns priorities to objects according to the clusters in which they appear. All elements of a cluster are assigned the same hoard priority, using an algorithm described in [26].

Seer offers two unique results. The first is its ability to identify the user's working set with almost no assistance from the user. The second is its ability to minimize the amount

of cached data associated with each cluster. Together, these results offer significant utility to users of disconnected operation.

Seer's weakness is its inability to build clusters that are meaningful to its users. In fact, the authors explicitly list this characteristic as a disappointment:

> "The clusters produced by Seer often have contents that are surprising to us, either by including apparently unrelated files or by separating a single project into a few clusters rather than the single grouping that would correctly represent it." [27]

This weakness is unfortunate because it makes incorporating these results into the CodaConsole interface difficult. Ideas for doing so, however, will be presented in Section 9.2.2.4.

8.1.6 Laputa

The approach taken by Laputa (described in [49]) can only be described as application-specific hoarding. The goal of this system is to provide disconnected operation to software developers. The authors exploit process information readily available in their software development environment, Marvel [18], to determine what files should be hoarded to support a given activity specified by the user. The status of their implementation is unclear and their experience with the system has not been published to date. Because this system is specific to a particular application, I do not include it in my general taxonomy.

8.2 Dashboard Metaphor

One would expect the dashboard metaphor to be a common metaphor for exposing failures to users. Because of the popularity of automobiles, it has been widely deployed and accepted by consumers, at least in the United States. To date, I have found surprisingly few systems that exploit this metaphor for use with untrained computer users.

One particularly good example however, called *Net.Medic* [2], monitors Internet performance for naïve Internet users. The system claims to monitor network performance, isolate the bottleneck, diagnose the cause, and even correct certain problems. It presents information to naïve users in the form of a dashboard, using the familiar red, yellow, green color scheme.

The Net.Medic dashboard follows the dashboard metaphor more closely than does the CodaConsole. The gauges presented by Net.Medic closely resemble gauges of an automobile. The interface presents both analog gauges for network speed and throughput and digital gauges for retrieval time and connection time. Net.Medic does not concern itself with continuous feedback to users regarding network performance. Users must explicitly examine the monitor. Because of this design assumption, their system does not worry about minimizing screen real estate during normal operation.

Two other examples, Dashboard95 (a product of Starfish Software) and Front Panel [30], apparently provide interfaces that help users maneuver through a crowded desktop. They give quick access to common tasks, such as electronic mail and calendars. They keep important information at hand, such as the status of inboxes and the current date and time. Front Panel, at least, allows users to customize a set of *bins*. These bins contain the user's current work tasks. The main difference between the dashboard metaphors used by Dashboard95 and Front Panel, and the metaphor used by Net.Medic and CodaConsole is the focus. The goal of the former systems is to present quick, easy shortcuts to common tasks (the function of the lighting and environmental controls on and to the right of the steering wheel). The goal of the latter systems is to present error conditions to users (the function of the display panel behind the steering wheel).

A Web search uncovered one additional use of the dashboard metaphor, a dashboard interface builder [1]. This system allows developers to quickly build dashboard interfaces to specialized astronomy equipment. The system has apparently been used at both the Keck Observatory and UCO/Lick Observatory.

The dashboard metaphor will increase in popularity as people recognize its benefits. The metaphor has been introduced in a number of specialized interfaces. As people take notice of these specialized interfaces, they will begin incorporating the metaphor into more general computer systems. Office97 already includes extensive control options similar to those of Dashboard95 and Front Panel. It also uses a small number of indicator lights to provide users with feedback about the status of print jobs and of battery power.

8.3 Open Implementation

Computer science undergraduates are taught in Software Engineering 101 (or perhaps even earlier) to use a "black box abstraction". They learn to design an interface to their libraries and hide the implementation behind that interface. This technique offers substantial benefits in reducing the complexity of the resulting system. Unfortunately, this reduction comes at a high cost.

When the designer begins implementing the "black box", certain design choices must be made. Making these decisions requires the implementer to make assumptions about the client of the interface. Kiczales provides a nice example[20]. He considers a windowing system and the impact of certain implementation decisions on one particular application, the display portion of a spreadsheet. The basic problem is that the windowing system's implementation assumed only 25-50 heavy-weight windows (e.g. an xterm) would be used—not a thousand or more light-weight windows (e.g. a cell of a spreadsheet). Once that assumption had been hard-coded into the "black box", the spreadsheet implementer could no longer use this windowing system.

Kiczales argues:

> It is impossible to hide all implementation issues behind a module interface. Some of these issues are crucial implementation-strategy decisions that will inevitably bias the

> performance of the resulting implementation. Module implementations must somehow be opened up to allow clients control over these issues as well [20].

He continues by advocating an approach that separates the primary interface to the functionality of the module from a secondary interface that allows the client programmer to tune the underlying implementation of that module for a particular application if necessary. He states three goals for an open implementation. The client programmer should be able to:

- Use the module's primary functionality alone when the default implementation is adequate;
- Control the module's implementation strategy when necessary; and
- Deal separately with functionality and implementation-strategy decisions.

A good example of an open implementation is Kaashoek's work on the Exokernel [17]. In this system, the designers point out that operating systems define the interface between applications and hardware. They also point out the unfortunate consequences of hard-coding the implementation of operating system abstractions, including the fact that applications cannot take advantage of domain-specific optimizations. The goal of their kernel was to give applications control over how machine resources are used while protecting applications from one another. They provide a low-level interface to the hardware, and build operating system abstractions on top of it in libraries that are linked into the application. Client programmers can pick those abstractions most appropriate to their domain or they can start from scratch if no off-the-shelf libraries will work.

My work falls into the category of an open implementation of the cache for a distributed, highly available file system. My work differs from these and similar approaches in that the client for whom I am opening the implementation is actually the end user. Although this makes certain issues critically important (specifically, the user interface), the problems and goals of my system are closely related to those described above. In fact, I can restate Kiczales's argument as it applies to a distributed file system for which network resources are scarce or expensive as follows:

> It is impossible to hide all resource usage decisions behind a "black box" file system. Some of these crucial decisions will inevitably bias the performance of the resulting system. Mobile file sytems must somehow be opened up to allow users control over these decisions.

This thesis describes a way to give users control over these decisions, without sacrificing usability. The goals that Kiczales set out also apply to my system. Specifically, the user should be able to:

- Use the system "as is" when the default policies are adequate;
- Control the policy decisions when necessary; and
- Deal with the file system and the policy decisions in largely separate ways.

One additional point that he does not make explicitly, but that he implies by the secondary (and optional) interface is that the system's policy mechanism should be as unobtrusive as possible, remaining in the background when not in active use.

8.4 Summary

This chapter has described work related to that described in this thesis. Many people have looked at the hoarding problem, though no one has looked at ways to make caching translucent to users. Surprisingly few systems exploit the dashboard metaphor to expose failures to users, though this metaphor is becoming more common. Finally, this work represents an open implementation at a level higher than that traditionally considered.

9 Conclusion

Systems that exploit caching for availability will become more widespread in the years to come. Mobile computers, both laptops and personal digital assistants, require the ability to operate while disconnected or weakly connected. As users of these systems tire of managing their "cached" data by hand, more sophisticated support will be added. Our experience in Coda will help designers of these systems as they begin to fully support mobile users.

In this dissertation, I describe an interface that supports users of such a system. The interface presents the user with indicator lights, a metaphor borrowed from the dashboard of an automobile. These indicator lights keep users informed about the status of the system, but also allow them to request further information about a problem. The system allows users the option of controlling the use of network and cache resources. It also takes care to reduce the burden that these features impose on users.

In this final chapter, I first discuss the contributions of my thesis. I then outline areas for future exploration including ideas that need further validation, ideas for improving the interface, and ideas for extending the translucence. I conclude by completing the fictional story begun in the introductory chapter and with a few final remarks.

9.1 Contributions

The overall contribution of this work is the recognition and demonstration that translucent caching is a feasible and effective approach to making systems that cache for availability usable. I demonstrate this thesis with the design, implementation, and evaluation of a graphical interface that makes caching translucent to users of the Coda file system. At the next level of detail, my thesis makes the following contributions:

- *Introduction of the concept of translucent caching and recognition of its critical role in systems that exploit caching for availability.*

  Unlike caching for performance, caching for availability cannot remain transparent to users during periods of poor network connectivity. When a cache miss occurs while a client is operating disconnected, users suddenly notice that their files are not stored locally. Similarly, when a miss occurs while a client is connected via a modem, users notice substantial delays in accessing large files. This lack of transparency causes usability problems. The approach I take to address this problem is to systematically expose internal details of

caching in a controlled way and to give the user the opportunity to influence how resources are used.

- *Design for a translucent caching interface.*

  This graphical interface makes caching translucent to users of the Coda file system. The system balances the tension caused by the need to preserve the Unix philosophy with the need to address the usability concerns discussed above. A key feature of Unix is the ability of programs to run unattended. Preserving this feature seems to preclude interacting with the user. My interface balances these conflicting goals: attentive users are given opportunities to control the system, yet programs running unattended (and inattentive users) are not significantly impeded.

- *Identification of techniques for reducing the burden of translucent caching.*

  Although translucent caching can improve usability, it also runs the risk of reducing usability. If a system using translucent caching is not careful, it could focus user attention on managing the cache and network, and detract from users' ability to get useful work done. This thesis examines the potential costs involved and introduces three techniques to reduce those costs. The first of these techniques, automatic program hoarding, reduces the cost of advising the system regarding what to cache. It automatically tracks references made by programs, differentiating between program data and user data to allow users to share program definitions between tasks. The second of these techniques reduces the cost of managing task definitions by identifying objects that the user may wish to add to or remove from a task definition. The third technique, modeling user patience, predicts whether users will be willing to wait for a file based upon that file's importance and the expected fetch time. The system then suppresses advice requests when the object can be fetched within the user's patience threshold. These techniques strive to reduce the burden that caching for availability places on users.

- *Implementation of a translucent caching interface and definition of a Translucent Caching API.*

  The interface, which was implemented within the context of the Coda file system, demonstrates the feasibility of these ideas. The system has recently been deployed to the Coda user community and we look forward to learning its benefits and limitations under real world conditions in the months ahead. The implementation has benefits beyond Coda, however. Because it was implemented external to Coda's cache manager, other highly available file systems could exploit the interface with a modest amount of effort.[37]

---

[37] The modifications necessary to implement this API within Coda's cache manager require approximately 2500 lines of code, and represent only a 3% increase to the overall size of Venus. Modifications to the graphical interface may also be required to customize the indicators to a different file system, but the basic events and ideas should be similar.

- *Development of an experiment to evaluate the effectiveness of the translucent caching interface in improving usability.*

  To evaluate the CodaConsole interface, I developed an experiment that examined questions of usability specific to translucent caching. The experiment was based upon a general evaluation technique common in HCI research, usability testing. As part of the experiment, I developed an on-line tutorial and exercises with which to teach participants about the interface and then observe them in action. Although the tutorial is obviously specific to the CodaConsole, the exercises could be employed, perhaps with minor modifications, to evaluate alternative approaches to the problem. Further, this work represents the first known application of HCI evaluation techniques to distributed systems research.

- *Results of a usability test performed on the CodaConsole interface.*

  The results of this test were striking: novice Coda users performed almost as well as expert Coda users! Users were presented with activities similar to those typically required of Coda users. They were asked to define and hoard tasks. They also experienced events that occur while using Coda, including disconnection, weak connection, cache misses, and authorization expiry. With the interface, users were able to understand what had happened and explain what actions they could take to resolve the various situations. This test validated my approach.

- *Feedback to the HCI community on my experience with usability testing.*

  My experience during the usability test lead to two observations of use to the HCI community. The first reveals the importance of verbal protocol to the process. During the analysis of the usability test, I realized that the transcripts of the verbal protocol contributed disproportionately to the usability findings. In fact, my analysis of the contribution of each piece of evidence to the usability findings shows that the verbal protocol was indeed the most important source of data. This result validates the use of verbal protocols in usability testing and explains the popularity of this technique with HCI evaluators. My second observation reveals the inadequacy of current techniques for estimating the number of users required during a test. The generally accepted rule for predicting how many users are required to uncover a certain percentage of findings did not work well for my usability test. In this test, the statistical prediction underestimated the number of users needed by 20-30% in precisely the range most critical to evaluators working in the field (e.g. six and fewer participants). This result contradicts previous experience with this prediction technique for which researchers found the model accurate to within a few percentage points. The thesis suggests an alternative predictor that worked well for my experimental data, but leaves open the question of its validity across other experiments.

9.2 Future Work

The work presented in this thesis represents an initial step toward translucent caching. The interface shows the feasibility of translucent caching and demonstrates improved usability. Additional areas of the system could, and should, be exposed to the user. As my work progressed, I also identified areas that deserve further attention, but that are beyond the scope of this thesis. In this section, I highlight these areas for future work.

9.2.1 Validating Aspects of the Interface

A number of important issues remain unvalidated. One such issue is our ability to predict the user's patience threshold. Another is the effectiveness of the heuristics used to offer users advice. A third is the effectiveness of my translation between user task priorities and Venus hoard priorities. Furthermore, although my thesis suggests that the interface will offer substantial benefits to users, the interface has not been validated under real-world conditions. In this section, I explain why each of these validations is necessary and what questions each validation should address.

9.2.1.1 Heuristics for Offering Advice

The technique for offering users advice regarding the content of the task definitions, presented in Section 6.2, depends upon the success of the heuristics. These heuristics have not, as yet, been validated. Thus, an interesting experiment would be to measure the effectiveness of these heuristics. In particular, such an experiment should address the following two questions for each heuristic:

- How much information does the heuristic identify?
- How frequently does the user modify her task definitions to reflect the information provided?

The data necessary to answer the first question would be trivial to collect. The system need only be instrumented to record the information as well as the parameter values used by each heuristic, and send it to a data collection site. The data necessary to answer the second question, however, presents a small problem. The difficulty here is that currently it would be nontrivial to determine whether or not the user actually used the information (though see Section 9.2.2.3 below for one way to address this problem).

Although I have no doubt that these heuristics will identify files that users have forgotten to include in a task definition or that users forgot they included in a task definition, I am concerned about the burden this advice will place on users. If the heuristics identify too many false positives, users are likely to ignore the advice. By collecting usage statistics regarding the advice we offer users, we can answer questions about how much information these heuristics identify in practice and how frequently that information influences users' task definitions. Once we know how well these simple heuristics perform

in practice, we may find that we need to explore more complex approaches to avoid false positives or to avoid presenting too much information to the user.

9.2.1.2 User Patience Model

The model of user patience presented in Section 6.3 depends upon the hypothesis that a user's patience can be predicted with adequate accuracy. To date, this hypothesis has not been verified. An interesting experiment would be to collect the data necessary to support or disprove this hypothesis and, if appropriate, to then determine the shape of the curve.

Because individual differences could have an enormous effect on the resulting data, the data from individual users would need to be analyzed separately. Further, each user would need to experience a large number of weakly connected cache misses covering a wide range of priorities in order to provide a sufficiently clear picture of any potential model. Thus, this project requires a large number of users providing data over a lengthy period of time.

Assume for a moment that such a model has been verified and quantified. That model could offer benefits well beyond the realm of Coda. For example, suppose a user is searching for information on the Web, perhaps in Yahoo or in a database of newspaper articles. If the search engine can reliably predict the quality of a match, the client could use that information to predict whether or not the user will be willing to wait for a particular page. Although the use of such information would likely be different in the Web domain, that information has the potential to reduce the burdensome delays we have all experienced while using the Web.

9.2.1.3 Task to Hoard Priority Translation

Alternative translations between task priorities meaningful to users and hoard priorities meaningful to Venus have not been compared experimentally. The current system uses the Fibonacci sequence to spread higher priority tasks further apart in the hoard priority space (see Section 6.3.2.1). The translation, although simple, has not been tested experimentally. An interesting experiment, then, would be to examine how different translations work in practice. Such an experiment should address the following questions:

- How many distinct tasks do users need? Similarly, how many well-separated tasks do users need?
- Should the translation be linear or nonlinear?
- How far apart in the hoard priority space must the highest priority tasks appear to allow Venus to make cache replacement decisions intuitive to users?

This experiment is intimately tied to the prioritized cache management policy used by Venus. Thus, a serious complication of this examination will be to differentiate between these two issues. Because these two issues are confounded, it may be appropriate to explore alternative cache management policies simultaneously.

9.2.1.4 Real-World Conditions

The user test suggests that the interface will help users substantially, particularly new Coda users. However, these results have not been confirmed by actual usage experience. An interesting study would reexamine the questions presented in Section 7.2 under real-world conditions. Designing such an experiment requires careful thought to find ways to collect data that will actually answer these questions.

For example, one measure of a user's success in defining tasks is the frequency with which she experiences cache misses. Unfortunately, the system views one cache miss as being equivalent to the next, whereas users often do not. The severity of a cache miss is highly dependent upon the user's current goals. One method of addressing this complication is to request advice from the user. Figure 4.7 shows an example of how an experimenter might collect severity data. In this query, the system has requested information regarding the severity of a disconnected cache miss. With such data, the experimenter could estimate the effectiveness of a user's task definitions, looking at the frequency of critical cache misses as well as less severe ones.

9.2.2 Improving the Interface

The results of the user study identified many ways in which we can improve the usability of the interface. These were summarized in Section 7.6 and described in detail in Appendix F. In this section however, I present further suggestions for improving the interface. These suggestions are not a result of the usability test but simply of my reflections on the interface while writing this thesis.

9.2.2.1 Parameters

One aspect of Venus, which the interface does not expose to the user, is the many parameters that Venus depends upon. Examples of these parameters include those controlling the user patience model, the prioritized cache management policy, the weak connectivity boundary, and the aging of records before reintegration. The parameters to Venus measure in the dozens. Some subset of these parameters could reasonably be exposed to the user with proper explanations of their purpose. We have never conducted experiments to determine the "best" parameter settings; instead, we have relied on reasonable performance in practice. Exposing these parameters to interested users will allow them to experiment, at least informally. They may discover settings that work better.

The proper place to expose these parameters is in a new tab of the *Control Panel*. On this tab, I would place another set of tabs that group related parameters. For instance, those related to the user patience model, to the prioritized cache management policy, to weak connectivity, and the like. Because Venus has reasonable defaults for each of these parameters, only interested users would need to explore this area of the interface.

9.2.2.2 Connectivity Information

The interface currently provides connectivity information (via the *Network* indicator light) at the wrong level of granularity (Comment 256). This indicator presents the information at the granularity of individual servers, but users want to know about files and directories. The interface would better serve users by tracking the connectivity of the individual volumes the user accesses and then presenting the connectivity of those volumes through an indicator light (perhaps replacing the *Network* indicator). The informational window associated with this indicator light would present this connectivity information in terms of the pathname at which each volume is mounted.

9.2.2.3 Incorporating Advice into Task Definitions

An obvious extension to the CodaConsole interface would be to connect the windows offering hoard advice to users with the actual program and data definitions used by the system. The basic idea would be to add a button to the Consider Adding and the Consider Removing windows (see Figure 4.11) that would either allow the user to add the identified object to a definition or to remove the object from all definitions. Such an extension would greatly simplify the process of incorporating advice. It also makes collecting the data necessary to validate these heuristics trivial (see Section 9.2.1.1).

9.2.2.4 Incorporating Ideas from Seer

Another interesting extension to the CodaConsole interface would incorporate the ideas used in Seer (see Section 8.1.5 and [26, 27]). The goal of such a system would be to reduce the burden that defining tasks places upon users. I see four approaches that could accomplish this goal.

The first method defines each Seer cluster as a task. This method completely removes the burden of defining tasks from the user, a major benefit. However, this benefit comes with a high cost. Because clusters are not always meaningful to users [27], the *Task* indicator also becomes meaningless. Users can tell that a cluster is not available, but they cannot determine what that unavailability implies. Users would also be unable to prioritize clusters for the same reason. The fact that clusters are not meaningful to users represents a fundamental problem with this approach. For this reason, in the following paragraphs I explore other ways to incorporate Seer with the CodaConsole. These other methods provide a lesser benefit in terms of the requirements for either the cache space or the user's attention, but do not sacrifice the meaningful feedback offered by the *Task* indicator. Of course, should extensions to Seer eventually produce clusters that are meaningful to users, those clusters could easily be substituted for the CodaConsole's user-defined tasks.

The second method exploits Seer clusters to offer advice to the user regarding their definitions. By examining clusters and comparing them with definitions created by users, the system could suggest forgotten files. Again, because the cluster may include files unrelated to this task (at least in the user's mind), the suggestions may be less than helpful to users. The success of this approach depends heavily upon what percentage of files

within a cluster are meaningful and intuitive to users. The current version of Seer is unlikely to achieve an accuracy high enough to support this method [24].

A third alternative would be to use the user's definitions to give feedback to the clustering algorithm. This feedback could easily [24] be incorporated as hints to the Seer clustering algorithm, with what is called an external investigator [27]. At the very least, such feedback could prevent Seer from dividing a single task into multiple clusters. It also has the potential to produce clusters of files that are meaningful, though perhaps noisy, to users. This approach represents a compromise between the automatic nature of Seer and the feedback offered by the CodaConsole. It also allows the user to choose the level of interaction. With no help, she would get Seer's current level of performance, with some feedback from the CodaConsole. After defining tasks, the CodaConsole would provide feedback completely meaningful to the user and Seer would presumably perform even better.

The fourth approach simply uses Seer's clusters to automatically extend the task definitions created by the user. The system could then forgo offering advice to the user about their task definitions. The lack of meaningful clusters would not be problematic because the user would continue to communicate with the system in terms of her defined tasks. The burden that task definitions place on users would be minimal because the definitions would no longer need to be exact.

These four approaches represent a range of possibilities. At one extreme, Seer is silently incorporated into the CodaConsole as tasks. This approach is the ideal because it completely removes the burden of defining tasks from the user, but it is currently impractical because the clusters that Seer produces are not meaningful to users. The other approaches reduce the burden that defining tasks places on users, but do not remove that burden entirely.

9.2.3 Extending the Translucence

The interface presented in Chapter 1 focuses on making caching translucent to the user. Other aspects of Coda must also be made translucent to the user. One such aspect is repairing object replicas that have become inconsistent because of optimistic replication. Another is allowing the user control over the amount of data that trickles back to the servers during reintegration. Although these aspects of the interface have not been implemented, a preliminary design[38] has been developed consistent with the interface presented earlier.

[38] Because the designs presented here represent "paper designs", the screens shown in Figure 9.1 and Figure 9.2 appear with a white background, making them visually distinct from the screens that have already been implemented.

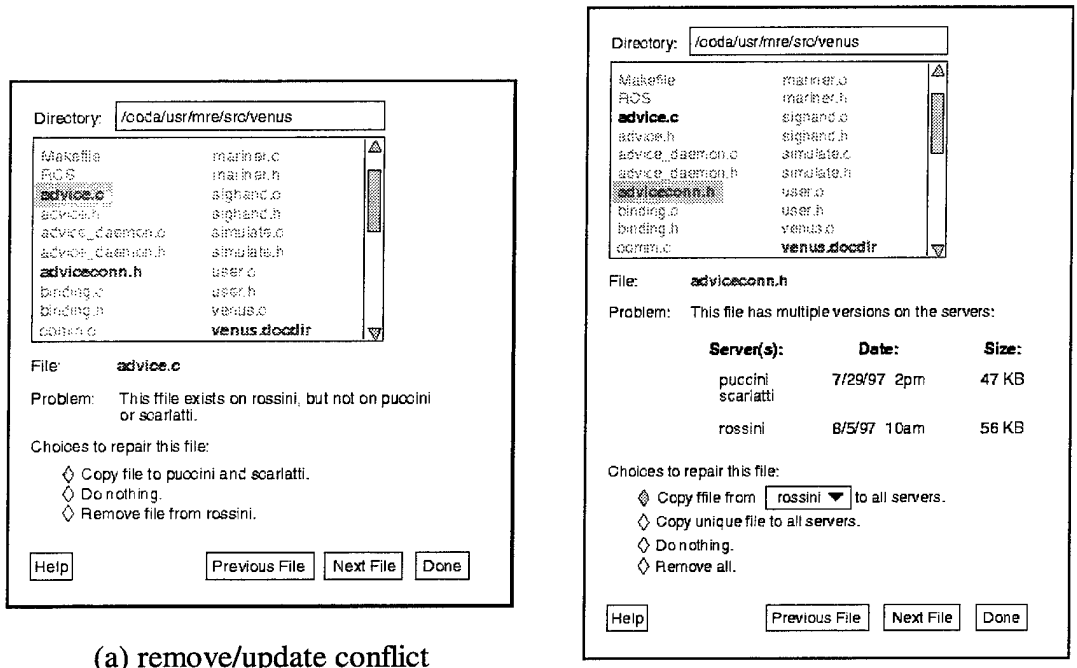
9.2.3.1 Repairing Objects in Conflict

Ideally, users could invoke the tools for manually repairing conflicting replicas (see Section 2.2.1 and [28]) of an object via the CodaConsole interface. The preliminary design for incorporating repair into the CodaConsole interface hooks this function into the *Repair* indicator light. The difference is that the user would now be able to double-click on the name of an object in conflict (in Figure 4.14, the user would be able to double-click on /coda/usr/mre/thesis/dissertation). After doing so, the user would see a screen similar to those shown in Figure 9.1. It would explain the cause of the conflict and give the user some choices as to how to repair the conflict. In the case of true conflict (Figure 9.1b), the user would also be able to look at the different versions of the file.

9.2.3.2 Controlling Trickle Reintegration

Trickle reintegration (see Section 2.2.3 and [34]) uses the network wisely without human intervention. Attentive users, however, should have the ability to control what data gets reintegrated. The preliminary design for giving users this control within the CodaConsole interface hooks into the *Reintegration* indicator light. Once implemented, the *Reintegration Information* window shown in Figure 4.13 would be replaced by that sketched in Figure 9.2(a). From this screen, the user would be able to select a set of subtrees to be reintegrated. In addition, the user would be able to double-click on the name of a subtree needing to be reintegrated to view a display of the CML, as sketched in Figure 9.2(b). From this screen, the user would be able to indicate how many of the updates should be reintegrated. Once reintegration has begun, a window showing the name of the current subtree and a progress meter would appear.

A further extension, one that would make this a task-based interaction, would allow the user to select which task's updates should be reintegrated. Such a version would be far more complicated to implement because of the need to ensure a proper ordering on the updates and is not discussed further here (for more information on the complexities involved, see [33]).

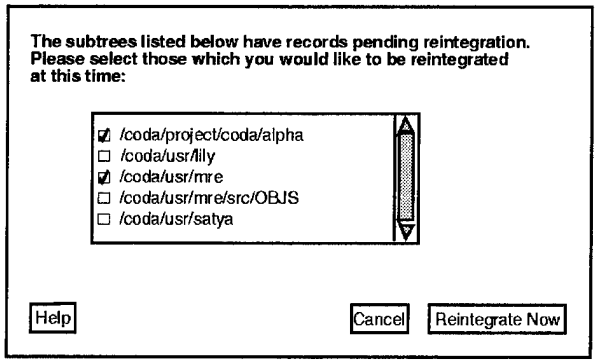


(a) remove/update conflict

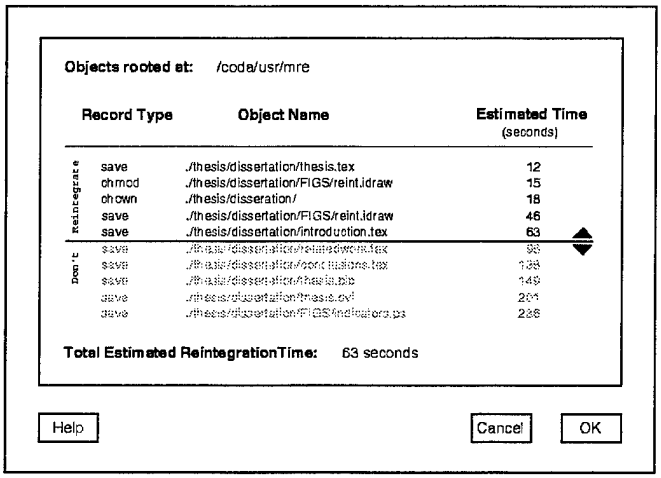
(b) update/update conflict

Figure 9.1: Repairing Objects in Conflict

This figure shows the preliminary design of the repair module for the CodaConsole interface. Windows similar to these would appear when the user double-clicks on the name of an object in conflict in the *Repair Information* window of Figure 4.14. View (a) shows the screen that would allow users to manually repair a directory with a remove/update conflict. In this case, the file exists on `rossini`, but not on `puccini` or `scarlatti`—perhaps one user deleted it while communicating with `puccini` and `scarlatti` but another user updated it while communicating only with `rossini`. View (b) shows the screen that would allow users to repair an update/update conflict manually. In this case, users who were operating partitioned from one or more servers both updated the file. Thus, the version of it stored on `puccini` and `scarlatti` differs from the one on `rossini`. In these drawings, objects whose names appear in red are in conflict, those whose names appear in black have fixes pending, and those whose names appear in gray are not in conflict. Servers whose names appear in blue represent hyperlinks that allow the user to examine the content of individual replicas. The next and previous object buttons take the user to the next or previous object in conflict. The done button begins the actual repair.



(a) Proposed Reintegration Information



(b) Controlling the Amount of Data with a Subtree

Figure 9.2: Reintegration Interface Paper Design

This figure shows the preliminary design of the reintegration module for the CodaConsole interface. View (a), the proposed replacement for the window shown in Figure 4.13, allows the user to select which subtrees (volumes) will be reintegrated. View (b), which would appear after the user double-clicked on a subtree shown in view (a), allows the user to control how many updates within a subtree (volume) get reintegrated. The line in the middle of the listing represents the reintegration barrier [34]. Those updates appearing above the barrier will be reintegrated; those below will be postponed. The user can move the barrier up or down using the arrow widgets. The total estimated reintegration time for those objects selected to be reintegrated is shown near the bottom of the window.

9.3 Final Remarks

Let me now return to our story of Peer, Yoni, Alex, and Leah. When last we heard of our intrepid users, they were having difficulty understanding what the system was doing and why. Alex had asked Yoni to look into a newly released interface add-on that he thought addressed the problems they were experiencing.

> Yoni found the documentation about the new interface and discovered that it did address this problem and more. He downloaded a demo version of the interface and found that it used a small number of indicator lights that kept him informed about the system's activities and about error conditions. Now, as he used his system remotely, he could tell when his data was available to his coworkers at the home office. He also discovered numerous unexpected benefits. With the new system, it became obvious when his tokens expired, when servers crashed, and when a task became unavailable. He quickly and easily discovered objects that became in conflict. He found the advice offered by the system about his task definitions to be useful in discovering files that he had inadvertently forgotten to hoard, and also in cleaning out task definitions that he no longer needed. After a few weeks of running the demo version of the software, he recommended to Alex that he purchase copies for their entire group.[39]

As distributed file systems and mobile computing continue to increase in popularity, users are going to begin experiencing situations similar to those described in this thesis. Users will observe failures. They will experience vastly different network latencies, bandwidths, and costs. The systems will have to adapt, and sometimes that adaptation will require changing the standard behaviors.

Systems will need to expose failures and behavior changes to users or else greatly reduce the usability of the resulting system. These details will need to be carefully presented to users in a meaningful way. This thesis shows one path these systems can take.

---

[39] And, yes, they all lived happily ever after.

A Translucent Caching API

Communication between Venus and the Advice Monitor occurs in both directions. Thus, Venus is a server for the Advice Monitor as well as a client of the Advice Monitor. For this reason, I present the interface is two segments. The first shows those calls for which Venus acts as a server and the Advice Monitor as a client. The second shows those calls for which Venus acts as a client and the Advice Monitor as a server.

A.1 Calls Serviced By Venus

The first half of the API includes those calls serviced by Venus. These calls expose certain *data* to the Advice Monitor.

A.1.1 NewAdviceService

Call:

long NewAdviceService (IN RPC2_String Hostname,
IN RPC2_Integer UserID,
IN RPC2_Integer Port,
IN RPC2_Integer PGrpID,
IN RPC2_Integer AdSrvVersion,
IN RPC2_Integer AdMonVersion,
OUT RPC2_Integer VenusMajorVersion,
OUT RPC2_Integer VenusMinorVersion);

Parameters:

Hostname	Hostname on which Advice Monitor is running (must be "localhost")
UserID	uid that Advice Monitor represents
Port	Port
PgrpID	Process Group ID of Advice Monitor
AdSrvVersion	Version of adsrv.rpc2 in use
AdMonVersion	Version of admon.rpc2 in use
VenusMajorVersion	Venus's major version number

VenusMinorVersion	Venus's minor version number

Completion Codes:

CAEVERSIONSKEW	AdSrvVersion or AdMonVersion passed from the Advice Monitor do not match those of Venus.
CAENOSUCHUSER	The user with UserID does not match any user known to this Venus.
CAEADVICEPENDING	A request for advice is outstanding to another Advice Monitor for this user. (Try again.)
CAESUCCESS	No errors were detected.

Description:

The NewAdviceService call announces the existence of a new Advice Monitor to Venus. This call sets up the advice connection within Venus and allows Venus to detect version skew between itself and the Advice Monitor.

A.1.2 ConnectionAlive

Call:

long ConnectionAlive(IN RPC2_Integer UserID);

Parameters:

UserID	UserID owning requesting Advice Monitor

Completion Codes:

CAENOSUCHUSER	The user with UserID does not match any user known to this Venus.
CAENOTVALID	Connection to this user is not currently valid.
CAESUCCESS	No errors were detected.

Description:

An Advice Monitor uses this call to verify the status of its connection to Venus.

A.1.3 GetServerInformation

Call:

long GetServerInformation(IN RPC2_Integer MaxServers, OUT ServerEnt servers[]);

Parameters:

MaxServers	The maximum number of servers for which Venus should return information.
servers[]	An array of server entries each containing the name of a server and an estimate of Venus's connectivity to that server

Completion Codes:

CAENOSERVERS	Venus is not currently in contact with any servers.
CAESUCCESS	No errors were detected.

Description:

An Advice Monitor uses this call to determine the names of all servers with which Venus is currently communicating. The ServerEnt type is defined as follows:

```
typedef RPC2_Struct {
        RPC2_String name;
        RPC2_Integer bw;
        } ServerEnt;
```

N.B. The implementation of arrays in RPC2 requires a long parameter before the array parameter. The value of the long should be the number of elements in the array.

A.1.4 RegisterInterest

Call:

long RegisterInterest(IN RPC2_Integer UserID, InterestValuePair events[]);

Parameters:

UserID	UserID owning requesting Advice Monitor
InterestValuePair[]	An array of events each with an indication of user's interest in them

Completion Codes:

CAENOSUCHUSER	The user with UserID does not match any user known to this Venus.
CAESUCCESS	No errors were detected.

Description:

An Advice Monitor uses this call to register the user's interests in various events. The InterestValuePair type is defined as follows:

```
typedef RPC2_Struct {
        InterestID event;
        RPC2_Integer value;
} InterestValuePair;
```

The list below shows the valid interest identifiers:

TokensAcquiredID	HoardWalkPeriodicOnID
TokensExpiredID	HoardWalkPeriodicOffID

ActivityPendingTokensID	ObjectInConflictID
SpaceInformationID	ObjectConsistentID
ServerAccessibleID	ReintegrationPendingTokensID
ServerInaccessibleID	ReintegrationEnabledID
NetworkQualityEstimateID	ReintegrationActiveID
ReconnectionID	ReintegrationCompletedID
ReadDisconnectedCacheMissEventID	TaskAvailabilityID
WeaklyConnectedCacheMissEventID	TaskUnavailabilityID
DisconnectedCacheMissEventID	ProgramAccessLogsID
HoardWalkAdviceRequestID	ReplacementLogsID
HoardWalkBeginID	InvokeASRID
HoardWalkEndID	

N.B. The implementation of arrays in RPC2 requires a long parameter before the array parameter. The value of the long should be the number of elements in the array.

A.1.5 GetCacheStatistics

Call:

long GetCacheStatistics(OUT RPC2_Integer FilesAllocated,
OUT RPC2_Integer FilesOccupied,
OUT RPC2_Integer BlocksAllocated,
OUT RPC2_Integer BlocksOccupied,
OUT RPC2_Integer RVMAllocated,
OUT RPC2_Integer RVMOccupied);

Parameters:

FilesAllocated	Number of fsobjs allocated by Venus
FilesOccupied	Number of fsobjs occupied by hoarded objects
BlocksAllocated	Number of cache blocks allocated by Venus
BlocksOccupied	Number of cache blocks occupied by hoarded objects
RVMAllocated	Amount of RVM space allocated
RVMOccupied	Amount of RVM space occupied

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

An Advice Monitor uses this call to determine the current space usage within Venus. Files and Blocks refer to the number of fsobjs and blocks maintained by Venus for its cache. RVM refers to the internal recoverable virtual memory used by Venus to store persistent state.

A.1.6 OutputUsageStatistics

Call:

long OutputUsageStatistics(IN RPC2_Integer UserID,
IN RPC2_String Pathname,
IN RPC2_Integer DisconnectionsSinceLastUse,
IN RPC2_Integer PercentDisconnectionsUsed,
IN RPC2_Integer TotalDisconnectionsUsed);

Parameters:

UserID	User owning requesting Advice Monitor
Pathname	Pathname of file in which to dump usage statistics
DisconnectionsSinceLastUse	Number of active disconnections since an object was last used during disconnected operation
PercentDisconnectionsUsed	Percentage of active disconnections during which object has been used
TotalDisconnectionsUsed	Number of active disconnections during which object was used

Completion Codes:

CAENOSUCHUSER	The user with UserID does not match any user known to this Venus.
CAESUCCESS	No errors were detected.

Description:

An Advice Monitor uses this call to request that Venus output disconnected usage statistics for cached files. Venus identifies objects which have not been used during the N most recent disconnections, where N = DisconnectionsSinceLastUse. Venus also identifies objects which have been used for fewer than M% of active disconnections (where M = PercentageDisconnectionsUsed) and which have been used during a minimum of T disconnections (where T = TotalDisconnectionsUsed). It outputs the relevant data to the specified file.

A.1.7 SetParameters

Call:

long SetParameters(IN RPC2_Integer UserID, IN ParameterValuePair parameters[]);

Parameters:

UserID	User owning requesting Advice Monitor
parameters	An array of parameters, each with a new value

Completion Codes:

CAENOTAUTHORIZED	This user is not authorized to change Venus parameter values.
CAESUCCESS	No errors were detected.

Description:
An Advice Monitor can use this call to set certain Venus parameters dynamically. The current list of parameters that can be set dynamically is small, but can be extended with minimal effort. This list contains:

AgeLimit	Minimum age of records (in seconds).
ReintLimit	Work Limit (in milliseconds).
ReintBarrier	Included for future extension to allow the Advice Monitor to modify the location of the reintegration barrier (see Section 9.2.3.2.)
WeakThreshold	Threshold below which Venus transitions into weakly connected operation (in Bytes/second).
PatienceAlpha	alpha parameter of the user patience model (in seconds)
PatienceBeta	beta parameter of the user patience model
PatienceGamma	gamma parameter of the user patience model

N.B. The implementation of arrays in RPC2 requires a long parameter before the array parameter. The value of the long should be the number of elements in the array.

A.1.8 ResultOfASR

Call:
long ResultOfASR(IN RPC2_Integer ASRid, IN RPC2_Integer result);

Parameters:

ASRid	Identification of ASR
result	Result of that ASR's execution

Completion Codes:

CAENOASR	No ASRs are currently in progress.
CAEUNEXPECTEDASR	ASRid does not match the ID of the ASR currently in progress.
CAESUCCESS	No errors were detected.

Description:
An Advice Monitor uses this call to return the result of an application-specific resolver.

A.1.9 Imminent Death

Call:
long ImminentDeath(IN RPC2_String HostName,
IN RPC2_Integer UserID,
IN RPC2_Integer Port);

Parameters:

HostName	HostName on which Advice Monitor running (ignored)
UserID	User owning requesting Advice Monitor
Port	Port on which Advice Monitor listening (ignored)

Completion Codes:

CAENOSUCHUSER	The user with UserID does not match any user known to this Venus.
CAESUCCESS	No errors were detected.

Description:

The Advice Monitor can use this call to inform Venus of its imminent death.

A.2 Calls Serviced By Advice Monitor

The second half of the API includes those calls serviced by the Advice Monitor. These calls expose certain *events* to the Advice Monitor.

A.2.1 LostConnection

Call:

long LostConnection();

Parameters:

None.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

The LostConnection() call serves to inform an Advice Monitor that it has lost its connection with Venus.

A.2.2 TokensAcquired

Call:

long TokensAcquired(IN RPC2_Integer EndTimestamp);

Parameters:

EndTimestamp	The time at which the user's tokens are scheduled to expire.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when the user obtains new tokens.

A.2.3 TokensExpired

Call:

long TokensExpired();

Parameters:

None.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a user's tokens have expired.

A.2.4 ActivityPendingTokens

Call:

long ActivityPendingTokens(IN ActivityID activityType, IN RPC2_String argument);

Parameters:

ActivityType	Used to identify the type of activity that requires the user to obtain tokens.
argument	An optional argument to specify a file or volume name requiring the specified activity.

Completion Codes:

CAEINVALIDARG	Activity provided is not valid.
CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a subsystem requires authentication before proceeding. For example, when an object needs to be repaired or when a reintegration is pending.

A.2.5 SpaceInformation

Call:

long SpaceInformation(IN RPC2_Integer PercentFilesFilledByHoardedData
IN RPC2_Integer PercentBlocksFilledByHoardedData,
IN RPC2_Integer PercentRVMFull,
IN Boolean RVMFragmented);

Parameters:

PercentFilesFilledByHoardedData	The percentage of cache files filled with hoarded data.
PercentBlocksFilledByHoardedData	The percentage of cache blocks filled with hoarded data.
PercentRVMFull	The percentage of RVM that is filled.
RVMFragmented	An indication of whether or not RVM is fragmented.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call to inform the Advice Server about the current space status.

A.2.6 ServerAccessible

Call:

long ServerAccessible(IN RPC2_String ServerName);

Parameters:

ServerName	The server that has become accessible.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a server becomes accessible.

A.2.7 ServerInaccessible

Call:

long ServerInaccessible(IN RPC2_String ServerName);

Parameters:

ServerName	The server that has become inaccessible.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a server becomes inaccessible.

A.2.8 ServerConnectionWeak

Call:

long ServerConnectionWeak(IN RPC2_String ServerName);

Parameters:

ServerName	The server whose connection has become weak.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when the connection to a server becomes weak.

A.2.9 ServerConnectionStrong

Call:

long ServerConnectionStrong(IN RPC2_String ServerName);

Parameters:

ServerName	The server whose connection has become strong.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when the connection to a server becomes strong.

A.2.10 NetworkQualityEstimate

Call:

long NetworkQualityEstimate(IN QualityEstimate serverList[]);

Parameters:

ServerList[]	An array of QualityEstimate structures specifying the estimated network quality for a number of servers.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call to inform the Advice Monitor of the strength of its connections to servers. The QualityEstimate structure contains the following information:

```
typedef RPC2_Struct {
        RPC2_String ServerName;
        RPC2_Integer BandwidthEstimate;
        Boolean Intermittent;
} QualityEstimate;
```

N.B. The implementation of arrays in RPC2 requires a long parameter before the array parameter. The value of the long should be the number of elements in the array.

A.2.11 Reconnection

Call:

long Reconnection(IN ReconnectionQuestionnaire Questionnaire,
OUT RPC2_Integer ReturnCode);

Parameters:

Questionnaire	A data structure containing statistics about the recent disconnected session.
ReturnCode	(Not used.)

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a volume transitions into the Hoarding state from any other state. The input structure contains the following information:

```
typedef RPC2_Struct {
        RPC2_Integer RQVersionNumber;
        RPC2_String VolumeName;
        VolumeId VID;
        RPC2_Unsigned CMLcount;
        RPC2_Unsigned TimeOfDisconnection;
        RPC2_Unsigned TimeOfReconnection;
        RPC2_Unsigned TimeOfLastDemandHoardWalk;
        RPC2_Unsigned NumberOfReboots;
        RPC2_Unsigned NumberOfCacheHits;
        RPC2_Unsigned NumberOfCacheMisses;
        RPC2_Unsigned NumberOfUniqueCacheHits;
        RPC2_Unsigned NumberOfObjectsNotReferenced;
} ReconnectionQuestionnaire;
```

A.2.12 ReadDisconnectedCacheMiss Event

Call:

```
long ReadDisconnectedCacheMissEvent(IN ObjectInformation objInfo,
        IN ProcessInformation processInfo,
        IN RPC2_Unsigned TimeOfMiss,
        OUT CacheMissAdvice Advice,
        OUT RPC2_Integer ReturnCode);
```

Parameters:

objInfo	Information about the missing object.
processInfo	Information about the requesting program.
TimeOfMiss	Time at which the miss occurred.
Advice	User's advice regarding this miss.
ReturnCode	Return value.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a cache miss occurs while Venus is operating read disconnected.

A.2.13 WeaklyConnectedCacheMiss Event

Call:

```
long WeaklyConnectedCacheMissEvent(IN ObjectInformation objInfo,
        IN ProcessInformation processInfo,
```

IN RPC2_Unsigned TimeOfMiss,
IN RPC2_Integer Length,
IN RPC2_Integer EstimatedBandwidth,
IN RPC2_String Vfile,
OUT CacheMissAdvice Advice,
OUT RPC2_Integer ReturnCode);

Parameters:

objInfo	Information about the missing object.
processInfo	Information about the requesting program.
TimeOfMiss	Time at which the miss occurred.
Length	Size of the missing object.
Estimated Bandwidth	Current estimated bandwidth.
Vfile	Name of cache file (on the local disk) in which Venus will store the data.
Advice	User's advice regarding this miss.
ReturnCode	Return value.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a cache miss occurs while Venus is operating weakly-connected.

A.2.14 DisconnectedCacheMissEvent

Call:

long DisconnectedCacheMissEvent(IN ObjectInformation objInfo,
IN ProcessInformation processInfo,
IN RPC2_Unsigned TimeOfMiss,
IN RPC2_Unsigned TimeOfDisconnection,
OUT RPC2_Integer ReturnCode);

Parameters:

objInfo	Information about the missing object.
processInfo	Information about the requesting program.
TimeOfMiss	Time at which the miss occurred.
TimeOfDisconnection	Time at which disconnection occurred.
ReturnCode	Return value.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a cache miss occurs during disconnected operation.

A.2.15 HoardWalkAdviceRequest

Call:

long HoardWalkAdviceRequest(IN RPC2_String InputPathname,
IN RPC2_String OutputPathname,
RPC2_Integer ReturnCode);

Parameters:

InputPathname	Name of file containing information about this hoard walk request.
OutputPathname	Name of file that should be used to provide advice for the objects contained in InputPathname.
ReturnCode	Not Used.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call to request hoard walk advice from the user.

A.2.16 HoardWalkBegin

Call:

long HoardWalkBegin();

Parameters:

None.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a hoard walk begins.

A.2.17 HoardWalkStatus

Call:

long HoardWalkStatus(IN RPC2_Integer Percentage);

Parameters:

Percentage	Percentage of the hoard walk completed to date.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call to inform the Advice Monitor of the walk's progress.

A.2.18 HoardWalkEnd

Call:

long HoardWalkEnd();

Parameters:

None.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a hoard walk completes.

A.2.19 HoardWalkPeriodicOn

Call:

long HoardWalkPeriodicOn();

Parameters:

None.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when periodic hoard walks are enabled.

A.2.20 HoardWalkPeriodicOff

Call:
long HoardWalkPeriodicOff();

Parameters:
None.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:
Venus makes this RPC call when periodic hoard walks are disabled.

A.2.21 ObjectInConflict

Call:
long ObjectInConflict(IN RPC2_String Pathname, IN ViceFid fid);

Parameters:

Pathname	Pathname of object in conflict.
fid	File identifier of object in conflict.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:
Venus makes this RPC call when it detects that an object is in conflict.

A.2.22 ObjectConsistent

Call:
long ObjectConsistent(IN RPC2_String Pathname, IN ViceFid fid);

Parameters:

Pathname	Pathname of object now consistent.
fid	File identifier of object now consistent.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when an object in conflict is successfully repaired.

A.2.23 ReintegrationPendingTokens

Call:

long ReintegrationPendingTokens(IN RPC2_String volumeID);

Parameters:

volumeID	Identifier of volume needing to be reintegrated.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a volume needs to be reintegrated, but for which the user's tokens are not valid.

A.2.24 ReintegrationEnabled

Call:

long ReintegrationEnabled(IN RPC2_String volumeID);

Parameters:

volumeID	Identifier of volume for which reintegration has been enabled.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when a volume has modification that need to be reintegrated, but which have not yet aged.

A.2.25 ReintegrationActive

Call:
long ReintegrationActive(IN RPC2_String volumeID);

Parameters:

volumeID	Identifier of volume currently being reintegrated.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:
Venus makes this RPC call when a volume is actively being reintegrated.

A.2.26 ReintegrationCompleted

Call:
long ReintegrationCompleted(IN RPC2_Volume volumeID);

Parameters:

volumeID	Identifier of volume for which reintegration has completed.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:
Venus makes this RPC call when reintegration of a volume completes.

A.2.27 TaskAvailability

Call:
long TaskAvailability(IN TallyInfo taskList[]);

Parameters:

TaskList[]	An array of TallyInfo structures describing the current availability of numerous tasks.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:
Venus makes this RPC call to inform the Advice Monitor of the availability of hoarded tasks. The call is triggered upon completion of each hoard walk.

N.B. The implementation of arrays in RPC2 requires a long parameter before the array parameter. The value of the long should be the number of elements in the array.

A.2.28 TaskUnavailability

Call:
long TaskUnavailability(IN RPC2_Integer TaskPriority, RPC2_Integer ElementSize);

Parameters:

TaskPriority	Priority of unavailable task.
ElementSize	Size of unavailable item.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:
Venus makes this RPC call when it evicts or invalidates a hoarded object.

A.2.29 ProgramAccessLogAvailable

Call:
long ProgramAccessLogAvailable(IN RPC2_String LogFile);

Parameters:

LogFile	Name of file containing program access log.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:
Venus makes this RPC call when it fills a program access log.

A.2.30 ReplacementLogAvailable

Call:

long ReplacementLogAvailable(IN RPC2_String LogFile);

Parameters:

LogFile	Name of file containing replacement log.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when it fills a replacement log.

A.2.31 InvokeASR

Call:

long InvokeASR(IN RPC2_String Pathname,
IN RPC2_Integer uid,
OUT RPC2_Integer ASRid,
OUT RPC2_Integer ReturnCode);

Parameters:

Pathname	Pathname of object needing application-specific resolution (ASR).
Uid	Identification of user invoking ASR.
ASRid	Identifier of applicaton-specific resolver.
ReturnCode	Return value.

Completion Codes:

CAESUCCESS	No errors were detected.

Description:

Venus makes this RPC call when it detects an object that is in conflict. The value of the ReturnCode will be either ADMON_SUCCESS (if the ASR was successfully started) or ADMON_FAIL (if the ASR was not successfully started).

B Materials

This appendix contains the materials used during the CodaConsole usability study. In the sections that follow, you will find:

- **Procedures:** used by the experimenter to maintain consistency between participants
- **Consent Form:** required of all participants
- **Background Questionnaire:** administered before beginning the Tutorial
- **Tutorial**[1]**:** taught user about the interface
- **Exercises**[2]**:** tested user understanding of interface and ability to operate it
- **Evaluation Questionnaire:** administered after completion of the Exercises
- **Answer Key:** used by the experimenter to score the accuracy of the user's responses
- **Par Key:** used by the experimenter to score the efficiency of the user's responses

For a discussion of the goals, methods, and results of the usability study, please refer to Chapter 7, "Evaluating the Interface".

[1] This appendix contains a hard-copy version of the on-line Tutorial.
[2] This appendix contains a hard-copy version of the on-line Exercises.

B.1 Procedures

1) Note Time: _____
2) Reboot telos
3) Label Tapes for Tutorial and Exercises (7)
   a) Audio
   b) Video (8mm)
   c) Video (SVHS)
4) Lab Startup Procedure
5) Verify all cameras and mikes operational
6) Login as bovik
   a) kinit as mre
   b) clog
   c) xinit
   d) xterm (2 – iconify one)
   e) Tutorial -subject <ID>
7) Wait for user to arrive.
8) Note Time: _____
9) Consent Form

   Good morning. Thank you for agreeing to participate in my user study. Please forgive me for reading from this script. Because it is important that I treat every user the same, I must read from a script.

   In today's user study, you will be using an experimental interface to the Coda File System. This study consists of four parts. First, you will begin by filling out a brief questionnaire. Then, you will go through a tutorial to teach you how to use the interface. This tutorial takes approximately 1 to 1.5 hours. After you complete the tutorial, we will take a break. Then, you will be asked to perform some exercises using the interface. These exercises are expected to take 1 - 1.5 hours. Once you have finished the exercises, you will be asked to fill out a brief evaluation questionnaire. Then, we can discuss any questions you might have.

   Before we begin, we must take care of some formalities. Please read through this consent form carefully. If you are still interested in participating, please sign it.

   Check consent form, look for initials.

10) Background Questionnaire

Thank you. Now, please read and answer each of the questions in this questionnaire. We will use this information to help us interpret the results of our study. Your responses will be kept confidential and will be reported with your participant ID number only.

Check background questionnaire.

11) Play game to demonstrate verbal protocol

While you are using our interface, we will want you to be thinking out loud. Basically, this just means that you should say what you are thinking. Both positive and negative comments are appreciated. Since many people are unfamiliar with this, we would like to demonstrate what we'd like you to do using a card game called Klondike.

Demonstrate.

Now, you try it with Othello...

Give participant feedback.

12) Ask user to get comfortable, but...

Let's go into the other room now. Please sit down and make yourself comfortable. If it gets too cold in here, we can move some carpeting pieces to "control" the air conditioning. If it gets too hot in here, feel free to open the door.

a) Please put on microphone(s).

b) Point out "Cheat Sheet"

You will be borrowing Harry Q. Bovik's account today. Harry uses ctwm as his window manager. You may find it useful to refer to the "Cheat Sheet" **<point to it>** to see how to manipulate windows in his environment. Before we begin, I'd like you to do four window manipulation tasks.

- Please move a window.
- Please put a window in the background.
- Please bring a window to the foreground.
- Please close a window.

c) Please do NOT start until I tell you to...

13) Begin Recording (3)

Ask user to start – "OK, please begin."

14) Note Time: _____

15) Wait for user to complete tutorial.

16) Note Time: _____

You have completed the first half of the study. We will now take a break of_____ minutes. During this time, it is important that you not discuss this study with anyone. Let's resume at: _____

17) Stop Recording and Rewind (3)

18) Remove and File Tutorial Tapes (3)

19) Reset audio tape counter

20) Insert Exercises Tapes (hit record on audio) (3)

21) Start Exercises Program: Exercises -subject <ID>

22) Wait for user to arrive

I hope you had a nice break. We will now begin the exercises portion of the study.

23) Ask user to get comfortable, but...

a) Point out the xterm – "should you need to use it during the exercises".

b) Please put on microphone.

c) Please do NOT start yet.

24) Begin Recording (3)

25) Ask user to start – "OK, please begin."

26) Note Time: _____

27) Wait for user to complete exercises

28) Note Time: _____

29) Ask them to fill out the Evaluation Questionnaire

You have completed the exercises. We would now like you to tell us about your experience with the interface you just used. Please read and answer each of the questions in this survey.

30) Stop Recording and Rewind all tapes (3)

31) Remove, flip protect bits, and file all tapes (3)

32) Insert Debriefing tape and begin recording

33) Debriefing

Do you have any questions or comments that you would like to discuss?

34) Thank them and give them T-Shirt! \textbf{Get Initials on Contact Form}

35) Stop Recording, rewind debriefing tape, and flip protect bits.

36) Remove and File Tape

37) Check that all record tabs are either flipped or removed

38) Log out

39) Shutdown Lab

40) Note Time: _____

41) Send mail to sesch@cs documenting lab usage time.

42) List critical observations

43) IncidentSummary <SubjectID> <IR Numbers> > ../<SubjectID>/IncidentReportSummary

44) ProcessExerciseData <SubjectID> ../<SubjectID>/ExercisesSummary.<SubjectID>

45) Cut scrolling from ActionLog and move closes to the correct locations

46) Print IncidentReportSummary, ExercisesSummary, Answers, ActionLog

47) Print Processed timing files and Raw exercise timing file.

48) Grade Answers

49) Calculate Par

50) Fill data into thesis

a) Fill Background Questionnaire data

b) Fill timing data for tutorial

c) Fill timing data for exercises

d) Fill Grades

e) Fill Par Score

f) Fill Evaluation Questionnaire data

g) Duplicate all papers and file.

51) File original contact info and consent forms.

B.2 Consent Form

Carnegie Mellon University
Experiment Participation Consent Form

Purpose of Study

The School of Computer Science at Carnegie Mellon University is asking you to participate in a study of computer software. By participating in this study you will help us evaluate the software and make it easier to learn and use. The study will take place in a laboratory in Wean Hall, where you will be observed as you use the software. We will teach you how to use the software. We will ask you to use the software, while recording information about how you do so. We will ask you to fill out two questionnaires, and we may interview you. We will use the information you give us, along with the information we collect from other people, to evaluate the software and recommend ways to improve it.

The purpose of this study is to learn about the usability of our software. **Any and all problems** that you encounter during the course of the study reflect problems in the interface – not you! You represent a typical user of our system. If you encounter problems with the interface, others will as well. Please be sure to point out **any** problems, even minor ones, you experience so that we can improve the interface for future users of our system.

Videotaping

Your work with the interface will be videotaped by one or more cameras. One or two cameras will be focusing on the screen, but will be shooting from over your shoulder. Another camera will provide a closeup of your hand on the mouse. Although no camera will be focused directly on your face, your identity could be recognized from your image on this video or from the sound of your voice as you speak (you will be wearing a microphone during the study and you will be asked to speak aloud as you use the system). You have the right to ask where the cameras are located so that you may avoid looking directly at them during the course of the study.

The researchers performing this study may want to use short clips of these videotapes (including your voice, statements, and image) for illustrative reasons during presentations of this work. These presentations include talks given to colleagues both at local seminars and thesis defenses and elsewhere at workshops, conferences, job interviews, and other meetings. The researchers will not reveal your name during these presentations -- users will only be referred to by a unique identifier (as described in the next section).

By signing this form, you give your permission for the researchers involved in this study to use the videotape(s) (including any captured image of you and any recording of your verbal comments) of your session for the purpose of evaluating this software as well as for illustrative reasons during presentations of this research as described above.

Initials: _____

(over)

Anonymity

In order to maintain anonymity in analysis and reporting, each participant will be assigned a unique identifier, independent of name and initials. The researchers will save the data and videotape by this identifier. Only members of the research group will view the tapes in detail. Only a minimal number of people will know the mapping between identifiers and names. All presentations (both written and oral) of this work will report the results using participant identifiers.

Breaks

We expect the study to take a total of about two hours. We have scheduled a break for you at the midpoint; however, you may take a break at any other time you wish. Merely inform the test administrator that you would like to do so.

Compensation

Upon completion of the study, you will receive a Coda T-shirt.

Withdrawing

You may withdraw from this study at any time for any reason.

Questions or Concerns

If you have questions or concerns about this study, you may contact either of the following people:

M. Satyanarayanan
Professor
Carnegie Mellon University
8115 Wean Hall
(412) 268-3743

Susan Burkett
Associate Provost
Carnegie Mellon University
402 Warner Hall
(412) 268-8746

Consent

I understand that in signing this consent form, I give M. Satyanarayanan, and his associates on this project, permission to present this work, in written, visual, and oral form, without further permission from me.

Signature: _____

Printed Name: _____

Date: _____

B.3 Background Questionnaire

The information in this questionnaire will help us identify users for our user study. In addition, it will help us interpret the results of the usability study. The answers you provide will be held in confidence and will be reported only as identified by your participant ID. Please circle your selection or answer each question, as appropriate.

What is your gender?

(a) Female
(a) Male

What is your highest degree obtained?

(a) None
(b) High School Diploma
(c) Bachelors
(d) Masters
(e) PhD

What is your job title?

(a) Administrative Support Staff
(b) PostDoc
(c) Professor
(d) Research Scientist
(e) Student
(f) Technical Support Staff
(g) Other: _____

If you are a student,

What is your field of study? _____

What is your current level of study? _____

What is your current year in that program? _____

Do you have any experience with human-computer interaction (HCI)?

(a) Yes
(b) No

If so, what?

1. During the last two weeks, how often did you use a computer?

(a) never
(b) once
(c) once a week
(d) every other day
(e) daily

2. Which operating systems are you proficient in using? (Circle all that apply.)

(a) DOS
(b) Mac OS
(c) A flavor of Unix
(d) Other: _____

3. a. Have you ever worked on a computer that had a windowing environment?

(a) yes
(b) No Skip to question 4

b. If so, which environments? (Circle all that apply.)

(a) DEC Windows
(b) MacIntosh Window Manager
(c) MS-Windows
(d) NeXTStep
(e) Presentation Manager
(f) Sun/Open Look or SunView
(g) X Windows
(h) Other: _____

4. The following questions relate to your experience with Unix-based systems.

   a. Have you ever used a Unix system?

      (a) yes
      (b) no Skip to question 7

   b. During the last two weeks, how often did you use Unix systems?

      (a) never
      (b) once
      (c) once a week
      (d) every other day
      (e) daily

   c. How many years experience do you having with Unix systems?

      (a) 1 or less
      (b) 2
      (c) 3
      (d) 4
      (e) 5 or more

   d. How do you rate your ability to use Unix?

      (a) poor
      (b) fair
      (c) ok
      (d) good
      (e) excellent

5. The following questions relate to your experience using X windows and related programs.

   a. Have you ever used X windows?

      (a) yes
      (b) no Skip to question 6

   b. If so, which window manager do you prefer? _____

6. The following questions relate to your experience using various Unix applications.

a. Which editors are you proficient in using? (Circle all that apply.)

(a) vi
(b) emacs
(c) gnu-emacs
(d) epoch
(e) other: _____

b. Which one do you prefer to use? _____

c. Which document processing tools are you proficient in using? (Circle all that apply.)

(a) scribe
(b) latex
(c) frame
(d) other: _____

d. Which one do you prefer to use? _____

e. Have you ever maintained or compiled any Unix applications, either here at Carnegie Mellon or elsewhere?

(a) yes
(b) no

f. If so, which one(s)?

7. The following questions relate to your experience with AFS.

   a. Have you ever used AFS?

      (a) yes
      (b) no Skip to question 8

   b. During the last two weeks, how often did you use AFS?

      (a) never
      (b) once
      (c) once a week
      (d) every other day
      (e) daily

   c. Does your primary computer have your home directory in AFS?

      (a) yes
      (b) no

   d. Do you store your e-mail in AFS?

      (a) yes
      (b) no

8. The following questions relate to your experience with Coda.

   a. Have you ever used the Coda File System?

      (a) yes
      (b) no

   b. Do you know anything about the Coda File System?

      (a) yes
      (b) no

   c. If so, what?

9. The following questions relate to your experience with the Windows 95 Briefcase.

   a. Have you ever used Briefcase?

      (a) yes
      (b) no

   b. Do you know anything about Briefcase?

      (a) yes
      (b) no

   c. If so, what?

Thank you for your time!

B.4 Tutorial

Order of Events:

1. Display Initial
2. Store "BeginTime"
3. Store Begin Time for "Study"
4. Store Begin Time for "Introduction"
5. Display Introduction1
6. Display Introduction2
7. Display Introduction3
8. Display Introduction4
9. Display Introduction5
10. Display Introduction6
11. Display Introduction7
12. Display Introduction8
13. Display Introduction9
14. Execute CodaConsole Interface
15. Display Introduction10
16. Display Introduction11
17. Store End Time for "Introduction"
18. Store Begin Time for "Urgency Colors"
19. Display UrgencyColors1
20. Display UrgencyColors2
21. Store End Time for "Urgency Colors"
22. Store Begin Time for "Network"
23. Fix Space Statistics
24. Simulate weakly connected operation with respect to `scarlatti`
25. Display Network1
26. Simulate disconnected operation with respect to `scarlatti` (timed for 1.5 s. delay)
27. Display Network2
28. Display Network3
29. Fix network connection to `scarlatti`
30. Store End Time for "Network"
31. Store Begin Time for "Task"
32. Display Task1
33. Display Task2
34. Display Task3
35. Display Task4
36. Display Task5
37. Display Task6
38. Display Task7
39. Display Task8
40. Display Task9
41. Display Task10
42. Display Task11
43. Display Task12
44. Display Task13
45. Display Task14

46. Display Task15
47. Display Task16
48. Display Task17
49. Display Task18
50. Display Task19
51. Display Task20
52. Display Task21
53. Store End Time for "Task"
54. Store Begin Time for "Advice"
55. Display Advice1
56. Simulate weakly connected operation with respect to all servers.
57. Simulate a "Weakly Connected Cache Miss Advice" event with a delay of 10 seconds.
58. Display Advice2
59. Configure "Weakly Connected Cache Miss Advice" event to pop-up a window on user's screen.
60. Display Advice3
61. Simulate a second "Weakly Connected Cache Miss Advice" event with a 13 second delay.
62. Display Advice4
63. Display Advice5
64. Store End Time for "Advice"
65. Store Begin Time for "HoardWalk"
66. Display HoardWalk1
67. Send the interface a request to begin a hoard walk.
68. Send the interface a request for hoard walk advice with a delay of 4 seconds.
69. Display HoardWalk2
70. Display HoardWalk3
71. Display HoardWalk4
72. Display HoardWalk5
73. Display HoardWalk6
74. Store End Time for "HoardWalk"
75. Store Begin Time for "Space"
76. Simulate that the RVM space is almost full (critical) with a delay of 3 seconds.
77. Display Space1
78. Fix the RVM space problem.
79. Display Space2
80. Simulate that the cache space is starting to fill (warning) with a delay of 3 seconds.
81. Display Space3
82. Fix the cache space warning with a delay of 1.5 seconds.
83. Display Space4
84. Store End Time for "Space"
85. Store Begin Time for "Tokens"
86. Display Tokens1
87. Display Tokens2
88. Display Tokens3
89. Store End Time for "Tokens"
90. Store Begin Time for "EventConfiguration"
91. Display EventConfiguration1
92. Display EventConfiguration2
93. Display EventConfiguration3
94. Display EventConfiguration4
95. Display EventConfiguration5
96. Store End Time for "EventConfiguration"
97. Simulate a "Token Expiry" event
98. Display TokenExpiry
99. Store End Time for "Study"
100. Shutdown Tutorial

Please wait to begin.

Screen 1: Initial

Thank you for agreeing to participate in our usability study.

To make it easy for you to distinguish between our interface and the on-line tutorial and exercises, we have made the background color of the on-line tutorial and exercises pink. Thus, if the window is approximately the same color as the walls, it's part of the study -- not the interface you're testing.

Each screen of text is followed by an "OK" button. Click on this button to continue to the next screen (or hit <Enter>).

Screen 2: Introduction1

Occasionally, the cursor changes to a picture of a clock or a picture of a mouse (a square with three rectangular boxes inside it). If this happens and it doesn't automatically go away, simply click on the X background. If it still doesn't go away, come get me. Similarly, if the program seems to hang or you believe something is seriously wrong, come get me.

Screen 3: Introduction2

We will now start the tutorial. The tutorial begins by explaining what problems the Coda File System addresses and how you might use it. The tutorial then continues by introducing a new interface to Coda and explaining each portion of that interface in detail. When you need to perform an action, you will see instructions preceded by "→". Like this:

→ Please click on the "OK" button below.

Screen 4: Introduction3

Coda is a distributed file system, and is a descendant of the Andrew File System that you use every day. It inherits many of the properties of AFS. When everything is working well, you would have a difficult time distinguishing between Coda and AFS.

As a user of AFS, you are probably aware that when a server goes down you cannot access data on that server. Similarly, if the network goes down, you cannot access any data in AFS. Users often find this frustrating since the machines sitting on their desks are frequently fully operational.

Coda was created to solve this problem. Our goal was to provide "high data availability". This term simply means that we wanted to be able to continue to work even if our desktop machine was completely unable to communicate with the servers (either because the server(s) crashed or because the network connections were down). A consequence of high availability is that we can voluntarily disconnect our laptop from the network (for instance, to take it on an airplane) and still continue to access our data.

When a Coda client machine (either a desktop or a laptop) is unable to communicate with the Coda servers, it is said to be operating "disconnected".

Screen 5: Introduction4

In case you weren't aware of this, both AFS and Coda cache files on your local disk after they fetch them from the AFS or Coda servers. AFS does this solely for performance reasons. Coda, as you will see, benefits in terms of performance too, but also uses this cache to provide availability.

The caveat to disconnected operation is that the user can only access data stored in the client's local cache. To achieve high availability then, we have to make sure that the cache contains the data that the user wants to access in the event of a disconnection. One might assume that a least recently used (LRU) caching policy would suffice. And, for short-term network failures, it often does. However, now that we are trying to provide high availablility for users of laptop computers, LRU simply doesn't suffice. The problem is that, while disconnected, the user may want to work on a project that they haven't been using recently.

To combat this problem, Coda provides a mechanism, called "hoarding", that allows the user to specify and prioritize tasks that he or she plans to work on during an upcoming disconnection. For example, the user might want to hoard a task for writing a paper or hacking on a program. One aspect of our system that you will test is the interface to this mechanism.

Screen 6: Introduction5

Users have reported that working disconnected can be very successful. However, it does have limitations. The obvious one is that if an object is missing from the cache, the user cannot access it. In addition to this, however, any updates the user makes are not available to colleagues back at work. Furthermore, any updates the user makes are vulnerable to loss (of the laptop machine) or damage (to its disk).

Coda recently introduced "weak connectivity" to help alleviate some of these limitations. Weak connectivity allows the client to communicate with the servers over a "weak" network connection. We define a "weak" network connection as one that is characterized by low bandwidth, high latency, or intermittance. For instance, while attending a conference, you might dialin and connect to the servers over a modem to transmit updates and to service cache misses. Or, while in a meeting or lecture, you might use the recently installed WaveLAN network to operate over a radio frequency network.

Screen 7: Introduction6

The variations in network bandwidth seen by a client of the Coda File System can range over four orders of magnitude. The client might be using a 100 Mb/s Ethernet, a 64 Kb/s Wavelan, or a 24.4 Kb/s modem. As you might imagine, adapting to this kind of variation is difficult.

To make matters worse, some of these network connections actually cost the user money (e.g., long-distance telephone charges). It seems reasonable for the user to expect to have some control over how the network gets used. For instance, the user's preference might be to fulfill cache misses so that she can continue working. On the other hand, the user might prefer to transmit updates back to the server so that those changes are available to her colleagues. To make matters even worse, the user's preferences are unlikely to be static -- they may prefer fetching to transmitting one day and the exact opposite the next. Users need to have the ability to dynamically prioritize those activities requiring network resources.

Futhermore, Coda is a complex system. When things break, it can be difficult (especially for new users) to diagnose the problem. When preparing for a disconnected session, it can be difficult (even for experienced users) to determine what projects are available in the cache and what ones are partially or completely unavailable.

Coda users need an interface to keep them informed about the state of the system and to help them prepare for disconnection. The interface you will be using today is intended to fill this gap.

Screen 8: Introduction7

We have just finished giving you background on the Coda File System. We will now turn our attention to the user interface you will be testing today. The interface is called the "CodaConsole". Portions of this user study have been simulated so that we can control when certain things happen.

Screen 9: Introduction8

Our interface borrows from that of another complex system -- the automobile. The dashboard indicator lights of a car illuminate words or pictures to attract the driver's attention. These lights warn of problems and also indicate system state. For example, if the oil pressure is low, the "OIL" indicator illuminates. Similarly, there are indicators that inform the user that they are running low on fuel, that the car will need preventative maintenance soon, that a car door is open, that the airbags or the antilock breaks are not functioning properly, and so on. Since dashboard indicator lights have been widely adopted and since cars are so prevalent in U.S. society, we expect most of our users will be familiar with this interface.

Although the dashboard indicator lights have been widely adopted, there is plenty of room for improvement. The goal of the Coda indicator lights is to warn users before a problem is so critical that they must simply stop using Coda (whenever possible). In other words, we don't want to have an indicator light that means "pull over to the side of the road 5 minutes ago or risk serious engine damage". Furthermore, we want our indicator lights to be dynamic so that users can explore to get more information than just "something is wrong with the airbags".

Screen 10: Introduction9

We have just requested that the CodaConsole be started. The initialization takes a few seconds. Soon, you will see a small window appear on the right hand side of your screen a few inches below the clock. This window contains our version of indicator lights. There are nine different indicators. The names of our indicators are: Control Panel, Tokens, Space, Network, Advice, Hoard Walk, Reintegration, Repair, and Task. We have purposely made the indicator window as small as possible so that users will be willing to allocate precious screen real estate to keeping the window visible at all times. If users cover this window with other windows, they may not notice if a problem develops.

When the indicator-lights window first appears, the indicators (the text) should all be green. Green means that everything is operating normally. If something were busy or if a problem were brewing, the indicator would reflect that by changing from green to yellow. Similarly, if a critical problem were occuring, the indicator would change to red. We hope these default colors are easy for you to remember and to see. We do, however, understand that people from other cultures and people who have complete or partial color blindness may have difficulty either remembering our color scheme or seeing the actual colors. One of the first things we will do once we get started is to select colors that you can see and remember. Until then, please be patient.

Screen 11: Introduction10

Next, we will begin showing you how to use the interface one indicator at a time. Although some of the tutorial is likely to seem obvious to you, other parts may well be confusing. Please take the time to go through the entire tutorial thoroughly (even the easy parts). It is important that you complete the entire training session prior to beginning work on the exercises.

Note that there is a "Help" button on every screen in the CodaConsole interface (except for the indicator panel itself). The help window that appears when you click on the button provides more information about the screen on which the help button appeared.

Screen 12: Introduction11

First, let us find colors that you can see and remember.

→ Please double-click on the "Control Panel" indicator light. (The arrow means that you need to take some action.)

The "Control Panel" window should soon appear on your screen. You should see a list of folder tabs across the top of the window. The one labelled "Event Configuration" should be visible. Others include "Urgency Colors", "Physical Connectivity", "Logical Connectivity", and "Behavior".

→ Please click on the tab labelled "Urgency Colors".

Screen 13: UrgencyColors1

You should now be looking at the "Urgency Colors" tab of the Control Panel. On this tab, you should see three sections split across the screen. The leftmost section contains a list of color names and shows you how each of them appears when written as text on a black background.

The rightmost section contains a small indicator window showing which colors are chosen for each of the three urgency levels: Normal, Warning, and Critical. By default, the color scheme is green-yellow-red.

The middle section contains three drop-down listboxes that allow you to change the color used for each of the urgency levels. The downward pointing arrow to the right of each color name signifies that a drop-down listbox is present. If you click on this arrow, a listbox will appear. The listbox will display the list of available colors, along with a scrollbar that allows you to navigate in the listbox. To select a new color, simply click on the name of the new color. Please experiment with choosing colors for each urgency level.

[For the record, I agree that this is a silly way to do this. If the widget library that I'm using allowed me to put the left list inside the drop-down listboxes, I'd do it in an instant.]

When you select a new color, you'll notice that the sample indicator panel on the right changes colors, but that the actual indicator lights do not change colors. This sample indicator panel will help you see how your choices will look next to each other. Once you are happy with your choices, you can click on the "Commit" button to have your color choices take effect. (Note that the "Commit" button is probably hidden by this screen.)

We will explore other areas of the Control Panel later in this tutorial so please don't go off and explore on your own just yet.

→ Please choose colors that are easy for you to read and remember.
→ Commit your choices.
→ Please close the "Control Panel" window.

Screen 14: UrgencyColors2

Now we're going to turn our attention to the "Network" indicator. You'll notice that this indicator light has changed to a yellow color. This indicates that there may be a problem. In this case, the problem is that our connection to at least one of the servers is weak (e.g., low bandwidth). Had the color been red instead of yellow, the indicator would have been showing that our connection to at least one server had been severed.

→ Please double click on the "Network" indicator for more detail regarding our connections to individual servers.

This window shows the network bandwidth to each Coda server. The meter to the right of each server's name shows the bandwidth to that server as a percentage of the maximum Ethernet bandwidth, 10 Mb/s. The color of the bar indicates whether we are strongly connected (green), weakly connected (yellow), or disconnected (red) from this server.

Screen 15: Network1

Soon, the bandwidth to `scarlatti` will drop to zero. Watch as the indicator light flashes red and the "Network Information" window automatically updates to show the change in `scarlatti`'s estimated bandwidth.

Screen 16: Network2

We have now completed our exploration of the "Network" indicator.

→ Please close the "Network Information" window.

Next, we will turn our attention to the "Task" indicator.

Screen 17: Network3

One of the user's primary goals in using this interface will be to prepare for disconnected (or weakly connected) operation. To prepare for disconnected operation, the user needs to specify what tasks or projects she plans to work on. Then Venus, the Coda cache manager, makes sure that the files and directories necessary for the user to work on these tasks are available through a process we call "hoarding."

What is a task? A task is simply a set of files and directories necessary for you to work on a particular project. This set contains user data, programs, and even other tasks. For instance, in order for me to work on "writing my dissertation", I need my thesis directory (a user data set), LaTeX (a program), and gnu-emacs (another program).

During this segment of the tutorial, you will learn how to define user data and programs, and how to combine these elements into tasks. In addition, you will learn how to specify what tasks you plan to work on.

→ Please double-click on the "Task" indicator light.

Screen 18: Task1

The "Task" indicator light pops up a window titled "Task Information". This window has three main sections. The first section shows the percentage of cache space occupied. The second section lists the names of all defined tasks. The third section lists those tasks that have been hoarded by the user.

The purpose of the "Task Information" window is to allow the user to select tasks for hoarding. However, before we show you how to do that, we'd like to give you a better understanding of what a task is and how you might define one. Recall that a task is a set of files and directories needed for you to work on a particular project. It contains user data, programs, and even other tasks. Shortly, we will examine each of these components in detail.

→ Please double-click on the "writing thesis" entry of the "Tasks" list. (You may need to use the scroll bar to do this.

Screen 19: Task2

Double-clicking on the name of a task pops up a "Task Definition" window showing the definition of the selected task. This window has three main sections: one for subtasks, one for programs, and one for user data. Each of these three sections is split into two parts. The list on the left side shows a complete listing of all currently defined tasks, programs, or user data respectively. The list on the right side shows those definitions that are included in this task.

Let's look at the last main section---the one labelled "User Data". The list on the left shows the name of every set of user data that has been defined to date (including "advice sources", and "thesis"). The list on the right shows the subset of user data sets that have been included in the "writing thesis" task definition (just "thesis"). The other two sections, the one labelled "Subtasks" and the one labelled "Programs", are similar.

By looking at the three lists on the right, we see that the definition for "writing thesis" contains the "writing" subtask, the programs for "ScreenCapture", and the "thesis" user data.

Screen 20: Task3

In order to view another task definition, double-click on its name in either the "Task Information" window (as you just did) or even in the "Task Definition" window (often more convenient).

Screen 21: Task4

Now, let's look at the user data for the "writing thesis" task in more detail.

→ Please double-click on the "thesis" element in either the predefined user data list or the "writing thesis' contains" list.

Screen 22: Task5

Double-clicking on the name of a user data set pops up a "User Data Definition" window showing the definition of the selected user data. The user data definition contains pathnames to files and/or directories. In the "thesis" definition, you should see three directories:

```
/coda/usr/bovik/thesis/
/coda/usr/bovik/thesis/dissertation/
/coda/usr/bovik/bib/
```

The first of these is the top-level of Harry Bovik's thesis area. The second is the directory in which he stores his dissertation. The third is the top-level of his bibliography directory.

You'll notice an empty entry widget after the bibliography directory. This empty entry is not part of the definition -- it just allows you to continue adding more paths to the definition.

Screen 23: Task6

If the pathname is a directory, the definition also contains some meta-information. The meta-information can be set to "none", "immediate children", or "all descendants".

If the meta-information is set to "none", then only the directory itself will be included. This means that you'd be able to look at the directory contents, but not at any of the children.

If the meta-information is set to "immediate children", then the directory and its immediate children will be included. This means that you'd be able to look at the directory contents and examine any children. If the child is a file, you'd be able to read and modify its contents. If the child is a directory, you'd only be able list its contents---you would not be able to access its children.

If the meta-information is set to "all descendants", then the directory and all its descendants will be included. (Be careful! This could be a very large amount of data!) You'd be able to look at all the children of this directory as described above, but in addition, you'd be able to access the children's children, and so on.

In this definition, all of the pathnames happen to be directories so Harry must consider what meta-information to select. In the case of the thesis area, he knows that this is a deep tree and that hoarding all of it could be a problem. Since he only needs the data relevant to writing, he has elected to hoard the `thesis` directory with the "immediate children" meta-information and the `dissertation` subdirectory with the "all descendants" meta-information option. In the case of the bibliography area, he knows that this is a shallow, small tree. He could choose either the "immediate children" option or the "all descendants" option without any problems. In this case, he has chosen "immediate children".

Screen 24: Task7

Now let's explore how we can define a new user data element.

→ Please click on the black down arrow button. (It's next to the "User Data Name:" entry at the top of the screen.)

You should now see a list of existing user data definitions.

→ Please click on the one named "New...."

Screen 25: Task8

You should now see an empty data definition form, with the name "New..." highlighted. You'll first want to name your definition something other than "New...".

→ Please type the name of a new set of data, perhaps "Grant Proposal" or "Writing SOSP Paper", and then hit <Enter> to move the cursor down into the pathname definition area.

Screen 26: Task9

→ Please type in pathnames (feel free to make some up) that should be included in the new definition.

If the pathname you type is a file, then, when you hit <Enter>, an empty entry widget will appear, allowing you to enter the next pathname. If the pathname you type is a directory (yes, the system does check), then, when you hit either <Tab> or <Enter>, the meta-information area will become enabled, allowing you to choose the appropriate meta-information for this entry. If you used <Enter>, an empty entry widget will also appear and your cursor will be moved down. You'll need to use the mouse to select the appropriate meta-information. If you used <Tab>, you'll be able to use <Tab> and <Enter> to move around and select the meta-information respectively. If you continue to tab after selecting the meta-information, a new entry widget will appear and your cursor will also be moved down.

→ When you have completed the definition, click the "Save" button.

Screen 27: Task10

We have now completed our look into defining user data.

→ Please close all windows titled "User Data Definition".

Screen 28: Task11

Let's now turn our attention to program definitions.

→ Please double-click on the "ScreenCapture" program in the "Task Definition" window.

Double-clicking on the name of a program pops up a "Program Definition" window showing the definition of the selected program. The program definition contains one or more pathnames to executable files. In the case of the "ScreenCapture" definition, you should see four pathnames: "xpr", "xv", "xwdtopnm", and "ppmtogif". (These are the programs that we use to capture screen dumps of the windows in this interface.)

Screen 29: Task12

Even if a program touches other files or directories, or executes other programs, you need not include those pathnames. Once you have indicated an interest in a program, the system will track what files and directories that program requires to run, although the system will ignore any files and directories in areas of the file system that you have defined as user data. By making this distinction between program data and user data, we can let you share program definitions among different tasks. Because the system doesn't start tracking programs until you've indicated an interest in them, you'll need to run the program a few times before disconnecting from the network. Note that because this Interface is still in a mock-up form, you will be unable to do this for any program definitions you create.

Screen 30: Task13

A program definition may contain more than one executable. Each time you hit <Enter> after the last pathname, a new entry widget will appear allowing you to continue adding more paths to the definition. Once again, note that this empty entry is not a part of the program definition.

Screen 31: Task14

Let's explore how to define a new program.

→ Please click on the black down arrow button. (It's next to the "Program Name:" entry at the top of the screen.)

You should now see a list of existing program definitions.

→ Please click on the one named "New...".

You should now see an empty program definition form. You can type the name of the new program definition, "Scribe Tools", and then hit <Enter>. For the purpose of this study, assume that any program you want to include in a definition can be found in /coda/misc/bin.

→ Type in executables that should be included in the new definition
(/coda/misc/bin/scribe and /coda/misc/bin/gv).

→ When you have completed the definition, click the "Save" button.

We have now completed our look into defining programs.

→ Please close all windows titled "Program Definition".

Screen 32: Task15

At this point, you've had some experience with programs and user data. Now it's time to look at combining these with subtasks to define a task. Suppose your advisor has forced you to switch from LaTeX to Scribe for writing your thesis. You would probably first want to change the "writing" task to be called "writing with latex" so that you can continue to use it for other tasks. To do this:

→ Click on the black down arrow button. (It's next to the "Task Names:" entry at the top of the screen.)
→ Select the "writing" task by clicking on it.
→ Type " with latex" after the word "writing".
→ Click the "Save" button.

Screen 33: Task16

Your next goal would be to define the "writing with scribe" task. Now that you've saved the "writing with latex" task, you can modify it to create this new task. (Or you can go up, click the black down arrow, select the task named "New...", and start from scratch.) We'll assume that you'll be modifying this task.

First, we should change the name of this task.

→ Click in the "Task Names:" entry area, fix the name, and hit <Enter>.

Next, we should remove LaTeX and xdvi since we can no longer use those tools.

→ Click on "latex" in the Programs area under the contains list.
→ Hit either the <Backspace> or <Delete> key.
→ Repeat for the "xdvi" entry.

Then, we will want to select the newly defined ScribeTools program definition.

→ Click on the "ScribeTools" in the Programs predefined list.

Finally, we want to save this definition.

→ Click the "Save" button.

Screen 34: Task17

You've just learned about user data, program, and task definitions. Now, let's go back and learn how to select tasks for hoarding.

→ Please close all the windows titled "Task Definition".

You should now be looking at a window titled "Task Information".
If you aren't, you should close all windows other than this one, and
then double-click on the "Task" indicator light.

Screen 35: Task18

Recall that this window allows us to select tasks for hoarding. The "Tasks" area contains a list of all existing tasks. To select a task for hoarding, we use a bizarre form of drag-and-drop---because the toolkit we're using doesn't support real drag-and-drop yet.

→ Click on the "writing thesis" task from this list.
→ Now, move the mouse down to the "Hoarded Tasks" list.
→ Finally, right click anywhere in this list.
→ Repeat for the "hacking advice" and "hacking venus" tasks. If you right click above/-below the name of another task, then the newly selected task will appear before/after that task.

You probably noticed that your machine beeped three times and the task indicator light turned red. The interface has just told you that a task is unavailable. This is because your newly selected tasks have not been hoarded yet. (We'll get to that shortly.)

Screen 36: Task19

Note that each task in the "Hoarded Tasks" list has a meter to the right of its name. This meter tells you how large this task is known to be (as measured in MB -- this number is currently mocked-up data). This meter also tells you what percentage of this task is currently available in the cache. If the meter is green, then the task is available in the cache for use during disconnected or weakly-connected operation. If the meter is red, then the task is partially or completely unavailable.

Also, notice that these two tasks have numbers to the left of their names. These numbers signify their priority during the hoard process. The item numbered 1 will have the highest priority, the one numbered 2 will have the next highest, and so forth. When you drop an item into the "Hoarded Tasks" list, the location at which you right-click in this list determines where the item is dropped and its priority. To change the priority of a hoarded task, you'll need to reselect it from the complete list of tasks and drop it in the correct position in the list of hoarded tasks. Note that each task may appear only once in the list of hoarded tasks. [If your priority numbers appear out of order, please excuse -- there's a bug that's been eluding me.]

Screen 37: Task20

We have now completed our explanation of how to define tasks and how to select them to be hoarded. Next, we'll discuss the "Advice" indicator and then the "Hoard Walk" indicator.

→ Please close the "Task Information" window.

Screen 38: Task21

To make effective use of disconnected operation, the user must define and prioritize tasks that she plans to access while away. The job of managing these tasks is sufficiently difficult that the system needs to provide as much help as possible to the user. Furthermore, the job of adapting to changes in network bandwidth of three or more orders of magnitude is sufficiently difficult that the system occasionally needs advice from the user.

Certain requests for advice require immediate attention. For example, when a user is operating weakly connected (perhaps, over a 9600 baud phone line) and she attempts to access a large file that is not cached, the time required to fetch that file can easily exceed even generous estimates of how long a user might be willing to wait. Rather than automatically performing a fetch that will take a substantial amount of time, we give the user a choice. She can fetch the file or not. Other types of advice are less critical and can be dealt with at the user's convenience. The "Advice" indicator allows the system to offer and request advice, demanding immediate attention only when absolutely necessary.

→ Please double-click on the "Advice" indicator.

Screen 39: Advice1

Notice that you are weakly connected to the Coda servers.

Shortly, the Advice indicator light will change to red indicating that a critical request for advice has arrived (well, been simulated). It will also show up in the "Advice Information" window at that time. This window has two main sections. The top section is a list of advice requests from the client's cache manager, called Venus. The bottom section is a list of advice offered by Venus. Each line is color-coded to indicate its urgency level. To examine a request or offer, the user simply double-clicks on its name.

→ Please double-click on the "Weak Miss" entry, read through the questionnaire, and choose "Don't Fetch".

Screen 40: Advice2

As you just saw, the "Weak Miss" Questionnaire shows the name of the missing object, the program that requested the object, and the estimated amount of time needed to fetch the object. The questionnaire also gives you a choice between fetching the file and not fetching it.

Since a process is waiting for this file request, it is important that this request for advice receive immediate attention from the user. One way to lessen the amount of time the system must wait for this advice (and therefore the amount of time the user must wait for this process to complete execution) is to automatically pop-up the request on the user's screen. We have just configured the system to automatically pop-up future such requests on your screen. (You'll learn how to do this a bit later.)

One complication of asking the user for advice is that she may not be available to answer the request. If that's the case, it might be better to go ahead and fetch the file than to waste valuable time waiting for an answer. Thus, the "Weak Miss" Questionnaire is written to timeout after 15 seconds if you haven't answered the question. At that time, it will ask if you need more time. The idea is that if you are simply thinking about your decision, you'll notice the second question and you can request more time. If you are unavailable, Venus can go ahead and service the demand fetch after waiting for the timeout.

→ Please close the "Advice Information" window.

Screen 41: Advice3

Momentarily, the same weak miss event will occur -- except this time it will automatically pop-up a window on your screen. This time, pretend you are considering your choices. In about 15 seconds, the timeout will expire and you'll see the questionnaire asking if you need more time.

→ Please wait for about 15 seconds.
→ Then select "Yes, please wait for my command".

The interface now shows the same information that was available in the original questionnaire. But it also shows that the interface (and therefore, the requesting process) is waiting for the user's decision.
→ Please make your decision---either way is fine.

Screen 42: Advice4

You've now explored a portion of the "Advice" indicator---the portion that allows Venus to request advice from the user under certain circumstances. It is important for the user to have the ability to influence the use of network resources when those resources become scarce. This facility gives the user that control.

The portion of the "Advice" indicator that we have not explored is the way in which Venus will offer advice to the user. We will not explore this aspect of the interface during this study.

Next, we will discuss the "Hoard Walk" indicator.

Screen 43: Advice5

Note that you are still weakly connected to the Coda servers.

The "Hoard Walk" indicator gives you feedback regarding the status of hoard walks, but it also gives you control over them. A "hoard walk" is the process that ensures that high priority tasks are cached in preference to lower priority tasks that have been accessed more recently. By default, the hoard walk process runs automatically every 10 minutes.

→ Please double-click on the "Hoard Walk" indicator.

You'll notice that this window tells you when the next periodic hoard walk is scheduled to begin. In addition, you can turn periodic hoard walks off, and you can request that a hoard walk begin immediately.

→ Please request that a hoard walk begin immediately.
→ Notice that the "Hoard Walk" indicator turns yellow and that the window changes to show a progress meter that indicates the status of the hoard walk.
→ Then, close the "Hoard Walk Information" window.

Screen 44: HoardWalk1

While weakly connected to the servers, the process that performs hoard walks may request advice before completing its duties. It requests advice when the amount of data it needs to fetch could take a substantial amount of time. The request allows you, the user, to control how much of the data gets fetched.

By default, when Venus requests such advice, the "Hoard Walk" indicator changes from yellow to red. You'll also notice the "Advice" indicator turn red as well. Since this particular form of advice is appropriate to both of these indicators, both of them indicate the request and both of them allow you to get to the advice request window. Let's examine how you can use this interface to control the amount of data fetched over a slow network connection.

→ Please double-click on the "Hoard Walk" indicator once again.

Screen 45: HoardWalk2

You should soon see a window labelled "Hoard Walk Advice". In this window, you should see an area labelled "Cache Status". This area shows you the percentage of cache files occupied (yes, there are a fixed number of cache files) as well as the percentage of cache space occupied (and a fixed number of blocks). You should also see an area labelled "Network Status". This area shows you the average network bandwidth as a percentage of the maximum Ethernet bandwidth. Finally, you should see an area labelled "Fetch Statistics". Ignore this area of the window for now since it is not yet fully implemented.

The main portion of the window displays a list of tasks, the cost of each task, and an indication of whether or not you've requested that this task be fetched. In addition, each task has a button indicating whether or not you've requested that the system stop asking about this task. For instance, portions of a large task may have been updated at the servers. If those portions of the task are not relevant to your current work, you may prefer that the system not update them until you connect over a strong network. Since hoard walks occur periodically, you may want the system to stop asking you about them.

For this segment of the tutorial, we will assume that you have specified some tasks (writing thesis, hacking advice, and hacking venus) that you plan to work on and that the hoard daemon is trying to make sure that those tasks are available in the file cache.

Screen 46: HoardWalk3

The only "task" listed initially is called "ALL Tasks Needing Data". This entry represents all tasks that are not fully cached -- it is not an actual task. To fetch all the necessary files, click on the button under the "Fetch?" label (note the "?"). To prevent Venus from fetching these files now as well as prevent it from requesting advice about any of these files during future hoard walks (until a strong connection has been made), click on the button under the "Stop Asking?" label.

The `+' sign to the left of the task name indicates that this node can be expanded to show more detail. To expand a node, you can either double-click on the name of the node ("ALL Tasks Needing Data"), or single-click on its `+' sign. (You can compress the node in a similar fashion.)

→ Please expand the node named "ALL Tasks Needing Data".

Screen 47: HoardWalk4

Here you see that three tasks need to have files fetched: "Writing Thesis", "Hacking Advice", and "Hacking Venus". Notice that the costs of the three tasks do not add up to the cost of the "ALL Tasks Needing Data" task. This is because some of the tasks require the same files. You can see this fact by expanding each of the nodes. If the name of a task appears in italics, that means that this node appears in more than one location. For example, the "Editing" task appears under each of the three higher-level tasks, and the "Building CODA" task appears under both of the "hacking" tasks. The reason the costs do not add up is because the cost of fetching each of these subtasks is charged only to the first task that requests that it be fetched---the other tasks obtain it for free.

→ Go ahead and expand nodes (e.g. "writing thesis") to see what's there–just don't select anything for fetch yet.

Screen 48: HoardWalk5

→ Please request that all the tasks be fetched by using the "Fetch?" button.
→ Then, click on the "Finish Hoard Walk" button in the lower right corner.

Notice that the "Hoard Walk" indicator turns yellow (the warning color) briefly while the hoard walk finishes its simulated fetching of the requested files.

This completes our overview of the "Hoard Walk" indicator.

→ Please close the window labelled "Hoard Walk Information".

Screen 49: HoardWalk6

Shortly, the space indicator will indicate an urgent problem.

→ Please double-click the "Space" indicator".

You should see that the RVM space is critically full (as indicated by the red color on the meter).

→ Please click on the "Help" button.

The help window will tell you what action (if any) you can take to alleviate this problem. Of course, since this event is simulated, you need not actually take any corrective action.

Screen 50: Space1

The RVM space problem has been solved. Notice that the indicator light reflects this new information and that the "Space Information" window disappears and then reappears displaying the new data.

Screen 51: Space2

Shortly, the space indicator will indicate a potential problem. Again, clicking on the "Help" button will explain the situation and tell you what action you (as the user) should take. Once again this event has been simulated and no action is actually necessary.

Screen 52: Space3

This demonstration of the "Space" indicator light is now complete.

→ Please close the "Space Information" window.

Screen 53: Space4

Let's now explore the "Tokens" indicator. For the purposes of this study, please do not click on the "Destroy Tokens" and "New Tokens" buttons. They work just as you would expect. (If you accidently destroy your tokens, you'll need to come get me.)

→ Please double-click on the word "Tokens" on the indicator panel.

Screen 54: Tokens1

A window titled "Authentication Information" should now be visible on your screen. This window should inform you when your tokens will expire---probably around this time tomorrow. (Coda tokens are similar to Kerberos tickets.)

Screen 55: Tokens2

Our brief exploration of the "Tokens" indicator light is complete.

→ Please close the "Authentication Information" window.

Screen 56: Tokens3

Our final segment of this tutorial shows you how to configure the way in which the CodaConsole alerts you of certain events.

What's an event? An event is simply something that happens while you are using Coda. For example, when your laptop loses connectivity to the Coda file servers, the "Operating Disconnected" event is said to have occurred. Another example of an event is the expiration of your Coda tokens.

What's a configuration? The configuration of an event tells the system what to do when a particular event occurs. For instance, do you want your machine to flash (blink) the appropriate indicator at you? Would you prefer that it NOT beep at you? Do you even want to be notified of this particular event?

→ Please double-click on the "Control Panel" indicator light.

Screen 57: EventConfiguration1

You should now be looking at the "Event Configuration" tab of the Control Panel. This tab has three sections. The first section (in the upper left region of the screen) contains the name of the indicator light for which we are configuring events. You can select a different indicator light by using the drop-down listbox. The second section (in the lower left region of the screen) contains the name of an event and its description. You can select a different event by using the drop-down listbox. The third section (on the right side of the screen) contains a number of configuration options described below.

The first option configures whether or not you will be notified if this event occurs. If you choose to be notified, then the remaining options can be configured to specify how you would like to be notified. For each event, you can choose how urgent you believe it to be, whether or not an information window should automatically appear on your screen, whether or not the bell should be used to alert you, and whether or not the indicator light should flash (blink) on and off to alert you.

The configuration of the "Tokens Acquired" event is shown. This event is currently configured to notify you. It has a normal urgency value (which simply means that it will use the "Normal" color you recently chose under the "Urgency Colors" tab). It does not pop-up an information window, it does not ring the bell, and it does not flash the indicator light.

Screen 58: EventConfiguration2

→ Under the "Advice" indicator, please select the "Read Disconnected Cache Miss" event.

Notice that this event is not configured to notify the user; therefore, the notification options are all disabled.

Please examine how the "Operating Disconnected" event (under the "Network" indicator) is configured. How would you describe this event and its configuration? The next screen gives the answer for you to verify your understanding of event configurations.

Screen 59: EventConfiguration3

The "Operating Disconnected" event is configured to notify the user with a critical urgency level by flashing the "Network" indicator light but not beeping or popping up a window.

Screen 60: EventConfiguration4

Because the other three tabs on the control panel, labelled "Physical Connectivity", "Logical Connectivity", and "Behavior", are irrelevant to this user study, we will not demonstrate their use.

→ Please close the "Control Panel" window.

Screen 61: EventConfiguration5

Oops! The tokens indicator shows that your tokens have expired. (Yes, this too is simulated.) This brings our tutorial to a close.

You may have noticed that we didn't discuss either the "Reintegration" or the "Repair" indicator lights. These two indicator lights have not yet been implemented. (Coming soon to an interface near you...)

The training session is now over. You can now take a 10 minute break. (Restrooms and a drinking fountain are available behind the elevator block. Vending machines are available on the 2nd floor.) When you return, we have a few exercises for you to perform using the interface. We expect these exercises to take approximately one hour.

Screen 62: TokenExpiry

B.5 Exercises

Order of Events:

1. Start CodaConsole.
2. Display Initial.
3. Store "BeginTime".
4. Display ExercisesIntro.
5. Begin logging user actions.
6. Store Begin Time for "Phase One".
7. Display ChooseUrgencyColors.
8. Display IdentifyCurrentNetworkConditions.
9. Display DescribeOneOrMoreTasksUnavailable.
10. Display FixOneOrMoreTasksUnavailable.
11. Display WhenDoTokensExpire.
12. Display DescribeTaskHoardingDetails.
13. Trigger Hoard Walk Advice Request (time for the near future).
14. Display ProvideHoardWalkAdvice.
15. Save Hoard Walk Advice results.
16. Display DescribeSpaceStatus.
17. Display DescribeMissEvents.
18. Display FixMissEvents.
19. Store End Time for "Phase One".
20. Store Begin Time for "Phase Two".
21. Display ReadBlizzardBackground.
22. Display DefineBugTask.
23. Display DefineHeaderFilesTask.
24. Display DefineCVTask.
25. Display PrioritizeTasks.
26. Display RequestImmediateHoardWalk.
27. Store End Time for "Phase Two".
28. Store Begin Time for "Phase Three".
29. Display ReadIncidentBackground.
30. Display ReadGameBackground.
31. Display RunGame.
32. Create "Incident Report" Button.
33. Configure Events for Phase Three.
34. At T+100, trigger RVM to fill to a critical level.
35. At T+220, trigger RVM to return to normal.
36. At T+300, trigger a task to become unavailable.
37. At T+398, trigger a hoard walk to begin.
38. At T+403, trigger the hoard walk to complete.
39. At T+528, trigger weakly connected operation.
40. At T+643, trigger a cache miss.
41. At T+740, trigger disconnected operation.
42. At T+848, trigger a cache miss.
43. At T+942, trigger strongly connected operation.
44. At T+1031, trigger token expiry.
45. At T+1143, trigger a reintegration.
46. At T+1200, display Final.

47. Store End Time for Phase Three.
48. Store "EndTime".

Please wait to begin.

Screen 63: Initial

During this portion of the user study, you will be presented with some exercises. The first set of exercises will ask you to use the interface to answer a few questions. The second set of exercises will ask you to prepare for a disconnected session. The final set will ask you to play a game while a background process accesses data in Coda. Remember, these exercises are for us to observe a ``typical" user of our system. If you have any problems with these exercises, it reflects poorly on our interface -- not on you! Also, please try to think aloud while you perform these activities. Just as during the Tutorial segment of this study, the windows associated with the Exercises segment have a pink background. This should help you distinguish between the interface of the study and that of the CodaConsole. We would like to apologize for the fact that you cannot cut and paste between the CodaConsole interface and this one. Since cut and paste is only really useful for the purposes of this study, we didn't implement it. However, cutting text from an xterm and pasting it into either interface should work just fine. Do you have any questions before we begin?

Screen 64: ExercisesIntro

Please choose colors for the urgency levels. Pick colors that you can see and remember. (If you are happy with the default colors, you need not pick new ones.

Screen 65: ChooseUrgencyColors

Please describe the current network conditions. What is our connectivity to each of the servers: strongly connected, weakly connected, or disconnected? Which server currently has the best nework connectivity? Which one has the worst?

Screen 66: IdentifyCurrentNetworkConditions

Under what conditions will the ``One Or More Tasks Unavailable" event occur? Does the interface notify the user of this event? At what urgency level? Will it beep at the user? Will it pop-up a window? Will it flash the indicator?

Screen 67: DescribeOneOrMoreTasksUnavailable

Since this is an important event, you want to be sure not to miss it. Please fix the settings for this event (``One Or More Tasks Unavailable'') so that it flashes and beeps, but does not pop-up.

Screen 68: FixOneOrMoreTasksUnavailable

At what time do your tokens expire?

Screen 69: WhenDoTokensExpire

What tasks are currently defined?
Which of these are hoarded?
Which of these tasks has the highest priority?
The lowest?
Which, if any, are completely available?

Screen 70: DescribeTaskHoardingDetails

Now, suppose you are attending a PI Workshop (attendance required to continue to receive grants from this particular funding agency) in Nowheresville, IA. You had to arrive a day and a half early to get cheap tickets and now you're bored to tears. Thankfully, you have plenty of work to do while you wait, but unfortunately, you were unable to hoard prior to leaving for the airport.

You need to hoard two tasks at this time. First, a colleague asked you to preview a submission for SOSP16. You need to pick up the latest version and read it so that you can send comments back to the authors. If you have time, you'd also like to work on a few letters of recommendation. These two tasks have already been defined (``SOSP16 Paper'' and ``Letters of Recommendation'').

You have selected two tasks to be hoarded and have recently made a connection via an (expensive) long-distance telephone line. Just a minute ago, you requested that a hoard walk be performed. Now, you notice that the hoard daemon has requested advice. Please examine the request, decide what objects should be fetched at this time, and describe your reasoning below.

Screen 71: ProvideHoardWalkAdvice

Please describe the current space status.
How much disk space is dedicated to the Coda cache?
What percentage of the RVM space is utilized?

Screen 72: DescribeSpaceStatus

Please describe the ``Weakly Connected Cache Miss Advice'' and the ``Disconnected Cache Miss Advice'' events and their configurations in the space below.

Screen 73: DescribeMissEvents

Please fix the settings for these events (``Weakly Connected Cache Miss Advice" and ``Disconnected Cache Miss Advice") to be consistent. I'd like them to be indicated at the ``critical" urgency level such that they pop-up and flash the indicator, but do not beep.

Screen 74: FixMissEvents

You have now completed the first set of exercises. In the second set of exercises, you will be asked to prepare to operate disconnected for a period of a few days.

A blizzard is presently hitting Ohio. It has a history of dumping large amounts of snow (more than 3 feet in parts of Ohio so far). The National Weather Service in Pittsburgh expects the storm to hit southwestern Pennsylvania late tonight or early tomorrow morning. Roads are expected to be treacherous, and possibly closed to all but emergency vehicles.

You have lots of work to do prior to commencing your job search, so you plan to take your laptop home and work from there disconnected. You will now define a number of tasks that you would like to work on during the blizzard. (Don't forget -- you can reuse data, program, and task definitions!) Once you have defined these tasks, we will ask you to hoard them and choose appropriate priorities. Please do NOT prioritize them before that time.

As you define these tasks, we ask you to keep three things in mind. First, pretend that the machine you are running on is an i386_mach box. Second, you may assume that all executables that can be found in /coda/misc/bin (though any predefined programs listing a different path do not need to be changed). Third, because this is a mock-up, you will not be able to test your hoard profiles or run the programs. These notes are available on the card taped to the desk for your reference.

Screen 75: ReadBlizzardBackground

You are behind on addressing recent bugs reported by users. You'd like to take this occasion to catch up. You'll need the Coda sources, the bug reports, and the build environment.

The sources to the code that you must modify are located in three places:
/coda/usr/bovik/src/venus/
/coda/usr/bovik/src/advice/
/coda/usr/bovik/src/console/

You've squirreled the original reports away in
/coda/usr/bovik/src/bugs/

The programs you need to build Coda have been defined as ``building coda".
The last time you hacked disconnected, you realized that you had forgotten to include gdb in your building Coda definition. It can be found in
/coda/misc/gnu-comp/i386_mach/omega/bin/gdb

Please create a task for fixing Coda bugs.

Screen 76: DefineBugTask

Oh, and the last time you hacked from home, you wanted to take a look at some header files that you had forgotten to hoard. They aren't really important, but it'd be nice to have them around. Unfortunately, even though each individual header file is relatively small, together they take up quite a lot of space -- so you don't want to just add them to your ``building coda'' programs since then they might be hoarded in preference to really important source files. They are located in several places:

```
/coda/project/coda/alpha/include/
/coda/project/coda/alpha/i386\_mach/include-special/
/coda/misc/c++/i386\_mach/alpha/include/
/coda/misc/tcl/common/beta/include/
```

Screen 77: DefineHeaderFilesTask

You need to write your CV before Tuesday when you will be talking to a representative from AT\&T Labs (assuming, of course, that the visitor can get here and that you can get to school). Since you haven't yet written your CV, you've borrowed the file for your advisor's CV to use as an example. Unfortunately, he still uses Scribe! Since you're time-pressured, you've decided to stick with Scribe. Please create a task for writing your CV. The necessary files are located in:

```
/coda/usr/bovik/personal/cv/
```

Unlike LaTex, Scribe outputs PostScript directly. To run Scribe, type ``scribe cv.mss''; it will produce a ``cv.ps'' file that you can preview with ``gv''. You may assume that the binaries for both scribe and gv are located in /coda/misc/bin.

Screen 78: DefineCVTask

Finally, you'll need to prioritize these tasks. Recognize that the data needed for all these tasks may or may not fit into your cache -- the priorities you choose now will decide what (if anything) gets left behind. To help you in prioritizing these tasks, we've listed all of the tasks you would like to work on during the blizzard, as well as what they are intended for. (The list below is in no particular priority order.) If you haven't already hoarded these tasks, please do so now. When you have finished, please use the space below to describe your reasons for choosing the priorities you did.

Fix bugs reported by users.
Various header files to support bug fixing.
Write your CV to give to this visitor on Tuesday.

Screen 79: PrioritizeTasks

Please request that a hoard walk be performed immediately.

Screen 80: RequestImmediateHoardWalk

During the final segment of this study, we will be simulating a more realistic usage scenario. (This final segment will take about 20 minutes.)

Please imagine that you are running your thesis through latex (or latex2e) in preparation for printing a copy of the document. This process takes a fairly long time since the document is rather lengthy and since latex requires you to run the document through twice (to update cross-references). For this reason, it is necessary for you to stick around and babysit it. Luckily, you're addicted to the computer game tetris...

As you are playing your game, you may notice problems or state changes being indicated through the indicator light interface. When this happens, you'll need to pause the game (use "p") and then click on the button labelled "Incident Report" (not yet visible) to describe the problem or state change to us on the pop-up form. Please report ALL state changes, even those changing from something bad to something good.

Note that the window containing the "Incident Report"' button will replace the window you are currently looking at until this segment of the study is completed. Please continue to play tetris throughout this segment of the study, even if it means starting a new game.

Screen 81: ReadIncidentBackground

Here are some excerpts taken from the tetris manual page:

The object of the game is to fit the shapes together forming complete rows, which then vanish. When the shapes fill up to the top, the game ends.

At the start of the game, a shape will appear at the top of the screen, falling one square at a time. When this shape ``touches down" on the bottom of the field, another will appear at the top. As the shape falls, you can move it to the left or right, rotate it counterclockwise, or drop it to the bottom by pressing the appropriate keys. As you fit shapes together, completed horizontal rows vanish, and any blocks above fall down to fill in. When the blocks stack up to the top of the screen, the game is over.

The default control keys are as follows: ``j" moves the block to the left; ``l" moves the block to the right; ``k" rotates the block counterclockwise; `` " causes the block to drop; ``p" causes the game to pause; and ``q" quits the game. (These key bindings appear on the game screen to remind you.)

If you have any questions about the game, please feel free to ask.

Screen 82: ReadGameBackground

Let the game begin!

→ Please execute tetris in the xterm.
→ Please click the ``ok" button below and then begin playing tetris.

Screen 83: RunGame

B.6 Evaluation Questionnaire

This questionnaire is designed to tell us about your experience with interface you used today. Please circle the number that most clearly expresses how you feel about each statement. Write in any comments you have below each question. If you need more room, please ask for a blank sheet of paper.

1. The following questions are related to the tutorial.

The tutorial coverage was:

1	2	3	4	5
very incomplete	incomplete	neither complete nor incomplete	complete	very complete

Comments: _____

b. Understanding the tutorial was:

1	2	3	4	5
very difficult	difficult	neither easy nor difficult	easy	very easy

Comments: _____

c. Did the tutorial allow you to grasp the scope of the capabilities of the interface?

yes
no

Comments: _____

d. Was the tutorial presented at an appropriate level of detail? (Circle all that apply.)

not appropriate, too little detail
appropriate amount of detail
not appropriate, too much detail

e. If you answered (i) or (iii) above, please provide one or more examples of where the level of detail was inappropriate.

2. The following questions relate to the indicator light interface.

a. Using this interface was:

1	2	3	4	5
very difficult	difficult	neither easy nor difficult	easy	very easy

Comments: _____

b. Finding the features I wanted in the indicator lights was:

1	2	3	4	5
very difficult	difficult	neither easy nor difficult	easy	very easy

Comments: _____

c. If you were a Coda user, how likely or unlikely would you be to keep the indicator lights visible on your screen all (or almost all) the time?

1	2	3	4	5
very unlikely	unlikely	neither likely nor unlikely	likely	very likely

Comments: _____

d. Did the interface present information at an appropriate level of detail? (Circle all that apply.)

not appropriate, too little detail
appropriate amount of detail
not appropriate, too much detail

e. If you answered (i) or (iii) above, please provide one or more examples of where the level of detail was inappropriate.

3. The following questions relate to hoarding, both in general and specifically in terms of your preparation for operating disconnected for the duration of the *blizzard*.

   a. Understanding the concept of hoarding was:

1	2	3	4	5
very difficult	difficult	neither easy nor difficult	easy	very easy

   Comments: _____

   b. Preparing for the *blizzard* disconnection was:

1	2	3	4	5
very difficult	difficult	neither easy nor difficult	easy	very easy

   Comments: _____

   c. My confidence in the task definitions I created in preparation for the *blizzard* is:

1	2	3	4	5
very low	low	neither high nor low	high	very high

   Comments: _____

4. As we mentioned in the tutorial, our interface is adapted from the dashboard indicator lights of a car. The following questions are related to your experiences with these indicator lights.

a. Do you prefer to drive a car that uses dashboard indicator lights to one that does not?

yes, I prefer a car *with* dashboard indicators lights.

no, I prefer a car *without* dashboard indicator lights.

Comments: _____

b. Dashboard indicator lights are:

1	2	3	4	5
very ineffective	ineffective	neither effective nor ineffective	effective	very effective

Comments: _____

c. Please describe an instance of when you used the dashboard indicator lights in the last month.

d. Please describe a particularly memorable event when you found the dashboard indicator lights to be either particularly effective or particularly ineffective.

5. a. If you are not currently a Coda user, would you be interested in becoming a Coda user?

yes
no
Comments: _____

b. If you are currently (or have been) a Coda user, would you be interested in using this interface?

yes
no
Comments: _____

6. a. Please describe the aspect of the interface that you liked most.

b. Please describe the aspect of the interface that you liked least.

Thank you for your time!

B.7 Answer Key to Exercises

This key defines the correct answer to each of the exercises in our usability study. Each bullet scores one *point*.

1. [3 pts] Please describe the current network conditions.
    - connectivity? strongly connected to all
    - best? haydn
    - worst? grieg
2. [6 pts] Describe the "One or more tasks unavailable" event and its configuration.
    - Triggered when one or more hoarded tasks become unavailable.
    - Notify – YES
    - Urgency level – CRITICAL
    - Pop-up – NO
    - Beep – NO
    - Flash – NO
3. [2 pts] Fix the settings for this event so it does not pop-up, but it does beep and flash.
    - Toggle beep.
    - Toggle flash

    ➔ Event:OneOrMoreTasksUnavailable:C:1:0:1:1
4. [1 pt] At what time do your tokens expire?
    - Tomorrow at about …
5. [12 pts] Please describe the hoarding details.
    a) What tasks are currently defined?
        - darpa grant
        - editing
        - recommendation letters
        - sosp16
        - writing

    b) Which of these are hoarded?
        - sosp16
        - recommendation letters
        - darpa grant
        - editing

    c) Which task has the highest priority?
        - sosp16

    d) What task has the lowest priority?
        - editing

    e) Which, if any, are completely available?
        - editing

6. [2 pts] Please provide hoard walk advice and describe your reasoning.
    - Only need "papers—sosp16" and "letters—recommendation", not editing
    - Don't need "darpa grant"
7. [3 pts] Please describe the current space status.
    - Status? All space areas are within normal parameters
    - Space dedicated to Coda cache? 97 MB
    - Percentage of RVM utilized? 33%
8. [8 pts] Describe the "Weakly Connected Cache Miss" and the "Disconnected Cache Miss" events and their configurations.
    a) Weakly Connected Cache Miss
        - Triggered by a cache miss while operating weakly connected
        - Notifies the user
        - Critical urgency level
        - pop-up – NO
        - flash – YES
        - beep – YES
    b) Disconnected Cache Miss
        - Triggered by a cache miss while operating disconnected
        - Does not notify the user
9. [4 pts] Fix the settings for these events to be consistent. I'd like them to be indicated at the "critical" urgency level such that they pop-up and flash the indicator, but that they do not beep.
    a) Weak Miss:
        - Toggle pop-up.
        - Toggle beep.
            → WeaklyConnectedCacheMiss:C:1:1:0:1
    b) Disco Miss:
        - Toggle notify.
        - Toggle flash
            → DisconnectedCacheMiss:C:1:1:0:1
10. [9 pts] Define a task for fixing bug reports.
    a) Data:
        - Coda sources [3 areas]
            - /coda/usr/bovik/src/venus/ with Meta: c or d
            - /coda/usr/bovik/src/advice/ with Meta: c or d
            - /coda/usr/bovik/src/console/ with Meta: c or d
        - [Bug reports]
            - /coda/usr/bovik/src/bugs/ with Meta: c or d
    b) Programs:
        - building coda
            - Add /coda/misc/gnu-comp/i386_mach/omega/bin/gdb
        - compress-related
            - [compress, uncompress, gzip, gunzip]
        - [tcl testing]
            - [wish]

c) Subtasks:
- editing

11. [5 pts] Define a task for miscellaneous header files.
    a) Data
    - Header Files
        - /coda/project/coda/alpha/include/ with Meta: c or d
        - /coda/project/coda/alpha/i386_mach/include-special/ with Meta: c or d
        - /coda/misc/tcl/common/beta/include/ with Meta: c or d
        - /coda/misc/c++/i386_mach/alpha/include/ with Meta: c or d

12. [6 pts] Define a task for your CV
    a) Data:
    - CV
        - /coda/usr/bovik/personal/cv/ with Meta: c or d

    b) Programs:
    - scribe tools or scribe program
        - scribe
    - [gv program]
        - gv

    c) Subtasks
    - editing

13. [2 pts] Prioritize these tasks and describe your reasoning.
    - [editing can be anywhere (even first or last) or nowhere]
    - First task must be consistent with user's reasoning – otherwise CV.
    - header files must be last

14. [1 pt] Request that a hoard walk be performed immediately.
15. [3 pts] Report "Filling RVM" event.
    - Urgency Level: Critical
    - Problem: The RVM space is almost full.
    - Solution: Contact my system administrator.

16. [2 pts] Report "Fixing RVM" event.
    - Urgency Level: Normal
    - Problem: The RVM space problem has fixed itself.
    - [Solution: NA]

17. [3 pts] Task Unavailable
    - Urgency Level: Critical
    - Problem: A task has become unavailable.
    - Solution: Perform a hoard walk.

18. [2 pts] Hoard Walk Begin event.
    - Urgency Level: Warning
    - Problem: A hoard walk has begun.
    - Solution: Wait until it completes or NA

19. [2 pts] Hoard Walk End & Task Available event
    - Urgency Level: Normal
    - Problem: Nothing, a hoard walk completed; task available again.
    - Solution: NA

20. [2 pts] Report "Operating Weakly Connected" event.
    - Urgency Level: Warning
    - Problem: We are now operating weakly-connected with all servers.
    - Solution: NA (or connect to a strong network if possible)
21. [2 pts] Report "Weak Miss" event.
    - Urgency Level: Critical
    - Problem:
        - We have experienced a cache miss while weakly connected
        - [Filename: intro.tex]
        - [Requesting program: latex2e]
        - [Expected Fetch Time: 23+ minutes]
        - *Note: Since answer does not require filename, user does not lose a point if they specify the wrong one. Same goes for program and time.*
    - Solution:
        - [Decide to fetch or not; find a better network.]
22. [2 pts] Report "Operating Disconnected" event.
    - Urgency Level: Critical
    - Problem: We are now operating disconnected from the Coda servers.
    - Solution: NA (or connect over a weak or strong network if possible).
23. [2 pts] Report "Disconnected Miss" event.
    - Urgency Level: Critical
    - Problem:
        - We have experienced a disconnected cache miss.
        - [Filename: intro.tex]
        - [Requesting Program: latex2e]
    - Solution:
        - [Answer the questionnaire and reconnect to the servers ASAP.]
24. [2 pts] Report "Operating Strongly Connected" event.
    - Urgency Level: Normal
    - Problem: Nothing, we have reestablished a strong network connection to the Coda servers.
    - Solution: NA
25. [3 pts] Report "Token Expiry" event.
    - Urgency Level: Warning
    - Problem: My authentication tokens have expired.
    - Solution: Reauthenticate
26. [3 pts] Report "Reintegration Pending Tokens" event.
    - Urgency Level: Critical
    - Problem: A subtree needs to be reintegrated and my tokens are invalid.
    - Solution: Reauthenticate

B.8 Par to Usability Study Exercises

Each of the exercises in our usability study has been given a par score. *Par* is defined as the number of actions necessary for an expert user with foreknowledge of the questions to complete the exercise. Actions are preceded by an arrow symbol. Output lines produced by the interface and taken from the action log are preceded by a bullet symbol. Actions in square brackets represent optional actions that are not counted against the user. Consecutive scrolling events are compressed into a single "scroll" action.

1) [par 0] ChooseColors.

⇒ [Select Control Panel Indicator Light.]

- `Control Panel Indicator: double-click`
- `Control Panel: Click on Events tab`
- `Control Panel Events Tab: Selecting Tokens indicator`
- `Control Panel Events Tab: Selecting Tokens Acquired event`
- `Control Panel Events Tab: Selecting Tokens indicator`
- `Control Panel Events Tab: Selecting Tokens Acquired event`

⇒ [Select "Urgency Colors" Tab.]

- `NoteBookWindow:Active:urgency`
- `Control Panel: Click on Urgency tab`
- `Control Panel Urgency Colors Tab: Changing Normal to ForestGreen`
- `Control Panel Urgency Colors Tab: Changing Warning to Gold`
- `Control Panel Urgency Colors Tab: Changing Critical to OrangeRed1`

⇒ [Choose color for "Normal" urgency.]

- `Tix Combo Arrow: click`
- `Control Panel Urgency Colors Tab: Changing Normal to DarkGreen`

⇒ [Choose color for "Warning" urgency.]

- `Tix Combo Arrow: click`
- `Control Panel Urgency Colors Tab: Changing Warning to Yellow`

⇒ [Choose color for "Critical" urgency.]

- `Tix Combo Arrow: click`
- `Control Panel Urgency Colors Tab: Changing Critical to DarkOrange1`

⇒ [Commit color choices.]

- `Control Panel Urgency Tab: Click on Commit Button`

⇒ [Close Control Panel Window.]

- `Control Panel Window: close`
- `Control Panel Urgency Tab: Click on Commit Button`
- `Control Panel Physical Tab: Click on Commit Button`
- `Control Panel Logical Tab: Click on Commit Button`
- `Control Panel Behavior Tab: Click on Commit Button`

2) [par 2] Describe Network Conditions.
   ⇒ Select Network Indicator.
   - `Network Indicator: double-click`

   ⇒ Close Network Information Window.
   - `Network Window: close`

3) [par 3] Describe "One Or More Tasks Unavailable" event and its configuration.
   ⇒ Select Control Panel Indicator.
   - `Control Panel Indicator: double-click`
   - `Control Panel: Click on Events tab`
   - `Control Panel Events Tab: Selecting Tokens indicator`
   - `Control Panel Events Tab: Selecting Tokens Acquired event`
   - `Control Panel Events Tab: Selecting Tokens indicator`
   - `Control Panel Events Tab: Selecting Tokens Acquired event`

   ⇒ Select Task indicator from combobox.
   - `Tix Combo Arrow: click`
   - `Control Panel Events Tab: Selecting Task indicator`
   - `Control Panel Events Tab: Selecting All Tasks Available event`

   ⇒ Select "OneOrMoreTasksUnavailable" from combobox.
   - `Tix Combo Arrow: click`
   - `Control Panel Events Tab: Selecting One Or More Tasks Unavailable event`

4) [par 4] Fix this event.
   ⇒ Toggle Beep.
   - `Control Panel Events Tab: Setting Notification for Notify:Beep to Yes`

   ⇒ Toggle Flash.
   - `Control Panel Events Tab: Setting Notification for Notify:Flash to Yes`

   ⇒ Commit Changes.
   - `Control Panel Events Tab: Click on Commit Button`

   ⇒ Close Control Panel Window.
   - `Control Panel Window: close`
   - `Control Panel Urgency Tab: Click on Commit Button`
   - `Control Panel Physical Tab: Click on Commit Button`
   - `Control Panel Logical Tab: Click on Commit Button`
   - `Control Panel Behavior Tab: Click on Commit Button`

5) [par 2] Identify when tokens expire.
   ⇒ Select Tokens Indicator Light.
   - `Tokens Indicator: double-click`

   ⇒ Close "Authentication Information" Window.
   - `Tokens Window: close`

6) [par 4] Describe task hoarding details.
   ⇒ Select Task Indicator Light.
   - `Task Indicator: double-click`

   ⇒ Scroll on task list.
   - `[TaskInformationWindow:scroll:.task.panes.allTasks.f.border.frame.taskl ist]*`

⇒ Scroll on selected task list.
- `[TaskInformationWindow:scroll:.task.panes.selectedTasks.f.border.frame.tasklist]*`

⇒ Close "Task Information" window.
- `Task Window: close`

7) [par 8] Provide Hoard Walk Advice.

⇒ Select Hoard Walk Indicator Light.
- `Hoard Walk Indicator: double-click`

⇒ Expand "ALL Tasks Needing Data" node.
- `HList: Select`
- `Hoard Walk Advice Window: expand node`

⇒ Expand "sosp16" node.
- `HList: Select`
- `Hoard Walk Advice Window: expand node`

⇒ Expand "recommendation letters" node.
- `HList: Select`
- `HList: Select`
- `HList: Select`
- `Hoard Walk Advice Window: expand node`

⇒ Select "papers – sosp16" to be fetched.
- `HoardWalkPendingAdviceWindow:fetch`

⇒ Select "letters – recommendation" to be fetched.
- `HoardWalkPendingAdviceWindow:fetch`

⇒ Trigger hoard walk to continue.
- `Hoard Walk Window: fetch`

⇒ Close "Hoard Walk Advice" window.
- `Hoard Walk Window: close`

8) [par 2] Describe Space Status.

⇒ Select Space Indicator Light.
- `Space Indicator: double-click`

⇒ Close "Space Information" Window.
- `Space Window: close`

9) [par 3] Describe "Weakly Connected Cache Miss Advice" and "Disconnected Cache Miss Advice" events and their configurations.

⇒ Select Control Panel Indicator Light.
- `Control Panel Indicator: double-click`
- `Control Panel: Click on Events tab`
- `Control Panel Events Tab: Selecting Tokens indicator`
- `Control Panel Events Tab: Selecting Tokens Acquired event`
- `Control Panel Events Tab: Selecting Tokens indicator`
- `Control Panel Events Tab: Selecting Tokens Acquired event`

⇒ Select "Advice" Indicator.
- `Tix Combo Arrow: click`
- `Control Panel Events Tab: Selecting Advice indicator`
- `Control Panel Events Tab: Selecting Weakly Connected Cache Miss event`

⇒ Select "Disconnected Cache Miss Advice" event.
- `Tix Combo Arrow: click`
- `Control Panel Events Tab: Selecting Disconnected Cache Miss event`

10) [par 8] Fix these configurations so that they pop-up and flash, but do not beep.

⇒ Toggle Notify.
- `Control Panel Events Tab: Setting Notification for Notify to Yes`

⇒ Toggle Flash.
- `Control Panel Events Tab: Setting Notification for Notify:Flash to Yes`

⇒ Commit Changes.
- `Control Panel Events Tab: Click on Commit Button`

⇒ Select "Weakly Connected Cache Miss Advice" event.
- `Tix Combo Arrow: click`
- `Control Panel Events Tab: Selecting Weakly Connected Cache Miss event`

⇒ Toggle Beep.
- `Control Panel Events Tab: Setting Notification for Notify:Beep to No`

⇒ Toggle Pop-up.
- `Control Panel Events Tab: Setting Notification for Notify:Popup to Yes`

⇒ Commit Changes.
- `Control Panel Events Tab: Click on Commit Button`

⇒ Close Control Panel Window.
- `Control Panel Window: close`
- `Control Panel Urgency Tab: Click on Commit Button`
- `Control Panel Physical Tab: Click on Commit Button`
- `Control Panel Logical Tab: Click on Commit Button`
- `Control Panel Behavior Tab: Click on Commit Button`

11) [par 36] Define a task to hoard for fixing Coda bugs.

⇒ Select Task Indicator Light.
- `Task Indicator: double-click`

⇒ Select a Task.
- `HList: Select`
- `Task Definition Window (ID=0): open`
- `Task Definition Window: Show New...`

⇒ Select "New..." Data Definition.
- `Task Definition Window: Select New... from Choices Listbox`
- `Data Definition Window (ID=0): open`
- `Data Definition Window: Show New...`

⇒ Type name and hit <Return>.
- `Data Definition Window: Show coda sources`

⇒ Type directory pathname and hit <Tab>.
- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.
- `Data Definition Window: Meta-Information Select Descendants`

⇒ Type directory pathname and hit <Tab>.
- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.
- `Data Definition Window: Meta-Information Select Descendants`

⇒ Type directory pathname and hits <Tab>.
- Data Element: <Tab> enables meta-information

⇒ Select meta-information.
- Data Definition Window: Meta-Information Select Descendants

⇒ Save new definition.
- Data Definition Window: save
- Data Definition Window: save as
- Data Definition Window: save
- Data Definition Window: Show coda sources

⇒ Select "New..." data definition from the combobox.
- Tix Combo Arrow: click
- Data Definition Window: Show New...

⇒ Type name and hit <Return>
- Data Definition Window: Show coda bug reports

⇒ Type directory pathname and hit <Tab>.
- Data Element: <Tab> enables meta-information

⇒ Select meta-information.
- Data Definition Window: Meta-Information Select Descendants

⇒ Save new definition.
- Data Definition Window: save
- Data Definition Window: save as
- Data Definition Window: save
- Data Definition Window: Show coda bug reports

⇒ Close data definition window.
- DataDefinitionWindow(ID=0):beginclose
- Data Definition Window: save
- DataDefinitionWindow(ID=0):endclose

⇒ Select "building coda" Program Definition.
- Task Definition Window: Select building coda from Choices Listbox
- Program Definition Window (ID=0): open
- Program Definition Window: Show building coda

⇒ Click in executable entry area.
- Program Definition Element: click

⇒ Type name of executable.
- Program Element: <Return> begin
- Program Element: <Return> end

⇒ Save program definition.
- Program Definition Window: save

⇒ Select "New..." program definition from the combobox.
- Tix Combo Arrow: click
- Program Definition Window: Show New...

⇒ Type name and hit <Return>.
- Program Definition Window: Show compress related

⇒ Type name of executable.
- Program Element: <Return> begin
- Program Element: <Return> end

⇒ Type name of executable.
- `Program Element: <Return> begin`
- `Program Element: <Return> end`

⇒ Save program definition.
- `Program Definition Window: save`
- `Program Definition Window: save as`
- `Program Definition Window: save`
- `Program Definition Window: Show compress related`

⇒ [Selects "New..." program definition from the combobox.]
- `Tix Combo Arrow: click`
- `Program Definition Window: Show New...`

⇒ [Type name and hit <Return>.]
- `Program Definition Window: Show tcl testing`

⇒ [Type name of executable.]
- `Program Element: <Return> begin`
- `Program Element: <Return> end`

⇒ [Save program definition.]
- `Program Definition Window: save`
- `Program Definition Window: save as`
- `Program Definition Window: save`
- `Program Definition Window: Show tcl testing`

⇒ [Close program definition window.]
- `ProgramDefinitionWindow(ID=0):beginclose`
- `Program Definition Window: save`
- `ProgramDefinitionWindow(0):endclose`

⇒ Click in entry area.
- `Tix Combo Entry: click`

⇒ Type name of new task and hit <Return>.
- `Task Definition Window: Show Coda bug fixing`

⇒ Select element.
- `Task Definition Window: Select coda bug reports from Choices Listbox`

⇒ Select element.
- `Task Definition Window: Select coda sources from Choices Listbox`

⇒ Select element.
- `Task Definition Window: Select building coda from Choices Listbox`

⇒ Select element.
- `Task Definition Window: Select compress related from Choices Listbox`

⇒ [Select element.]
- `Task Definition Window: Select tcl testing from Choices Listbox`

⇒ Select element.
- `Task Definition Window: Select editing from Choices Listbox`

⇒ Save task definition.
- `Task Definition Window: save`
- `Task Definition Window: save as`
- `Task Definition Window: save`
- `Task Definition Window: Show Coda bug fixing`

12) [par 16] Define a task to hoard header files.

⇒ Select a task.

- `Tix Combo Arrow: click`
- `Task Definition Window: Show New...`

⇒ Type name and hit <Return>.

- `Task Definition Window: Show header files`

⇒ Select "New..." data definition.

- `Task Definition Window: Select New... from Choices Listbox`
- `Data Definition Window (ID=1): open`
- `Data Definition Window: Show New...`

⇒ Type new name and hit <Return>

- `Data Definition Window: Show header files`

⇒ Type pathname and hit <Tab>.

- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.

- `Data Definition Window: Meta-Information Select Descendants`

⇒ Type pathname and hit <Tab>.

- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.

- `Data Definition Window: Meta-Information Select Descendants`

⇒ Type pathname and hit <Tab>.

- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.

- `Data Definition Window: Meta-Information Select Descendants`

⇒ Type pathname and hit <Tab>.

- `Data Element: <Tab> enables meta-information`

⇒ Select meta-information.

- `Data Definition Window: Meta-Information Select Descendants`

⇒ Save definition.

- `Data Definition Window: save`
- `Data Definition Window: save as`
- `Data Definition Window: save`
- `Data Definition Window: Show header files`

⇒ Close data definition window.

- `DataDefinitionWindow(ID=1):beginclose`
- `Data Definition Window: save`
- `DataDefinitionWindow(ID=1):endclose`

⇒ Select element.

- `Task Definition Window: Select header files from Choices Listbox`

⇒ Save definition.

- `Task Definition Window: save`
- `Task Definition Window: save as`
- `Task Definition Window: save`
- `Task Definition Window: Show header files`

13) [par 21] Define a task to hoard your CV.

⇒ Select "New..." from combobox.

- Tix Combo Arrow: click
- Task Definition Window: Show New...

⇒ Type name and hit <Return>.

- Task Definition Window: Show cv

⇒ Scroll.

- [TaskDefinitionWindow:scroll]*

⇒ Select element.

- Task Definition Window: Select editing from Choices Listbox

⇒ Select "New..." program definition.

- Task Definition Window: Select New... from Choices Listbox
- Program Definition Window (ID=1): open
- Program Definition Window: Show New...

⇒ Type name of program and hit <Return>.

- Program Definition Window: Show scribe tools

⇒ Type name of executable and hit <Return>.

- Program Element: <Return> begin
- Program Element: <Return> end

⇒ Type name of executable and hit <Return>.

- Program Element: <Return> begin
- Program Element: <Return> end

⇒ Save program definition.

- Program Definition Window: save
- Program Definition Window: save as
- Program Definition Window: save
- Program Definition Window: Show scribe tools

⇒ Close program definition.

- ProgramDefinitionWindow(ID=1):beginclose
- Program Definition Window: save
- ProgramDefinitionWindow(1):endclose

⇒ Scroll.

- [TaskDefinitionWindow:scroll]*

⇒ Select element.

- Task Definition Window: Select scribe tools from Choices Listbox

⇒ Select "New..." data definition.

- Task Definition Window: Select New... from Choices Listbox
- Data Definition Window (ID=2): open
- Data Definition Window: Show New...

⇒ Type name and hit <Return>.

- Data Definition Window: Show cv directory

⇒ Type pathname and hit <Return>.

- Data Element: <Return> in pathname

⇒ Select meta-information.

- Data Definition Window: Meta-Information Select Descendants

⇒ Save data definition.
- Data Definition Window: save
- Data Definition Window: save as
- Data Definition Window: save
- Data Definition Window: Show cv directory

⇒ Close data definition window.
- DataDefinitionWindow(ID=2):beginclose
- Data Definition Window: save
- DataDefinitionWindow(ID=2):endclose

⇒ Select element.
- Task Definition Window: Select cv directory from Choices Listbox

⇒ Save task definition.
- Task Definition Window: save
- Task Definition Window: save as
- Task Definition Window: save
- Task Definition Window: Show cv

⇒ Close task definition window.
- TaskDefinitionWindow(ID=1):beginclose
- Task Definition Window: save
- TaskDefinitionWindow(ID=1):endclose

14) [par 14] Prioritize these tasks.

⇒ Select hoarded task.
- Task Information Window: Select selsosp16

⇒ Unhoard task.
- Task Information Window: Un-hoarding sosp16

⇒ Select hoarded task.
- Task Information Window: Select selrecommendation*-*letters

⇒ Unhoard task.
- Task Information Window: Un-hoarding recommendation letters

⇒ Select hoarded task.
- Task Information Window: Select seldarpa*-*grant

⇒ Unhoard task.
- Task Information Window: Un-hoarding darpa grant

⇒ Select task.
- HList: Select

⇒ Hoard task.
- Task Information Window: Hoarding Coda bug fixing

⇒ Select task.
- HList: Select

⇒ Hoard task.
- Task Information Window: Hoarding header files

⇒ Select task.
- HList: Select

⇒ Hoard task.
- Task Information Window: Hoarding cv

⇒ Scroll to verify hoard priorities.
- `[TaskInformationWindow:scroll:.task.panes.selectedTasks.f.border.frame.tasklist]*`

⇒ Close "Task Information" window.
- `Task Window: close`

15) [par 3] Please request an Immediate Hoard Walk.

⇒ Select Hoard Walk Indicator Light.
- `Hoard Walk Indicator: double-click`

⇒ Request immediate hoard walk.
- `HoardWalkInactive:walknow`

⇒ Close "Hoard Walk Information" window.
- `Hoard Walk Window: close`

16) [par 2] Incident: Filling RVM Space.

⇒ Select Space Indicator Light.
- `Space Indicator: double-click`

⇒ Close "Space Information" window.
- `Space Window: close`

17) [par 0] Incident: Fixing RVM Space.

18) [par 3] Incident: Task Unavailable.

⇒ Select Task Indicator Light.
- `Task Indicator: double-click`

⇒ Scroll to verify just one task is unavailable.
- `[TaskInformationWindow:scroll:.task.panes.selectedTasks.f.border.frame.tasklist]*`

⇒ Close "Task Information" window.
- `Task Window: close`

19) [par 0] Incident: Hoard Walk Begin Event

20) [par 0] Incident: Hoard Walk Finish Walk

21) [par 2] Incident: Operating Weakly Connected.

⇒ Select Network Indicator Light.
- `Network Indicator: double-click`

⇒ Close "Network Information" window.
- `Network Window: close`

22) [par 1] Incident: Weak Miss Event.
- `Advice Invokation: automatic`
- `Advice Information Window: advice invokation`

⇒ Select answer.
- `WeakMissQuery:select1`

23) [par 1] Incident: Operating Disconnected.
- `Advice Invokation: automatic`

⇒ Close "Network Information" window.
- `Network Window: close`

24) [par 2] Incident: Disconnected Cache Miss Event.
- `Advice Invokation: automatic`
- `Advice Information Window: advice invokation`

⇒ Select answer.
- `[[DiscoMiss:scale]*[DiscoMiss: scale motion]*]*`

⇒ Complete questionnaire.

- `Disco Miss: done`

25) [par 0] Incident: Operating Strongly Connected.

26) [par 0] Incident: Tokens Expiry Event.

- Do not count action against user if they type a password and click "Submit".

27) [par 4] Incident: Reintegration Pending Event.

⇒ Select Tokens Indicator Light.

- `Tokens Indicator: double-click`

⇒ Close "Authentication Information" window.

- `Tokens Window: close`

⇒ Select Reintegration Indicator Light.

- `Reintegration Indicator: double-click`

⇒ Close "Reintegration Information" window.

- `Reintegration Window: close`

C Experimental Data

This appendix contains data collected during the CodaConsole usability study. In it, you will find demographic information, quantitative data, and qualitative data. The section containing demographic information includes information obtained from the participants using the Background Questionnaire. The section containing quantitative data represents the raw data collected during the Tutorial and Exercise segments of the study. The section containing qualitative data represents the results of the Evaluation Questionnaire that participants were asked to complete at the conclusion of the Exercises. The materials used to collect this information are available in Appendix B : "Materials".

C.1 Demographic Data

Question	N1	N2	N4	E1	E2	E3
Gender?	Male	Male	Female	Male	Male	Male
Highest Degree?	HS	Masters	Bachelors	Masters	Masters	Masters
Job Title?	Student	Student	Student	Student	Student	Student
Field of Study?	CS	CS	CS	CS	CS	CS
Level of Study?	Ph.D.	Ph.D.	Ph.D.	Ph.D.	Ph.D.	Ph.D.
Year in Program?	2	1	2	6	5	8
HCI Experience?	Yes	Yes	No	No	No	No
Frequency of Computer Use?	Daily	Daily	Daily	Daily	Daily	Daily

Figure C.1: Demographics

This table shows the demographics of participants in our usability study. Participants N1 and N2 each had some prior experience in human-computer interaction. N1 had worked on several shipping applications for the MacIntosh. N2 had worked as an interface researcher in the labs of two different companies (total of 3 years). In addition, N2 has a masters in CS with an emphasis in HCI.

Question	N1	N2	N4	E1	E2	E3
Operating System Proficiency?						
DOS/Windows	✓	✓	✓	✓	✓	✓
MacOS	✓	✓			✓	✓
Flavor of Unix	✓	✓	✓	✓	✓	✓
Others	✓					✓

Figure C.2: Proficiency with Various Operating Systems

This table shows each of the participant's proficiency with various operating systems.

Question	N1	N2	N4	E1	E2	E3
Use Window Environment?	Yes	Yes	Yes	Yes	Yes	Yes
DEC Windows				✓		
MacIntosh	✓	✓	✓	✓	✓	✓
MS-Windows	✓	✓	✓	✓	✓	✓
NeXTStep	✓		✓	✓	✓	✓
Presentation Manager	✓					
Sun/OpenLook or View			✓	✓		✓
X windows	✓	✓	✓	✓	✓	✓
Other		✓				✓

Figure C.3: Use of Various Windowing Systems
This table shows each participant's experience with various windowing systems.

Question	N1	N2	N4	E1	E2	E3
Used Unix-based System?	Yes	Yes	Yes	Yes	Yes	Yes
How frequently?	Daily	Daily	Daily	Daily	Daily	Daily
Years experience?	2	5+	5+	5+	5+	5+
Rate ability?	Good	Excel't	Ok	Good	Good	Excel't

Figure C.4: Experience with Unix-based Operating Systems
This table shows each participant's experience using Unix-based operating systems.

X Window Manager Preferences	N1	N2	N4	E1	E2	E3
mwm		✓			✓	
twm			✓			✓
tvtwm				✓		
ctwm			✓		✓	
AfterStep	✓					

Figure C.5: Window Manager Preferences
This table shows each participant's preferred X window manager.

Editor Proficiencies?	N1	N2	N4	E1	E2	E3
vi				✓	✓	
emacs						
gnu-emacs	✓					
Epoch					✓	
Xemacs						

Figure C.6: Editor Proficiencies

This table shows each user's proficiency with commonly used editors available under Unix. The editor marked with a represents the user's preferred Unix-based editor.

Document Tool Proficiencies?	N1	N2	N4	E1	E2	E3
scribe				✓		
latex						✓
frame			✓		✓	✓
word				✓		

Figure C.7: Document Processing Proficiencies

This table shows each participant's proficiency with various document processing tools commonly used available for Unix. The one marked with a represents the user's preferred document processing tool. User E3 has no preference.

Have you ever maintained or compiled any Unix applications, either here at Carnegie Mellon or elsewhere?

⇒ (N1) gnu-gcc | vscm
⇒ (N2) java compiler; misc. other stuff
⇒ (N4) No
⇒ (E1) rpc2, scylla, informix, gdb, ghostscript, ghostview, gmax, flex, bison, etc.
⇒ (E2) many (tcl/tk, epoch, …)
⇒ (E3) cboard, mdos, pilot-link

File System Usage	N1	N2	N4	E1	E2	E3
Experience with AFS?	Yes	Yes	Yes	Yes	Yes	Yes
How frequent?	Daily	Daily	Daily	Daily	Daily	Daily
home directory?	Yes	Yes	No*	Yes	No	No
e-mail?	Yes	Yes	Yes	Yes	Yes	No
Knowledge of Coda?	Yes	No	Yes	Yes	Yes	Yes
Experience with Coda?	No	No	No	Yes	Yes	Yes

Figure C.8: Distributed File System Experience

This table shows each participant's experience with afs and with Coda. N4 notes that, although her home directory is not in AFS, it used to be located there. Below are the user's responses to the question "What do you know about Coda?".

⇒ (N1) It's AFS++

⇒ (N4) it's a file system (apparently). Also, Coda people walk around with notebooks a lot.

⇒ (E1) Medium-level familiarity with implementation details. High familiarity with design.

⇒ (E2) Mostly user level things (hoard, spy, cfs ...)

⇒ (E3) Have browsed source code and suggested bug fixes.

Related Experience	N1	N2	N4	E1	E2	E3
Knowledge of Briefcase?	No	No	No	Yes	No	Yes
Experience with Briefcase?	No	No	No	No	No	No

Figure C.9: Related Experience with Briefcase

This table shows each participant's knowledge of and experience with Briefcase. Only two users had prior knowledge of Briefcase. Below are their responses to the question "What do you know about Briefcase?".

⇒ (E1) utility to provide some (minimal) degree of synchronization of files between different machines. Users must explicitly name all files to keep synch.

⇒ (E3) it syncs in some lame fashion a user-defined set of files (if you can call what they have a file system, which is debatable).

C.2 Quantitative Data

Screen	Time Required (in seconds)			Novice	Time Required (in seconds)			Expert	Overall	
Identifier	N1	N2	N4	Avg	E1	E2	E3	Avg	Avg	StdDev
Introduction	**626.3**	**516.0**	**391.3**	**511.2**	**544.0**	**629.1**	**352.4**	**508.5**	**509.9**	**116.5**
Introduction1	32.4	25.8	50.6	36.3	33.1	27.4	14.3	24.9	30.6	11.9
Introduction2	29.7	19.9	7.8	19.1	21.4	21.2	12.2	18.3	18.7	7.7
Introduction3	29.7	22.4	16.4	22.8	22.5	24.5	7.4	18.1	20.5	7.7
Introduction4	79.7	57.8	43.5	60.3	59.4	71.3	44.4	58.4	59.4	14.4
Introduction5	82.0	66.9	35.1	61.3	62.5	75.2	46.0	61.2	61.3	17.7
Introduction6	58.4	46.2	27.7	44.1	56.5	67.0	39.9	54.5	49.3	14.2
Introduction7	83.7	85.2	55.8	74.9	79.2	98.1	42.9	73.4	74.1	20.6
Introduction8	19.7	13.1	13.6	15.5	15.7	35.7	5.4	18.9	17.2	10.2
Introduction9	63.0	57.0	55.4	58.5	66.6	78.0	60.5	68.4	63.4	8.2
Introduction10	108.3	85.3	59.4	84.3	87.9	93.8	45.7	75.8	80.1	23.2
Introduction11	39.6	35.9	26.0	33.8	39.2	36.3	33.9	36.5	35.1	5.0
Urgency Colors	**186.8**	**150.5**	**172.2**	**169.8**	**221.2**	**204.1**	**82.8**	**169.4**	**169.6**	**49.1**
UrgencyColors1	44.2	26.8	22.8	31.3	38.7	43.5	28.2	36.8	34.0	9.2
UrgencyColors2	142.6	124.3	149.4	138.8	182.5	160.5	54.5	132.5	135.6	44.2
Network	**96.6**	**79.8**	**84.0**	**86.8**	**90.1**	**108.5**	**101.8**	**100.1**	**93.5**	**10.9**
Network1	71.0	52.9	46.6	56.8	60.0	79.7	68.9	69.5	63.2	12.3
Network2	13.2	18.1	18.0	16.4	19.1	18.9	20.9	19.6	18.0	2.6
Network3	12.0	8.4	19.1	13.2	10.7	10.1	11.7	10.8	12.0	3.7
Task	**1575.8**	**1502.2**	**917.2**	**1331.7**	**1413.7**	**1310.9**	**897.8**	**1207.5**	**1269.6**	**294.2**
Task1	66.5	74.1	61.6	67.4	79.4	68.5	44.1	64.0	65.7	12.2
Task2	57.8	83.2	41.5	60.8	89.4	70.0	56.4	71.9	66.4	18.0
Task3	179.8	137.0	57.0	124.6	115.6	115.0	70.1	100.2	112.4	44.8
Task4	16.6	34.6	18.9	23.4	22.0	17.4	12.6	17.3	20.3	7.6
Task5	59.4	26.7	8.3	31.5	26.1	48.3	14.9	29.8	30.6	19.6
Task6	203.8	82.6	42.6	109.7	63.6	44.1	60.1	55.9	82.8	61.1
Task7	133.1	134.8	85.0	117.6	117.7	112.5	73.7	101.3	109.5	25.1
Task8	25.0	29.6	20.6	25.1	18.1	25.1	15.4	19.5	22.3	5.2
Task9	55.2	24.4	27.7	35.8	39.2	33.7	25.4	32.8	34.3	11.7
Task10	118.6	268.3	100.2	162.4	204.3	148.3	80.4	144.3	153.4	71.0
Task11	7.2	16.3	9.3	10.9	37.5	18.0	10.4	22.0	16.4	11.1
Task12	40.8	32.7	32.4	35.3	43.4	42.1	31.2	38.9	37.1	5.6
Task13	47.4	64.2	63.4	58.3	53.1	57.1	37.2	49.1	53.7	10.3
Task14	10.6	11.9	11.1	11.2	16.0	25.2	8.9	16.7	14.0	6.0
Task15	106.2	75.9	78.6	86.9	94.3	73.6	58.4	75.4	81.2	16.8
Task16	75.9	47.2	45.2	56.1	45.5	71.4	58.5	58.5	57.3	13.7
Task17	152.3	130.2	57.4	113.3	110.1	99.8	89.7	99.9	106.6	32.9

(continued from previous page)

Screen	Time Required (in seconds)			Novice	Time Required (in seconds)			Expert	Overall	
Identifier	N1	N2	N4	Avg	E1	E2	E3	Avg	Avg	StdDev
Task18	24.3	22.2	16.9	21.1	18.2	29.1	13.3	20.2	20.7	5.7
Task19	104.1	117.7	69.2	97.0	122.9	93.7	77.9	98.2	97.6	21.4
Task20	65.8	72.2	59.1	65.7	85.1	104.2	43.6	77.6	71.7	21.1
Task21	25.2	15.9	11.0	17.4	11.9	13.5	15.2	13.5	15.4	5.1
Advice	**302.8**	**331.5**	**226.0**	**286.8**	**296.7**	**337.6**	**203.2**	**279.2**	**283.0**	**55.7**
Advice1	71.1	65.0	45.9	60.7	61.2	74.6	39.7	58.5	59.6	13.9
Advice2	63.3	107.3	56.1	75.6	85.3	87.4	44.0	72.2	73.9	23.5
Advice3	80.3	68.4	53.5	67.4	77.6	91.3	59.1	76.0	71.7	14.1
Advice4	49.7	53.7	44.8	49.4	44.5	52.4	45.5	47.5	48.4	4.1
Advice5	38.3	37.1	25.6	33.7	28.1	31.8	14.9	24.9	29.3	8.6
Hoard Walk	**406.1**	**473.2**	**307.8**	**395.7**	**446.0**	**472.5**	**355.9**	**424.8**	**410.2**	**67.2**
HoardWalk1	73.4	71.5	60.7	68.5	95.5	91.2	54.7	80.5	74.5	16.2
HoardWalk2	61.5	76.4	37.8	58.6	59.3	78.0	42.6	60.0	59.3	16.7
HoardWalk3	98.2	95.7	79.9	91.3	87.2	93.3	77.6	86.0	88.7	8.5
HoardWalk4	49.2	62.4	48.4	53.3	64.1	61.5	38.8	54.8	54.1	10.1
HoardWalk5	72.1	105.5	53.2	76.9	63.8	94.6	103.4	87.3	82.1	22.0
HoardWalk6	51.4	61.5	27.8	46.9	76.0	53.8	38.7	56.2	51.5	16.9
Space	**109.4**	**160.9**	**99.4**	**123.2**	**152.7**	**125.7**	**132.1**	**136.8**	**130.0**	**23.9**
Space1	62.6	89.7	54.1	68.8	78.0	70.4	76.3	74.9	71.9	12.5
Space2	10.8	28.0	13.7	17.5	13.7	19.6	15.2	16.2	16.8	6.2
Space3	25.0	34.4	25.2	28.2	50.4	29.6	34.5	38.2	33.2	9.4
Space4	10.7	8.3	5.7	8.2	10.2	5.6	5.7	7.2	7.7	2.4
Tokens	**54.4**	**58.9**	**39.9**	**51.1**	**50.3**	**47.8**	**65.5**	**54.5**	**52.8**	**8.9**
Tokens1	33.0	27.0	19.5	26.5	28.1	21.4	47.3	32.3	29.4	10.0
Tokens2	13.2	17.1	9.4	13.2	15.4	21.1	12.6	16.4	14.8	4.0
Tokens3	8.1	14.1	11.0	11.1	6.7	5.2	5.6	5.8	8.5	3.5
Event Configuration	**263.9**	**316.8**	**243.6**	**274.8**	**302.7**	**256.0**	**322.0**	**293.6**	**284.2**	**33.7**
EventConfig1	51.4	67.0	45.5	54.6	45.6	51.4	52.2	49.7	52.2	7.9
EventConfig2	129.8	104.2	93.3	109.1	131.1	95.7	113.2	113.3	111.2	16.5
EventConfig3	56.7	123.6	75.1	85.1	104.7	81.5	144.7	110.3	97.7	32.8
EventConfig4	14.3	8.7	6.4	9.8	4.2	13.2	4.2	7.2	8.5	4.4
EventConfig5	11.6	13.8	22.4	15.9	17.0	14.1	7.7	12.9	14.4	5.0
Token Expiry	**29.6**	**41.3**	**20.3**	**30.4**	**32.6**	**31.6**	**56.6**	**40.3**	**35.3**	**12.4**
Entire Tutorial	**3656.6**	**3770.4**	**2503.4**	**3310.1**	**3554.6**	**3530.7**	**2579.5**	**3221.6**	**3265.9**	**568.0**

Figure C.10: Timing Data Collected During the Tutorial

This table contains the time required for each user to complete each individual screen of the tutorial. In addition, we provide the total time required for all screens. For each screen, we calculate the average and standard deviation. All times are in seconds.

Screen Identifier	Time Required (in seconds) N1	N2	N4	Novice Avg	Time Required (in seconds) E1	E2	E3	Expert Avg	Overall Avg	StdDev
Exercises Introduction	72.0	50.6	54.7	59.1	65.8	74.6	52.5	64.3	61.7	10.5
Phase One	**1028.2**	**950.2**	**1169.6**	**1049.3**	**882.5**	**774.6**	**888.2**	**848.4**	**948.9**	**136.7**
Choose Urgency Colors	10.1	25.1	17.4	17.5	21.9	9.2	13.1	14.7	16.1	6.5
Network Conditions	67.3	42.4	83.0	64.2	40.9	46.9	64.3	50.7	57.5	16.8
Task Unavailable	111.3	88.6	62.9	87.6	85.0	74.0	79.4	79.5	83.5	16.3
Fix Task Unavailable	17.7	20.4	13.3	17.1	27.8	14.3	18.2	20.1	18.6	5.2
Token Expiration	26.5	15.5	16.5	19.5	22.3	17.7	26.0	22.0	20.8	4.9
Hoarding Details	112.7	91.9	85.6	96.7	101.3	103.2	99.4	101.3	99.0	9.4
Provide HW Advice	379.4	427.4	579.8	462.2	290.9	266.0	281.3	279.4	370.8	120.3
Space Status	39.4	70.9	65.8	58.7	77.7	33.5	33.4	48.2	53.5	20.2
Miss Events	209.1	135.2	206.4	183.6	168.8	157.1	227.8	184.6	184.1	35.7
Fix Miss Events	54.5	32.6	38.6	41.9	45.8	52.6	45.0	47.8	44.9	8.3
Phase Two	**1264.4**	**1326.8**	**1381.2**	**1324.1**	**1859.5**	**1127.7**	**1372.0**	**1453.1**	**1388.6**	**248.8**
Read Blizzard Bkgd	95.3	102.3	99.9	99.2	87.6	92.2	103.9	94.6	96.9	6.3
Define Bugs Task	425.5	394.9	335.1	385.2	830.9	305.8	386.8	507.8	446.5	193.2
Define Header Task	191.0	188.3	219.9	199.7	299.9	191.4	344.3	278.5	239.1	66.8
Define CV Task	355.2	352.9	259.6	322.6	352.6	202.8	189.0	248.1	285.4	78.4
Prioritize Tasks	193.4	276.6	423.3	297.8	271.8	241.0	249.7	254.2	276.0	78.0
Request HW	4.0	11.6	43.3	19.6	16.7	94.3	98.4	69.8	44.7	42.1
Phase Three	**794.2**	**726.9**	**770.2**	**763.8**	**623.0**	**666.6**	**831.8**	**707.1**	**735.5**	**79.2**
Read Incident Bkgd	74.7	98.4	79.5	84.2	82.4	99.7	140.4	107.5	95.9	24.1
Read Game Bkgd	37.4	60.1	46.8	48.1	50.0	68.1	17.2	45.1	46.6	17.9
Run Game	15.4	12.7	9.8	12.6	24.1	11.1	29.3	21.5	17.1	7.9
Filling RVM	33.3	35.2	59.7	42.7	37.6	37.1	76.3	50.3	46.5	17.5
Fixing RVM	Ø	21.1	16.6	18.9	31.8	27.4	24.8	28.0	24.3	5.8
Task Unavailable	116.0	118.4	134.4	122.9	121.6	69.0	92.4	94.3	108.6	23.7
Begin Hoard Walk	73.3	45.4	40.5	53.1	19.4	17.1	56.0	30.8	41.9	21.5
Task Available/End	38.6	20.5	32.4	30.5	10.2	22.4	11.3	14.6	22.6	11.3
Operating Weakly	46.9	57.7	41.3	48.6	37.2	64.5	61.1	54.3	51.5	11.2
Weak Miss	99.3	45.4	95.8	80.2	41.5	54.4	45.2	47.0	63.6	26.7
Operating Disco	51.9	57.0	58.3	55.7	33.4	40.1	79.0	50.8	53.3	15.9
Disco Miss	35.0	66.7	54.8	52.2	48.9	47.2	60.4	52.2	52.2	11.1
Operating Strongly	23.5	7.2	25.5	18.7	12.3	16.6	21.3	16.7	17.7	7.0
Token Expiry	79.8	29.4	19.9	43.0	23.6	33.9	41.6	33.0	38.0	21.8
Reintegration Pending	69.1	51.7	54.9	58.6	49.0	58.0	75.5	60.8	59.7	10.4
All Exercises	**3726.0**	**3725.4**	**3943.8**	**3798.4**	**4174.4**	**3385.4**	**3728.2**	**3762.7**	**3780.5**	**263.3**

Figure C.11: Timing Data Collected During the Exercises

This table contains the time required (in seconds) for each user to complete each exercise. In addition, we table includes the total time for all exercises, as well as for the three phases. For each exercise, we calculate the average and standard deviation. The computer-recorded time for the final exercise (Reintegration Pending) for user N4 was lost. The time reported is calculated from the video recording of the session.

Name of Event	Event Configuration I	Color	P	B	F	Time To Begin Investigation (seconds) Novice 1	Novice 2	Novice 4	Expert 1	Expert 2	Expert 3	Avg	Std Dev
Fill RVM	S	G→R			✓	9.7	11.8	6.6	17.0	4.8	8.1	9.7	4.3
Fixing RVM	S	R→G				Ø	11.5	10.2	86.9*	14.8	5.7	10.6	3.8
Task Unavailable	T	G→R		✓	✓	6.4	9.5	5.7	5.0	3.8	4.5	4.2	0.5
Hoard Walk Begin	H	G→Y				4.9	32.3	48.7	29.7	6.2	16.2	--	--
End Walk & Task Available	H T	Y→G R→G			 ✓	19.4	18.5	11.1	10.8	14.9	13.5	--	--
Weakly Connected	N	G→Y				41.3	5.8	8.7	4.5	4.3	4.5	11.5	14.7
Weakly Connected Miss	A	G→R	✓		✓	12.3	40.5	5.3	13.6	11.1	17.3	20.6	13.5
Disconnected	N	Y→R	✓			37.9	72.9	8.3	2.9	11.1	13.6	3.0	5.6
Disconnected Miss	A	G→R	✓		✓	10.8	31.0	4.5	36.0	49.3	17.3	17.2	13.6
Strongly Connected	N	R→G				3.2	8.6	8.9	3.5	4.1	6.0	6.1	2.7
Token Expiry	K	G→Y		✓		7.1	4.5	5.8	5.0	4.1	5.0	4.9	0.6
Reintegration Pending	K R	Y→R G→R			✓	19.4	24.2	8.0	8.2	19.3	6.6	13.3	8.0

Figure C.12: Time Until Investigation Begins

This table shows the time until the user begins investigating various events. For each event, we specify how it was configured. This configuration specifies on which indicator the event was alerted to the user (A=Advice, K=Tokens, N=Network, R=Reintegration, S=Space, and T=Task), what color change occurred (R=Red, Y=Yellow, and G=Green) and whether or not the indicator popped up a window (P), beeped the bell (B), or flashed the indicator light (F). For each user and event, the time until the user began investigated the event is listed. All times are in seconds. The empty set notation (Ø) means that the user did not investigate this event. Numbers shown in gray represent data that is considered suspect because the timing overlapped with either the previous or next event.

*User E1 did not fill out an incident report until the next event occurred. At the time of the next event, the user commented that they had forgotten to fill out a form. There was, however, no observable indication that the user noticed this event when it actually occurred.

Problem Identifier	Pts	Novice Users				Expert Users				Overall	
		1	2	4	Avg	1	2	3	Avg	Avg	StdDev
Describe current network conditions	3	3	3	3	3.0	3	3	3	3.0	3.0	0
Describe "Tasks unavailable" event & configuration	6	5	6	6	5.7	6	6	6	6.0	5.8	0.4
Fix configuration for this event	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Determine token expiration time	1	1	1	1	1.0	1	1	1	1.0	1.0	0
Describe hoarding details	12	12	12	12	12.0	12	12	12	12.0	12.0	0
Provide hoard walk advice	2	2	1	0*	1.0	1	1	1	1.0	1.0	0.6
Describe current space status	3	2	3	3	2.7	1	2	3	2.0	2.3	0.8
Describe "Weakly Connected Cache Miss" event & "Disconnected Cache Miss" event	8	6	8	8	7.3	8	8	8	8.0	7.7	0.8
Fix configuration for above events	4	4	4	4	4.0	4	4	4	4.0	4.0	0
Define "Fixing Bugs" task	9	7	8	7	7.3	9	8	7	8.0	7.7	0.8
Define "Header Files" task	5	5	5	5	5.0	5	5	5	5.0	5.0	0
Define "CV" task	6	5	6	6	5.7	6	6	6	6.0	5.8	0.4
Prioritize newly defined tasks	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Request immediate hoard walk	1	1	1	0*	0.7	1	1	1	1.0	0.8	0.4
Describe "Filling RVM" problem	3	3	3	3	3.0	3	3	3	3.0	3.0	0
Describe "Fixed RVM" problem	2	Ø	2	2	1.3	2	2	2	2.0	1.7	0.8
Describe "Task Unavailable" problem	3	3	3	3	3.0	3	3	3	3.0	3.0	0
Describe "Hoard Walk Begin" event	2	2	2	1	1.7	2	2	2	2.0	1.8	0.4
Describe "Hoard Walk End/Task Available" event	2	2	2	2	2.0	2	2	1	1.7	1.8	0.4
Describe "Operating Weakly Connected" problem	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Weak Miss" event	2	2	1	2	1.7	2	2	2	2.0	1.8	0.4
Describe "Operating Disconnected" problem	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Disconnected Cache Miss" event	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Operating Strongly Connected" problem	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Tokens Expiry Event"	3	3	3	3	3.0	3	3	3	3.0	3.0	0
Describe "Reintegration Pending Tokens" event	3	3	3	3	3.0	3	3	3	3.0	3.0	0
Total Points	92	83	89	86	86.0	89	89	88	88.7	87.3	2.4
Percentage Correct	100	90	97	93	93.3	97	97	96	96.7	95.0	2.9

Figure C.13: Correctness Ratings

This table shows the score each user received on each of the exercises as well as their total and percent correct. The empty set notation (Ø) indicates the user did not perform this exercise. User N4 explained precisely which files she would have requested to have fetched during the "Provide hoard walk advice" exercises; unfortunately, she failed to actually select those objects to be fetched in the interface. Had she completed the exercise as she described, her score would have been a 1. Because of her failure to complete this exercise, she was also unable to complete the "Request immediate hoard walk" exercise – though she did the most reasonable thing given the situation.

Problem Identifier	Par	Novice Users 1	2	4	Avg	Expert Users 1	2	3	Avg	Overall Avg	StdDev
Describe current network conditions	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Task Unavailable" event & configuration	3	5	3	6	4.7	6	3	3	4.0	4.3	1.5
Fix configuration for this event	4	3	4	3	3.3	4	4	4	4.0	3.7	0.5
Determine token expiration time	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe hoarding details	4	6	2*	4	4.0	6	4	3	4.3	4.2	1.6
Provide hoard walk advice	10	35*	46	31	37.3	20	36	26	27.3	32.3	9.0
Describe current space status	2	2	2	2	2.0	2	2	2	2.0	2.0	0
Describe "Weakly Connected Cache Miss" event & "Disconnected Cache Miss" event	3	7	5	3	5.0	3	3	15	7.0	6.0	4.7
Fix configuration for above events	8	12*	9	8	9.7	9	17	8	11.3	10.5	3.5
Define "Fixing Bugs" task	36	62	47	40	49.7	96	44	58	66.0	57.8	20.5
Define "Header Files" task	16	26	26	25	25.7	47	37	100	61.3	43.5	29.0
Define "CV" task	21	73	46	47	55.3	57	31	37	41.7	48.5	15.0
Prioritize newly defined tasks	14	5*	33*	38	25.3	27	24	37	29.3	27.3	12.2
Request immediate hoard walk	3	3	3	7	4.3	3	5	7	5.0	4.7	2.0
Describe "Filling RVM" problem	2	3	6	6	5.0	4	6	6	5.3	5.2	1.3
Describe "Fixed RVM" problem	0	Ø	0	0	0	0	2	2	1.3	0.8	1.1
Describe "Task Unavailable" problem	3	7	10	5	7.3	7	2	3	4.0	5.7	2.9
Describe "Hoard Walk Begin" event	0	1	0	0	0.3	0	0	1	0.3	0.3	0.5
Describe "Hoard Walk End/Task Available" event	0	3	5	8	5.3	7	4	4	5.0	5.2	1.9
Describe "Operating Weakly Connected" problem	2	2	4	2	2.7	3	1	6	3.3	3.0	1.8
Describe "Weak Miss" event	1	3	3	2	2.7	3	1	2	2.0	2.3	0.8
Describe "Operating Disconnected" problem	1	5	4	3	4.0	0*	0	7	2.3	3.2	2.8
Describe "Disconnected Cache Miss" event	2	2	2	2	2.0	2	4	2	2.7	2.3	0.8
Describe "Operating Strongly Connected" problem	0	2	0	2	1.3	2	1	0	1.0	1.2	1.0
Describe "Tokens Expiry Event"	0	1	3	2	2.0	2	3	2	2.3	2.2	0.8
Describe "Reintegration Pending Tokens" event	4	1	2	10	4.3	4	2	4	3.3	3.8	3.3
Total	143	273	269	260	267	318	240	343	300	284	39
Efficiency (Score/Par)	1.0	1.9	1.9	1.8	1.9	2.2	1.7	2.4	2.1	2.0	0.3

Figure C.14: Efficiency Ratings

This table shows the number of steps each user took to perform each of these required tasks. The empty set notation (Ø) indicates that the user did not perform the task. Underlined numbers represent exercises for which this user was more efficient than the interface expert. User N1 requested significantly less data be fetched during the "Provide Hoard Walk Advice" exercise though he required more steps to provide that advice. User N1 hit the certain buttons multiple times in a row (as many as eight). Consecutive events such as this are counted as two actions for the purposes of measuring the user's efficiency. Primarily this effects the two "event configuration" exercises and the three "task definition" exercises. User N1 prioritized tasks as he defined them so his efficiency on the "Prioritize newly defined tasks" exercise is artificially low. User N2 has an artificially high rating on this same task because he went back to reorganize the "Define 'Header Files' Task" while performing this task. User N2 has an artificially low efficiency score for the "Describe hoarding details" exercises because he resized the window and avoided scrolling. User E1 already had the "Network Information" window visible when he performed the "Describe 'Operating Disconnected' problem" exercise.

Window	Count	Display Time (s) Min	Max
Control Panel	31	0.2	1.2
Token Information	24	0.0	0.7
Space Information	24	0.1	0.7
Network Information	30	0.1	0.9
Advice Information	21	0.1	0.7
Hoard Walk Advice	32	0.0	0.8
Task Information	50	0.1	3.4

Figure C.15: Results for the TimeToDisplay Metric

This table shows the time required to display each of the relevant windows. For each window, we show the number of measurements observed over all six users during the study, the minimum display time and the maximum display time. The measurements exhibit a high variance (one typical of garbage collected languages). For this reason, we do not present the average or standard deviation since those metrics require a normal distribution to be meaningful.

C.3 Qualitative Data

This section contains all responses to the evaluation questionnaire administered shortly after each participant completed the Exercises. For responses with a fixed set of answers, the individual users choosing each response is shown, with any additional comments indicated below and identified by the participant's identification in parentheses. For example in the question below, N1 thought the tutorial was neither complete nor incomplete and N2, N4, E1, E2, and E3 thought the tutorial was complete. In addition, we see that N1, E1, E2, and E3 provided additional comments on this issue. For discussion questions, all answers are shown; the participant's identification number (in parentheses) identifies each response.

1) The following questions are related to the tutorial:

   a) The tutorial coverage was:

very complete	Incomplete	neither complete nor incomplete	complete	very complete
		N1	N2, N4, E1, E2, E3	

⇒ (N1) The tutorial didn't address some features which could get me out of situations. [Debriefing reveals that user was referring to the lack of material covering reintegration.]

⇒ (E1) The only thing not in the tutorial was the reintegration indicator, which was used in the final test.

⇒ (E2) (the reintegration stuff at the end was not covered)

⇒ (E3) didn't say anything about reintegration

b) Understanding the tutorial was:

very difficult	Difficult	neither easy nor difficult	easy	Very easy
			N1, N2, N4, E1, E2	E3

⇒ (E1) I had some difficulty with <Tab> vs. <Enter> especially in task and subwindows.

c) Did the tutorial allow you to grasp the scope of the capabilities of the inteface?

YES: N1, N2, N4, E1, E2, E3

NO:

⇒ (N1) What was covered

⇒ (E1) At least I think to

d) Was the tutorial presented at an appropriate level of detail? (Circle all that apply.)

not appropriate, too little detail:

appropriate amount of detail: N1, N2, N4, E2, E3

not appropriate, too much detail: E1

⇒ (E1) I didn't need the Coda background, or full descriptions of all events.

⇒ (E2) (or maybe a little too much detail, but then I am not used to reading manuals ☺)

2) The following questions relate to the indicator light interface.

a) Using this interface was:

very difficult	difficult	neither easy nor difficult	easy	very easy
			N1, N2, E1	N4, E2, E3

⇒ (E1) Some confusion about which indicator to open. [Debriefing revealed that the user was referring to the Task and Hoard Walk indicators.]

b) Finding the features I wanted in the indicator lights was:

very difficult	difficult	neither easy nor difficult	Easy	very easy
		N1, E3	N2, N4, E1, E2	

⇒ (N1) Some event were not where I expected them. [Debriefing revealed user was referring to difficult finding events by their indicator in the "Event Configuration" tab.]

⇒ (E1) Some confusion about which indicator to open.

⇒ (E2) (see 3b and 6b)

⇒ (E3) a "all events" pseudo-event would have been nice; the event config dialogue box was a little clumsy

c) If you were a Coda user, how likely or unlikely would you be to keep the indicator lights visible on your screen all (or almost all) the time?

Very unlikely	Unlikely	neither likely nor unlikely	Likely	very likely
			N2, N4, E2, E3	N1, E1

⇒ (E1) Absolutely

⇒ (E3) Actually, a codacon provides immediate if terse, feedback for most events (so, ex., I know my venus hasn't died), so it might be more important to me.

d) Did the interface present information at an appropriate level of detail? (Circle all that apply.)

not appropriate, too little detail: N1, E3

appropriate amount of detail: N2, N4, E1, E2

not appropriate, too much detail:

⇒ (N1) The first level interface was appropriate, but help didn't tell me what was going on.

⇒ (E3) the "no such file" indication for task definition was pretty subtle (i.e., outright confusing).

3) The following questions relate to hoarding, both in general and specifically in terms of your preparation for operating disconnected for the duration of the blizzard.

a) Understanding the concept of hoarding was:

very difficult	difficult	neither easy nor difficult	easy	very easy
			N1, N2, E2	N4, E1, E3

⇒ (E1) I was already quite familiar.

⇒ (E2) I have had experience with it before. It is nice to have the system track which files are used by which programs.

b) Preparing for the blizzard disconnection was:

very difficult	difficult	neither easy nor difficult	easy	very easy
		E3	N1, N2, N4, E1, E2	

⇒ (E1) I had to think about how exactly to structure the tasks.

⇒ (E2) trickiest part was editting [*sic*] the file/directory lists, wasn't sure of the tab/enter interaction

c) My confidence in the task definitions I created in preparation for the blizzard is:

very low	low	neither low nor high	high	very high
			N1, N2, N4, E1, E2, E3	

⇒ (N4) ask me again after the blizzard

⇒ (E1) I think everything will work, though I might have missed something. [Debriefing reveals that user lacks confidence based upon past knowledge of failed hoards.]

⇒ (E2) wasn't sure about SOSP16 proofreading, whether "gv" was available (sorry, this might have been from previous part)

4) As we mentioned in the tutorial, our interface is adapted from the dashboard indicator lights of a car. The following questions are related to your experiences with these indicator lights.

a) Do you prefer to drive a car that uses indicator lights to one that does not?

YES (prefer one *with* indicator lights): N1, N4, E1, E2, E3

NO (prefer one *without* indicator lights): N2

⇒ (N2) Indicator lights are all right, but I prefer gauges for some measurements (eg. battery, oil pressure). Lights tend to come on only when the problem is severe. In your interface, however, they worked fine.

⇒ (N4) (especially "door ajar" on rental cars with stiff doors.)

⇒ (E2) (no experience with cars without)

⇒ (E3) I prefer both lights and guages [*sic*]

b) Dashboard indicator lights are:

Very ineffective	ineffective	neither effective nor ineffective	Effective	very effective
		N2	N1, N4, E1, E2, E3	

⇒ (N4) except when false positives cause user to ignore e.g. "check oil"

⇒ (E1) I rarely miss them.

⇒ (E3) they should blink

c) Please describe an instance of when you used the dashboard indicator lights in the last month.

⇒ (N1) I use them to tell me when to get gas.

⇒ (N2) Haven't used them.

⇒ (N4) actually, I don't own a car.

⇒ (E1) Not in the last month - but I have used them in the past for failed tail-lights.

⇒ (E2) None (have not driven in the last month)

⇒ (E3) I check "door ajar" pretty often, but otherwise I "use" them by having nothing happen.

d) Please describe a particularly memorable event when you found the dashboard indicator lights to be either particularly effective or particularly ineffective.

⇒ (N1) I have a check system light which tells me nothing. My car may be out of washer fluid or may need new spark plugs to anything in between.

⇒ (N2) On my old Dodge Dart, the oil light would come on when it was almost bone dry (nothing on dipstick). Would have preferred an oil pressure gauge.

⇒ (N4) N/A

⇒ (E1) Ineffective: the light failure sensor failed, but the taillights were fine.

⇒ (E2) low fuel warning was quite useful once, in the middle of a long trip (had lost track of gas usage).

⇒ (E3) Once my radiator had a leak. Because I have a temperature guage [*sic*], I was able to drive the car until just before overheating, refill it, and continue. A mere light would not have sufficed.

5) a) If you are not currently a Coda user, would you be interested in becoming one?

YES: N1, N2

NO: N4

⇒ (N2) I don't use a laptop, but having it keep track of file usage and caching appropriately seems nice for network outages.

⇒ (N4) but I would be in I tended to travel or work at home (or if officemates tended to kick out network connection)

b) If you are currently (or have been) a Coda user, would you be interested in using this interface?

YES: E1, E2, E3

NO:

⇒ (E1) Especially with the customization options.

⇒ (E2) *much* nicer than the old hoard/spy interface. Ability to stop long transfers over weakly connected links would also be useful.

6) a) Please describe the aspect of the interface that you liked most.

⇒ (N1) I liked the cache management for hoarding. [Debriefing revealed that user was referring to the "Task Information" window.]

⇒ (N2) Easy to define tasks. I like that it tells you the state of the network and keeps track of what files are important (eg. asking if your current task is "seriously impeded").

Under AFS, when the network goes down, you just have to guess at what's going on.

⇒ (N4) task definition

configurability of event notification

⇒ (E1) Customization - deciding what pops up and what isn't so interruptive.

⇒ (E2) Simple ways to organize file lists, and to track file usage by programs. Easy to get at other system info (like bandwidth & cache usage)

⇒ (E3) Status of hoard tasks (X% available)

Status of machine, network

b) Please describe the aspect of the interface that you liked least.

⇒ (N1) I hated entering filenames. There should at least be path completion. [During debriefing, user suggested using <Tab> to do filename completion in the data and program definition windows.]

⇒ (N2) A couple of times I deleted tasks when I meant to delete an item in a task.

⇒ (N4) Places where download time is expressed in seconds (popup had it in minutes, seconds which is ok)

⇒ (E1) <Tab> vs. <Enter>: entering information on the task windows.

⇒ (E2) Occasionally I wondered what had been saved by the "save" buttons - ie, whether all tasks were saved at once. Also, the distinction between changing the name of a task and creating a new task was not completely clear. On the indicator lights - I sometimes worried that I might miss some change - perhaps and [*sic*] indication of "this has changed since you last looked at it" would be useful.

⇒ (E3) I generally find complex graphical user interfaces to be tedious. The "no such file" indication was a real challenge.

D Transcripts

Each participant in this study was videotaped during the Tutorial and the Exercises. All of the later users were also audiotaped during the Debriefing. This appendix contains partial transcripts of these sessions. Only those portions of the sessions considered interesting were transcribed. In particular, large segments of the sessions containing only the user reading verbatim from the script were omitted. Other segments of the sessions were paraphrased. The times provided in the leftmost column are approximate. The tables below use the following notation:

Notation	Meaning
Normal Font	Indicates verbatim of the participant.
Bold Font	Indicates verbatim of the experimenter.
Underlining	Indicates speaker's emphasis.
"Quotation"	Indicates verbatim of the script (Tutorial or Exercises).
(Parenthesis)	Indicates transcriber's uncertainty over verbatim.
Italics	Indicates a note from the experimenter.
[Square Brackets]	Indicates a note from the transcriber.
Das–	Indicates user stopped speaking in the middle of an utterance.
~42	Indicates user's comment contradicts the numbered comment.
42	Indicates problem addressed for later users.
{42}	Indicates a problem implied by user's comment, or a situation where user would have benefited from suggestion described by numbered comment.
:	Indicates verbatim continues, but has been omitted.
Harry / Harriet	Indicates the experimenter using the participant's name (name changed to protect participant's anonymity).

The organization of this Appendix is alphabetical by participant identification number. The chronological order of the participants was: P1, P2, P3, P4, P5, N1, E1, S1, E2, N2, N3, E3, N4, and C1.

D.1 C1

Participant C1 was one of my committee members. He volunteered to participate in my usability test to better understand the problem, my interface, and the study. During the course of his run, I interact with him more than would normally be appropriate. I did this for two reasons. The first is that we needed to complete the Exercises and Debriefing portions of the study by a certain time. For this reason, these two segments of the study were done in parallel. The second is that the purpose of our work was to familiarize him with my thesis work. As such, I felt that it would be appropriate to stop him and explain what was happening, rather than have him struggle through a known problem.

D.1.1 Tutorial

Time	Transcript	ID
5:00	[Introduction7] Hmm. Now this is confusing me a little bit, or it seems much too subtle to me that, that in fact, that the difference between things is whether one wants to transmit or receive– just haven't thought about it that much.	
	[Moves Tutorial screen; clicks "OK"; the next window appears in the SE corner]. Ahh. [frustration]	1
8:20	["Please double-click on the 'Control Panel' indicator light."] Hmm. Indicator. I don't know where the indicator light is. "Please... double-click... on the Control Panel indicator light." I think I've got it already. "The 'Control Panel' window should soon appear on your screen. You should see a list of folders... tabs across the top of the window." [double-clicks] OK. I get it.	111
10:20	[Somewhat color-blind.] OK. So Normal is forest green... Ah... So I don't have any forest green. Warning is gold. So, I can tell these... pretty well, but let me... see what alternatives I have here. Orange Red. Dark Orange. Yeah, that's a little better fer, better for me. Or is it? Well, I think what I'll do... I'm going to change this so that, ah, Normal is just White. And Warning is Yellow. And Critical is Orange Red. OK. Now those are three that I can really tell the difference. OK.	
	[Tutorial says "green", but the user was using "white".]	106
15:00	["Let's look at the last main section, the one labeled 'User Data'". User rereads this instruction in Task3.]	
	Oh, ok. So those are subtasks. The programs you need to– so I guess the subtasks was just sort of a recursion going on here.	222 211
18:00	Oh. OK. So I just finally realized that the "predefined" things are things that are already there, or, or, that I, that I have a choice of, and things on the right so I get to drag things over or something, I guess, when I really want to define something. Hmm. I didn't understand that at first.	223

19:00 **Is the mike okay?**
Oh, sorry.
That's okay. It's easy to knock off.

20:30 Is set to none... I wonder what it means I'm not able to look at the directory, the contents or any of the children. Maybe that means it won't be hoarded. Who knows? 281

21:55 OK. So, he– we choose "all descendants" of thesis, but only immediate children of thesis. Nah this is beginning to worry me a little bit because I think I would tend to choose "all descendants" all the time, just to make sure I got all the stuff, so um, I'm just thinking if this really were my thesis it seems unlikely that I wouldn't want all of it all the time whereas if, if I'm off in some library, a set of libraries or something, then I definitely wouldn't want all of it all the time. In any case, I don't see, I don't see much value in just hoarding– I see kinda only marginal value of hoarding any particular directory unless I'm hoarding some of the stuff under it. Oh well. 113 58

Oh. Wrong <Enter>. Hit <u>that</u> <Enter>. [Numeric keypad <Enter> vs. <Return>– equivalent <Enter>.] 282

26:10 In real life, I would call this "proposal". And then you type <Enter>. 35
So, I'm going to stop– You're about to have a problem. Um.. When I say, feel free to make some up, it means use whatever path.. pathnames come to mind– not fake ones.
<u>Not</u> fake ones.
It should be real directories.
Ah, oh. OK.
Peopl–
You mean I should choose one that...
So, so for instance [mumble] Um... So one of the usability problems is that it gives you no feedback if that directory doesn't exist.
Right.
OK. Um, and, so that you don't run into– so that you don't spend a lot of time fighting this problem, I'm just going to come in and tell you. So if this, if this doesn't exist, (then) it thinks its a file.
Uh– OK.
So, for instance, if we go, I'm gonna just change this a little. OK. 'Cause this is ta–, just for you to test.
Yeah.
And then I'm going to hit <Enter>.
Yeah.
And then it enabled all that meta-information.
Ah.
This– this has to be a <u>real</u> directory.
Alright.
It can be <u>any</u> directory. It could be /tmp.
OK. Alright.
When I said feel free to make some up, what I meant was pick any, you know, pick /tmp– not <u>literally</u> make up directories.
Ah. OK. Alright.
It was a bad instruction.

35:50 [TaskUnavailable causes system to beep three times.]
Ew. So. Repeat for hacking advice. Oops! Didn't want to do that. Back. And go down here and click the right button. And there is number 2– hacking advice. So now it's got that. Hmm. A Task is red. And these things are red. So, I wonder what this means– This means that... I wonder what the red bars and 90% means. Red means it's in trouble. And maybe it means that 90% of the stuff has <u>not</u> been hoarded yet. 268

	OK. Well. I see– I see that, but I guess, I, now I'm wondering why... it just seems to me to be a binary thing. Either it's all available or it isn't so it seems a little strange to have a meter which is red up until the last moment and it suddenly turns green. But, what the hell.	114
	Hmm... Ah. [mumble] Hmm... I was dozing off there.	
39:25	Advice Needed. Advice Offered. It seems strange to me to have both kinds of advice in the same box just because they are called advice since, since needed and offered ar- are, such different– Now actually it's a question of who needs the advice– whether it's the user or the, ah, computer.	115
	Hoard Walk Advice. In this window– Hmm. Why– So why is this one called Advice as well? I think Advice is being over used here.	224
47:10	...shows you the average network bandwidth. Does this bandwidth mean the bandwidth, mean the bandwidth I've been getting recently or the bandwidth I'm capable of getting recently?	225
48:10	Oh, OK. So that that tells me the detail. This is like a Microsoft interface here. This tells me the detail of what's happening.	
48:25	This is the cost in seconds and whether they're fetched or stopped asking. I don't know. R-I actually don't know at this point whether this thing is working on the hoarding or not, or whether it's just thinking about doing it.	235
52:10	"Please request that all the tasks be fetched using the "Fetch?" button." I don't understand how to do that at all. "Please request that all the tasks be fetched by using the 'fetch question mark' button." OK. Well that seems to do it.	283
54:00	[Space2] Huh-oh. Wonderful! Boy that's the fastest facilities action I've ever seen.	
55:40	Double-click on the word "Tokens" on the indicator panel. OK. So it's a little out of sequence here because I didn't– I– ah, it.. started talking about "Destroy Tokens" and "New Tokens" before I double-clicked to even see them.	117
	Hmm. Dozing off again.	
59:25	So this just says: Notify– yes; the urgency is normal; you don't pop-up; you don't beep and you don't flash. Sooo, what do you do? Oh I guess you just put something somewhere. A window I might see later.	38
1:00:10	Under the Advice indicator, please sel– Under the Advice indicator please select the "Read Disconnect Cache Miss" event. So you're the advice indicator.	226
1:01:45	[Last paragraph of EventConfiguration3] I'm actually getting kind of lost here about why I'm doing all of this, but that's another problem.	284
	It's not done with a pop-up, but it is done with a flash. : indicator light– not just flashing the whole screen or something like that.	118

D.1.2 Exercises & Debriefing

Due to time constraints, the debriefing of participant C1 occurred simultaneously with Phase Three of the Exercises. A note is entered in the table below to mark the position when the debriefing began.

Time	Transcript	ID
	Oh, one thing. We found a bug after we started. If you use any funny characters, like parenthesis, in task names...Use letters in task names. Don't use anything else, otherwise... Just use letters. **...it can crash the interface. We have to fix this problem.**	
	[Changes normal color to gray.]	
4:21	[One or More Tasks unavailable] Hmmm... I wonder what that means... OK. I just don't remember that, but my guess is... Ew, who knows... Ah, OK, so I'm going to say "yes" here [will interface notify the user?] because it's obviously, ah, since, ah, you asked me further questions the answer must be yes. But, I don't. OK. The idea is I should go and look at, ah, some settings here. So I want to go look at the Control Panel Behavior, and a, ah, logical,	147
	no, no. So Advice? No, Advice will not, doesn't, tell me this. [mumbles] This just tells me the state of the Control Panel. Maybe I can use Help– nope. So, I vaguely remember that I had a way of indicating this thing to control it somehow but I've forgotten. Maybe it was the urgency– event configuration. OK. And this is the, ah, no, here we are. Does the interface notify? At what urgency level? Yeah, so I must have slept through this part. So this is an event– operating disconnect. I wonder where I find the event (and- ah) Physical Connectivity. Disconnect from All. Haydn. Urgency Colors. Event Configuration. OK. So I just don't remember, um, the answer	119
	to this question, but, so I'll, I'll just guess that it happens when we're disconnected and then by reading this little form here I can set yes it will notify; it will do it at the critical	
	So since we're not <u>really</u> doing this with you as a user. Yeah. **This is one of the usability problems that we've, um, documented. And that is, um, I changed this interface, um, based on comments made by one of the pilots– that they wanted to know what indicator the events were– were coming in on. I ju- used to have just a list of events; it was one of these things... [points to listbox]** Yeah, right. **...it had a list of all the events and I never told them what indicator it came in on. Alright, so when they said "Oh, show me what indicator" I thought "Well, you know, the best way to organize this is to select the indicator and then it makes the list of events much smaller." Alright?** Yeah. **But the problem is that you have to know what indicator event– an event is notified on. So this is, this is by far the worst usability problem that I've found.** Right. Ah– Although, I mean, this is like, sort of like, a quiz question and whether it's important for the person to know **close door – feedback**	

so it makes it... harder for me to answer this question what I really needed– what you're saying is that all these events are really here but I just had to, ah, know under what thing they would, they would occur.
Right.
But, I don't know if it's a really terrible usability thing because this question– suppose I never in my life– well, maybe this is an important, ah, this is probably an important one, so yeah, I guess I, I guess I should know when that's going to happen. Let me see if I can answer this now.
:
I don't know if you saw, yeah, you what saw what I was doing. I was sort of looking at one event thing, but I missed the other ones.

[When do Tokens expire?]
OK, so, I do that by looking at the Tokens thing I guess.

[Task Details]
Now the fact that some of these have not been successfully hoarded, or completely hoarded, yet I'm not going to worry about.
: [What's available]
Hmm... Actually, you know, I've forgotten. I assumed the red meant there was a problem [reads Help].

[Hoard Walk Advice]
Well, I don't know. This guy's blinking at me, but who cares... OK, so this, so the hoard walk wants some advice, and [User opens Hoard Walk Advice through Advice indicator, closes advice request window, and can't get back to it because Advice considers the request completed by the close.] Oops! So now I'm lost, ah, because it was asking for some advice. Now let's go look at the hoard walk thing to see what that says. So now I'm stuck because [mumbles]. Now advice on that hoard daemon has now the [mumbles]. "Please examine the request, decide what objects should be fetched at this time, and describe you reasoning below." Ah– Advice indicator's 120
14:35 xterm. Help. Well. [mumbles] So I did click the entry. The problem was it then sort of went away. Ah, and I, I as far as I know, I never gave it any advice. So, um, I'm just going to say all that.
[Door opens]
Hey. I'm going to flunk this.
No. I noted your problem. Um. Turns out you can get to– This is the window that came up when you double-clicked on advice. It shouldn't have disappeared like it did.
Uh. OK. I double-clicked on Advice and this came up. Oh!
You can get to– This is the one that, that, is appropriate through both the Advice and the Hoard Walk indicator and you can do it from both ways. And what I just noticed a minute ago is that it disappears the second you bring up the window from the Advice indicator, which it shouldn't do until you answer the question.
OK, so I, so I could advise it now...
15:45 **Yeah.**

OK. So I said to fetch that [SOSP16] first in case anybody cares.

17:17 [Hoard Walk pops on top when it finishes the walk.] 121
17:30 Go ahead and hoard walk. I don't care. Get that out of my face.

24:13 So I double-click in that window [double-clicking an empty "contains" list brings up a "New..." data definition window!?!] and luckily it happened to be there. [The user then expects the newly defined element to appear in the "contains" list.] 122

27:?? Well, I don't see... the data. Oh? Did that work by dragging-n-dropping. No, Ah– 23
that I can get rid of that by deleting. [User hit "Delete Task" button.] Uh, uh, uh... 6
Ew. So I just went all, to all that work typing those guys in, but now they don't appear

here. [We apparently lost the data definition.] I'm bummed. 139

[User types a path as the name of a definition]

28:40 So, now I'll save it as the bugs list. And I'm going to quit [close] and I can now drag [user attempts to drag a data element from the "predefined" list to the "contains" list] 23 this thing over here. Yuck! Don't like this interface. For that, it's too, too much monkey business as Chuck Berry used to say.

32:25 [Programs– modifying]
Add /coda/m-i-s-c/gnucomp/i-3-8-6 mach/omega/b-i-n/g-d-b. So I've now added that guy. Ew! I hit a carriage return and it disappeared. Or did it? Ew, how horrible! So 107 on a Mac I would say <Control-Z> or something and now I gotta type this whole thing 92 again. This is not so good.
[Re-enters path]
So to be on the safe side here. I'm just going to save this guy. Ew! I hate this. It just went away, but I saved it anyway.
What did it do?
[Participant explains that when he hit a <Return>, it just disappeared. There was some discussion over whether or not the pathname had been highlighted, but in fact, it had not been.]

[Tries to cut from one pathname to the next.]

Oh. So now I have my headers, and for some reason it moved "misc headers" all the 23 way over there immediately. So that's all fine. I really don't dig the use of the 123 "predefined" list and the actual misc headers list. It seems to be a s-... sort of goofy way of doing it all.

Well, now among the predefined programs, we don't have either Scribe or gv or any of that. Ah.
:
But now I've got to go back and figure out how to get all this Scribe crap in there. Well let's go look...cd [uses xterm] So my problem is I know I've got to go and get Scribe. [mumbles] So the CV files are there. To run Scribe... So I need to find where Scribe lives right now and I haven't ah, ah, yeah, my my Unix skills are deserting me now, 'cause I don't know how to find stuff.
[Experimenter points to the portion of the screen that tells him the location.] 124
Oh. /coda/misc/bin. Yeah. Thank you. I didn't even read that.
[User notices that /coda/misc/bin does not exist.]

47:00 And now the programs for Scribe, ah, are here. Now I'm just going to stuff in all this. Ah. C-a-d/coda/misc/bin. So I'm going to say those are– gee! You know, I'm surprised that I don't have, ah, I don't have– oh– /coda/misc/bin slash. Oops. Ah. Revert. C-o I'm trying to /coda/misc/bin slash. So now I save this. But what's bothering is I wanted to say give me everything in all the subdirectories and it's not 125 giving me those little buttons.
:
But I'm still kind of bothered... that... it's not giving me the opportunity to say... fet– fetch everything there is. Ahhh. Hmm. Anyway. So anyway "write CV" has– Well, to a first approximation I'm not sure this is going to work, but "write CV" is got a subtask called Scribe stuff and Scribe stuff says bring me all the programs. Ah. So this was a little more elaborate than I wanted it to be, but there you are...

49:?? OK. So here are my tasks. So I guess I must want to do– Ah, Do I do a hoard walk to decide how to set priorities? Gee, I don't remember. OK. It's not– it's not there. 126 Tokens and Space– No, not there. Advice– well maybe– it must be a task. Oh. So these are the– these are the Tasks. Hmm. OK. So these are the tasks I've got. And I decided to–. Hmm. So, writing CV, editing, Darpa Grant, Scribe stuff, writing SOSP. Uh. Now what's making me sad is that I don't have, uh, my bug fixing things. Well,

too bad. That's so. So how would I get a task? So those are the tasks up there, and I want to–. If you haven't already hoarded these tasks, please do so now. So this is, this
50:30 is the Task Information. So maybe I do something in the Hoard Walk here. [User double-clicks on the Hoard Walk indicator. The window is already visible, but no 208
indication is given to the user.] So Task Information and... Let me try this again. If I say Hoard Walk Information... Well, I tell it to go a Hoard Walk. That's fine. If I double-click on Task, it brings up Tasks. It's got a hoarded tasks and it tells me what their priorities are. Ah. Advice doesn't do anything for me. Repair. Nah– nothing.
51:20 Ah. Well, maybe the Help will tell me what to do.
[Help steals focus] 13
[Reads help.]
Oh! OK. [Reads more of the help screen.]
Alright. Alright. It's all coming back to me now thanks to the help letter. 277

[Bug task is missing. **Experimenter instructs not to worry about it, to just verbally explain where you would have put it.**]
OK, so now we'll say do the Hoard Walk, which I've already done a couple of times 121
strangely enough.

Due to time constraints, the Debriefing occurs concurrently with Phase Three of the Exercises. The user did not play tetris.

The task thing turned red and was blinking at me a little while ago, but when I double- 9
clicked it, I actually don't see anything, ah, terribly wrong here so I'm not worried. 253

[Why aren't meters a binary thing? The reasoning is that users may find it helpful to know which tasks are <u>almost</u> available. Clarify to user that a partially available task is indicated by a red meter.]

[RVM fills]
And I guess its, ah, critical. I guess it's, it's red isn't it? So that must mean critical.

59:00 **OK, so the next comment was you thought it was strange to have both of the advice needed and advice offered all in the same window.** 115
Yeah, I mean that was just... I don't know why, I mean I sort of began to get the idea that, that it was a kind of a dialog box, but, but actually the whole notion of, of Venus offering advice if that's what it– who was offering the advice was... ah, was kind of strange to me and certainly I don't... ah...
It's kind of hard to envision, since it's not there yet.
Yeah, (Since, yeah, since, since, since it,) since it's not telling me anything, I don't know what it was– what it was for and calling the thing advice, I, it actually advice is, [mumble] the name advice actually is little bit screwy to me in the sense that, ah, from 227
my Macintosh experience it's just a dialog box demanding an answer.
:
I don't know what I would call this, but... Both calling it advice and having the advice 227
go both ways so that you don't know which is the man and which is the machine is a {224}
little...a little strange.

Oh, OK, so my space problem went away miraculously even though I didn't, ah, didn't call anybody from facilities.

And as I noted, I, I was, the other thing that was sort of bothering me at the end was, 123
um, the sort of multiple layers of things that I would need to do to create a task in the 203
sense that, uh, that that I was both I was both the- the- there was the predefined list and the things that are listed on the task I would think that it would. I would say "Hey, look, a task is, a task is, a task is a list of program, a list of files and, uh, or a list of programs, a list of data and some subtasks if you want them. Ah, and, just give me a list of those things, and I don't or actually I, nnn, maybe I just didn't figure it out so and then I would think...

[TaskUnavailable event occurs, causing system to beep three times.].
...So, I interpret this thing as being a warning because it just, ah, notes that CV is not available ah, but I sort of knew that because so I assume assume that the thing is just supposed to be hoarding it so my answer is that I just wait... 254
:

So I guess what I, I think it's doing is, ah, telling me that because it thinks that I wanna, because it's a high priority and it thinks I want to write my CV right now that 254
maybe that's why it's upset, but I'm not upset at all.
[Explains why meter is flip flopping– user started his own hoard walk.]
OK, so now I've got another task complaint here. And it comes up here and it tells me. Oh, it says that "Write CV" is, ah, 100%– that's good news. So in effect it looks like everything is 100%...

[User had commented about getting lost during event configuration section of tutorial. User was unsure why he was doing all this stuff.]
I don't remember exactly what I was doing, [mumble] I was having that frustrating time of not being able to answer a question I was asked.
[User doesn't remember why he made the comment.]

[User understands weakly connected notification.]

[User reads and understands weak miss query, but has forgotten that primary task is 149
latexing his thesis.]

[User understands disconnected notification.]

[User understands disconnected miss, but decided he is not latexing his thesis today.] 149

Oh, yeah, and what happened, at one point, at one point I was doing something in some windows, and in a previous window, ah, after I'd done the carriage return the thing very nicely, first of all, I thought it was kind of bizarre that, I understand why it might be, but it was like, a little bit disconcerting that I had to do the carriage return before the thing would let me press buttons saying "None", "All, ah-, all immediate 20
files" or "All files recursively". But then, now, but then, [mumble] but I got, but I got used to doing that I said "OK, I understand how to do that every time I typed in a file or a directory or something, I have do that." But then, in that particular window 125
[program definition window] I was in those, those dia–, those buttons just weren't there.
[Experimenter explains the way program definitions work.]
If I were doing this, I would just sort of always assume that the person wanted <u>all</u> the subdirectories that were there and then would give them some other option to specify something else. Because after all
so you would make the default "all descendants".
All, yeah, I would make the default "all descendants", because I actually don't know 58
what to do if, oh, I would make the default "all descendants" because, in fact, what are the choices. One choice is getting all the directories. I don't care about the directories at all. I want the <u>stuff</u>. So saying don't give me, don't even give me the immediate
descendants– I would always say "all descendants" and then if there were 113
too much {57}
If that was too much stuff, then I would just say, "Oh, well, you're forced to go down and specify these things, at, at a lower level".
In more detail.

Because there's, in no way do I want, the sort of... It's just not meaningful for me to say "just give me the directory". I don't want the directory. I understand why Venus might want it, but I don't want it. And, to say give me the immediate descendants, um, I don't act– well, [mumble] at this point, I don't (know?) whether there are any subdirectories there or not, that's the major thing so to say give me just the immediate ones would require me to know that there's some subdirectory in there, like a help subdirectory, that I don't actually need. And that would be, I'd really have to have a lot of detailed knowledge for that. So, I would always say, "give me all the descendants". And then, to cope with this thing you were saying about maybe that

would be too much. In those cases, I would– I just wouldn't be able to use that directory at all. I would have to go down, and list explicitly list the three subdirectories that I did want, ah, ah, for the descendants. And so that would, sor- that all that part of the interface would just be- go away from my point of view, and that would be fine. You just say "If I name something, you're gonna get that- you're gonna get everything thing that's there." Uh, and then, and then, and then the person after, you know, then when they get burned a couple of times, and say "gee I really can't take all that stuff", then they would just go down and specify a longer, a longer list of things that they did want. That's the way I, that's the way I would do that.

[User missed the token expiry event – was talking.]

[Experimenter tells user not to bother trying to guess the password. User understands that the tokens have expired. User explains that he doesn't know what reintegration is, but that he assumes that it doesn't require the network. Asks the system to go ahead.] 238 239

[You commented that you didn't like predefined list. Please explain.] 123
Okay, define a task. OK. And now my job is to fill in a bunch of things here.
So you want people to type them in?
No, but, ah...
You think maybe checkboxes?
Well, well, but no, the thing, the thing that was bothering me that I didn't... [mumble] In the cases where I did have to type them in, which was I bet, which was I think was like the Scribe...
data and programs
I mean, in the cases where I did have to type things in, I had to go and get them onto my predefined list. All I'm saying is that, it was sorta like, in order to get something in this box, I have to get it onto the pre-, I have to get something there first. And, then in order to get there, I gotta click that and by the time I'm doing that, (I'm) forgetting where I'm at. So I suppose what I would say is that, ah, that what would seem more natural to me, is if you would just say "Okay, here is the list of tasks. Okay, I mean, this is just sorta like a file directory. Right? I mean, the way, I think, the way, for example, that actually, that, I did that one time by the way, and I said, "Oh, okay, this is good." Here's what bother- oh, here's what bothered me. [chuckles] Is I said "Okay, I'm supposed to put something in there." And, I said, "Well, gee, how do I put something in there?" Uh, and I'd lost...
Did you double-click there and it came up with "New..."? 122
Yeah. Yeah. [chuckles]
Well, that's a new one...
Well, but then, and then, and then I dutifully typed in a bunch of stuff. And then when I closed this...
You didn't save it and there's no indication... 127
...there was nothing there. And I was, yeah, and I was, and I was pissed.
That's a bug. I'm sorry. I saw you do that.
[User may have saved. May not work if you double-click in (empty?) contains list.]
:

In the Macintosh world, ah, if I'm supposed to fill this out. Ah, I'm just supposed to use some other general mechanism to go and find a list of names. I mean, see actually, so I don't know, ah, ah, when I'm looking at all this stuff, I'm, I'm actually lost about what these things actually are... I mean I understand that these are, these are titles for things and underneath the titles are a bunch of files, ah, but all the, the indirection through all these names and everything is getting me down. So, if I were, you know, if I were, the most intuitive interface for me would be to say "Look. It's a list of files. {203}
It's a list of files that you're supposed to put in here. And, if that's too gross for you,

you can always use the subtask mechanism to have an alias for a bunch of files. Ah, Ah, or it's a list of files... So, or, I guess, I don't know. But if, if I were defining the task structure, I would just say "Look. A task, a task, underneath it all, a task is a list of subdirectories."
So you want hoard profiles.
Hoard profiles?
That's exactly what we have now.
[Both: laughter]

Well, I don't know wheth-, I don't actually know. I mean- I. So you're probably trying to respond to... What I'm complaining about is just getting confused here. It may be that if I, if all, that, put it this way, the most obvious thing for me would be to write "a task, a task is equal to a list of subdirectories". Now, for the moment I understand that I can, so in Unix the way I could, well, come to think of it, yeah, so, a task is equal to a list of subdirectories, ah, which, in fact, I could accomp-, I guess I could accomplish in Unix by inventing myself a, a directory and then putting a bunch of symbolic links in it, and I'd be done. That would be, I mean, that would be, like, coherent with the Unix world of doing things. So you could, 'cause when you first, I mean, when you, I certainly understand the problem is that the subdir-, that the, um, that the tasks are invented, the tasks are invented to overcome the fact that the directory structure doesn't reflect the task structure perfectly. Ah, but what, and, okay so we're, so all I'm, all I'm saying is that, yeah, this, this learning this new, learning the interface to the task system was a, was a ah, is a kind of learning task that I was faced with, that any user is faced with, and I anyway, if you just said to me, if I want, [mumble] Well, suppose you never invented the task thing, then what I would say to the Coda users of the world is "Well, look, you specify, for, for hoarding purposes, if you, if you have a bunch of things that you want to be fetched, just invent yourself a directory and point at all the things, you know, that you want using symbolic links and that will organize it just fine for you. And, and, that, and that, that actually wouldn't look- might not look as, look as nice as this, but, [mumble] for all the Unix users, they'll say "Oh, OK. I get it. I can just do this." Because they already have that skill, whereas I had to learn, and I guess this is always a problem, [mumble...] Even though this thing might be better designed for the specific thing you're doing, it also required learning. So, so these things here, so yeah, so the thing that was confusing me is, after a-, after awhile, I began to get a, I said "Okay, when we get down to bedrock here, everything here is actually, underneath it all is a list of, is a list of, is a list of paths." Um. And so going through all the, the, having to do all these preliminary steps in order to...
all the graphical part
1:20:15 Yeah, ah, but it's a, a, it's a tradeoff. I mean, after awhile, I mean, at one point there, I got ambitious and defined a subtask called Scribe and everything, but I didn't, ah, ah...
:

[User discusses how mh uses a bunch of little Unix commands in the spirit of Unix. Could we try the same thing here? Potential problem is that programs and data are getting treated differently.]
[Experimenter discuss difference between programs and data.]

Another way of describing this interface is would just be, um, by rehearsal. You say, "Okay, look, I'm going to go away, I'm going to go away for a week, and when I'm away that week, here's the kind of things I'm going to do." [...] Then, I would just rumble around and touch things. And, I would be touching both the programs and the data. So, it isn't obvious to me that the data should be treated differently from programs.

[Explains that we want to reuse program definitions. That much of what programs touch are independent of user data.]

Yeah, I think, actually, I think that's a, that's a good observation, but actually it's one that I would try to...that's kinda, that, that's kind of a, you're basically saying that if we sort of look at all the files you're going to look at there are going to be... certain, to describe this in a neutral way between program and data, you're going to say "We know that if you ask for one set of things, that you're probably going to get a whole set of other things too because that's just the way it is and if these things, and that's particularly true among the programs because the programs have all their own binaries and everything that they're fetching in and... I mean, even independent of the font files and things, maybe 90% of what a program touches is its gonna touch no, no matter what was whether it's you using it or somebody else using it, and then there's the 10% of special stuff which may happen just for you, or, or may happen commonly across all the things that you do or maybe just for, for certain things you do. But, in any case, uh, as a naïve user, I don't understand that thing and so may(be) it shouldn't be in the user interface that somehow that you're, what you do is, I mean another completely much mo-, much more automagic thing might be you just tell the person to rehearse their activities and when, and y- you sort of know the difference between progra- somehow you know the difference between program and data, maybe that's a problem but..., let's su-, let's su-, let's suppose that underneath you're able to say, "Oh, th- th- he's executing a bunch of stuff here and it's Scribe and TeX or everything or it's being fetched out of binary files or something and so you collect the st- the statistics for that thing... you- underneath it all, you're sort of collecting the statistics. I guess another way of looking at this is to say...if I describe this in sort of a neutral way, it would be... when... when the person starts fetching something from, their directories come, direct-, there's a certain predilection in directories, there're some kind of directories when you go into them, that once you go in there, you're probably going to get everything in the directory, which might be true fo-, for Scribe and for program directories. Then there're other kinds of directories where there is no presumption in the fact that you're, you know, which like, [...] your home directory or just your memos files or something, there is no predilection that if you've gone into it, you're likely to get more than one thing from it, and... and that's probably a true property, ah, but, I would try, I mean, my, my, you know, just my intuition after, after a couple of hours is that I would just make the user in-, I would make the, I would not try to expose that distinction in the user interface, but just to make it work for you underneath by being slightly more, more clever or something, I mean after all, this is ah, the thing about this is that it's, about the whole hoarding concept is that it's all meant to be statistical, right? So the person is kind of, in the sense that, if the person ends up on the other side of the world and they doesn't, doesn't have and they don't have their files, they can't sue you because in some sense, I haven't been given enough control over it so that I could guarantee that everything I need is going to be there, so if, if the thing's making a mistake, [mumble] you're trying to be, you know, semi-automatic about it so, ah, okay, I mean, after a couple of hours, the interface is more confused than, more confused for me than I than it's actually doing me any good.

Now after a couple of months I would find "oh, I really need", you know, one of the biggest problems you probably have in designing this interface is that a lot of the people who are giving you suggestions for it are, are experienced users who are trying to overcome problems they've been living with for months or years. Whereas for me, it's like a, two hours of problems so I don't, so I don't even see that it's solving a problem for me.

[Discuss who the subjects are in the user study. Experimenter explains that they are all graduate students. 3 Coda experts; 3 novices.]

Yeah, OK. Well so, well anyway, but, this is, this is, this is good. In a sense, what I, what I, I guess what I'm impressed by is that actually, you now, I could sort of se-, the tutorial, even though it was a pain in the butt to write, really sort of works. I mean you know, I mean, it's sort of like having a, um... So, I would sort of count one of your,

ah, contributions here is being able to, to describe the functionality of Coda in, you know, 100 words or less, or a thousand words or something which I don't think was ever-[...]
45 minutes or less
Yeah, right, well so which I don't think was ever done before and is an import- important part of system design, probably, probably more important to do it, I mean the, I mean, all the HCI bigots will say it's more important to... Ah, this should have been done N years ago so that, so that maybe design decisions would have been done differently so that, ah, so that ah, ah, it would have been easier to describe and easier to use. Ah. But that's, that's the, that's the, you know, the eternal tension between the ah, the engineers and the, ah, whoever the other people are who worry about users. Uh, I mean, for many, I don't know, actually, Coda seems to be pretty well organized, but most, many systems I know who have been done by the kinds of people doing Coda always look to me like a bunch of pieces out on the garage floor and you come in and you're, you're supposed to make them do something and they, and, and you have to, you know, spend, I mean, all the systems we used to build at Xerox and everywhere, they were always like, just bunches of pieces which you had to put together and it was gonna take a year to figure out how to put them together. So this, ah, this makes it more plausible.

D.2 E1

D.2.1 Tutorial

Time	Transcript	ID
	So, walls...I'm guessing walls is this red outline (around the xterm).	285
7:29	[CodaConsole started] Okay, something just popped up. A little box. Control Panel, Tokens, Space,	

	Network, Advice, everything is green. I'll bet that's good.	
8:47	[Color blind– other cultures] OK. So, I can do those colors. I wonder…so, traffic lights have that, th– the ordering and I wonder if something like that might work here too.	
12:55	One of what servers? Had the color been red instead of yellow, the indicator would have been showing that our connection to at least one server had been severed. OK. Double-click on the network indicator for more detail regarding our connection to individual servers. Aha! So that will tell me. I see. OK. Cool.	
13:29	So what if we have 100 Mb/s? Can that be scaled I wonder.	50
14:15	Soon the bandwidth to scarlatti will drop to 0, which it just did, and Oh! It's blinking yellow and red? Ah, I thought it would have just changed red. This indicator flashes red and the network information window automatically updates to show the change in scarlatti's estimated bandwidth. OK. Oh, so it stops flashing... So maybe that's just to draw my attention.	9 33
16:00	[First impressions of Task Information window.] So, cache space. Alright. So the task, I'll read, instead of trying to figure it out. Although, let's see, so it looks like there are 97 MBs available and that 70% of it is either available, 70% of that 97 is available or 97 MB is available. Let's read.	269
	It's too bad this window manager seems to pop everything up up there.	1
22:10	[First impression of Data Definition.] So it looks like a hoard database shown in widget form.	
22:15	So I wonder if there's they have the old hoard now and forever or just now notation as well, I wonder, I'll bet it's just now and forever.	
22:30	So I wonder if they gray out if it's a file.	
23:00	Meta-information is a weird name here.	209
23:30	Ah, yes, so they are grayed out when nothing's there.	
	Interesting that it's not <Tab> [after naming the data definition, you hit <Enter> to move the cursor down into the definition area.]	3
24:35	[Task10: <Tab>/<Enter> explanation for data definition meta-information.] Alright, I'm confused. "If the pathname you type is a directory, then, when you hit either <Tab> or <Enter>, the meta-information area will be enabled." Alright, so let's go see if I can rename this. Ah... Better not try that. <Tab> just selects the window and another <Tab>. OK. Hit either <Tab> or <Enter> the information window "will become enabled, allowing you to choose the appropriate meta-information for this entry. If you used <Enter>, an entry empty entry." OK. So <Tab>, <Tab> sends me to the meta-information entry as does <Enter> and <Enter> additionally adds an empty entry widget. OK. "You'll need to use the mouse to select the appropriate meta-information. If you used <Tab>", mouse, no keyboard shortcuts, "you'll be able to use <Tab> and <Enter> to move around and select the meta-information." Ah! OK. "If you continue to <Tab> after selecting the meta-information, a new entry widget will appear." Well, let's see.	3
	So, <Tab> zipped me to "Save"… So I really want <Enter> there.	3
	If I make it wider, what happens. Ew. Bummer.	4
	[Sees the scrollbar on the Task Definition window because an entry exceeds the horizontal width of the widget.] Oh, OK, so there's a little scroll bar down here. Well, that's good.	
	Actually, it's a group of programs it looks like.	

28:30	Executable, but I wonder if there's other stuff.	
29:00	Oh?! ... Once you have indicated an interest in a program, the system will track what files and directories that program requires to run. Neat... Although the system will ignore any files and directories in areas of the file system that you have defined as user data. So how do I do that? By making this distinction between program data and user data, we can let you share program definitions among different tasks. ... So, it's just like spy, but somehow I can decide that portions, subtrees of the space aren't spyable.	
30:10	A new set... So it seems like program is confusing 'cause I think of program as one executable, not a collection.	210
32:50	[modifying task] First, we should change the name of this task. OK, so I'm confused... Oh, so if I change the name and save it, it'll be a new task. I guess. It's not intuitive, but I guess that's what's going on.	5
	[Surprised there isn't a button to remove a choice from a definition.]	6
	I didn't expect it to alphabetize caps and uncaps. OK.	7
	Oh, just clicking it adds it.	23
35:10	[Selects a task for hoarding.] That beeping is surprising... I wonder if that beeping means I'm in trouble cache space-wise. :	261
	Ah! Because it hasn't been hoarded. So, what are those bars? Maybe what's available– what's hoarded. That could be.	268
36:50	So, there's no notion of yellow. Ew– can I change the colors here I wonder 'cause what if I'm colorblind here too.	255 240
38:15	[System must help user prepare for disconnection.] Amen to that.	
39:30	[Notice that you're weakly connected.] And everybody is at 1 although there is no actual color.	8
	OK, so I wonder if, they also order it by priority.	
	[15 second timeout] That seems (mighty) short, but... [continues reading...] Ah!	
43:??	So, I wonder if it would keep asking me over and over for the same file.	241
44:45	[Request for hoard walk advice arrives.] Yep, I see bunches of people blinking. That's weird so advice blinks green/red and hoard walk blinks yellow/red.	9
45:15	[Both indicators allow you to get to the advice request window.] Oh– that's strange.	10
45:30	Ah– Advice. OK. Interesting. So it's not the same window I got last time.	11
47:10	And I'd bet it's one of those little collapsible tree dudes.	
	Some of them are grayed out– some are bold-faced. I wonder what that means.	
48:45	[Expand each of the nodes.] Ah, OK. That's what the italics means. "If the task name appears in italics it means that this node appears in more than one location." Very neat! [for example] Fair	

	enough.	
49:20	So that's weird. It looks like, I have to click the plus sign. OK. One. Ah, maybe I'm just not in the middle of it. Yeah. Weird. OK.	204
	Do I just use the fetch button up there. I guess. Then I click on finish hoard walk. Ah. OK. So it doesn't actually do anything until I tell it to go.	235
	Task is flashing at me now. I wonder why that's doing that.	253
50:45	Ew... RVM. Interesting. [...Reading...] I know what RVM is. I wonder what people who don't know what RVM is think.	212
	RVM space is nearly full. Call Josh [Coda system administrator's name]. I like it.	
51:45	So <Return> might... OK should be the default in the Help menu so I can just press <Enter> and it goes away.	12
	[Be careful not to hoard more data than will fit in your cache.]	15
52:10	Hmm... I wonder if it could predict that I would be over, or would be close. Interesting. OK. So, the question I'm wondering is "What if, it seems like at the time I told hoard to hoard things it could tell me that I gonna get close or not close and then I might worry about this or not worry about it."	
53:30	Hmm... I wonder if it shouldn't be "Tokens Information."	213
55:55	So, acquired changes Tokens back to green, if they go away, they get warned; if there's something waiting for them, it's critical. Huh? OK. And then I– OK. Cool. So this is how this all. Ah– neat.	
56:33	So, I guess, what I'm guessing what this does is it switches it from whatever it used to be to green, which so you can't just not notify.	38

D.2.2 Exercises

Time	Transcript	ID
2:33	["One or More Task Unavailable" Event occur.] Let's open the task an– Oh. This is a Control Panel kinda question.	
5:15	User window resize box slider. [Not taught.]	
6:10	[Uses Task Help Window successfully to remind self what green meter means.]	278
12:15	[Space Information Window] I'm not sure if that's what's dedicated or what's used?	270
12:50	[Configuration for Miss Events] I'd bet it's advice. Yes, how 'bout that.	17
18:50	[Looks at src directories to see how big the trees are.]	{57}
19:15	And, then, these are all compressed so I gonna need something to uncompress them– potentially.	{185}
20:35	Hmm. OK. I may have the wrong window... Oh, I have to open a task that's defined. OK. That's right. So, I don't see. Oh, programs defined as. Alright. So let's open a new set. Alright, now I'm back on track.	18

[User explores what's already defined.]

23:01	[Starts defining a task; then goes exploring; then tries to come back.] Ew.. Oh, that's not good! Oh, I didn't save it, and I think I just lost where I started. Well, let's start again. OK. So, it would have been nice if, as I, if I, as I switched away from the task that I was starting to define, it warned me that I was going to lose it.	19
26:20	[Types in pathname– realizes must hit <Tab> or <Enter> to enable meta-] It would be nice if, if after there I could click on rather than having to <Tab> over to activate this and click– if I could click on it without having to do that <Tab> and then at the click it could check to see if it was a directory.	20
26:37	That's a little strange too but I don't know what I would want it to do differently. Um, that <Enter> was surprising 'cause when I had it back up here and pressed enter when the "None" was highlighted, it switch from "all descendants" to "None" instead of doing what I sort of expected which was switching me to this new box. So this is kind of a little bit confusing for me.	3
27:40	Oh, I forgot how to delete this. Hmm. The one thing I'd like to do is I'm going to move Coda sources into hacking Coda rather than BugFixCoda and then just add the bugs bits for bug fix so that hacking might not include fixing prevously reported bugs. So, let's see. I'm going to need to find out. Hmm... To delete an element from a task– Ah– that's right. So select it. Yeah, it's really strange not to have a delete entry down here somewhere, somehow. Right. So Coda sources is selected– Ew– maybe I need to close this. Yeah, And it's deleted.	6 13
29:15	So, I thought <Tab> would create this new box for me. The sort of difference between <Tab> and <Enter> seems confusing to me. And, it's almost– if there's a consistent way that its used everywhere, I'm not getting it somehow. Which I kinda knew that was going to happen from the Tutorial.	3
35:20	Seeing the end of it [pathname in Data Definition element] is okay, I guess. If I resize it, yeah it doesn't really change. OK. I'm a little concerned that I can't see this whole pathname. Um. Let's see. O...	4 21
	[Double -click selects the element twice.] Yeah, I tried to double-click on xdvi, to see it. That was sort of strange. Um.	22
39:??	Oh! That was cute! So tabbing got me to save, saving got me the new box.	3
40:??	Ew. Revert. Oh, that's very strange. Huh. That's weird. So it's weird for me that a single click actually moves that over. Um and I've been having a bit of trouble with single clicks moving things over unexpectedly.	23
40:25	Boy, it almost would be nice to have one of those tree views for these as well so I could look at them without having all of these windows. Um, 'cause it seems like I'm getting window overload here.	110 24
42:52	First, I don't need these tasks that are here at the moment... How did I get rid of them? [Uses Help successfully.]	102 279
44:30	[Hoarding a new task causes system to beep three times signifying TaskUnavailable.] Whoops! Oh, OK, that's beeping at me because it's not all loaded.	
46:30	Where'd it go? Read headers. Yeah, I'm seeing quickly that I'd need to have sort of a careful naming scheme so that I could find all of these things. When I needed to.	26
56:02	[TaskUnavailable event causes system to beep three times] OK. Let's see. Incident report. There is a critical... Oh I, Ah– Duh. Alright. So first of all I missed this one. Um. So I noticed awhile ago that the space went back and then I forgot to fill out an incident report. [fills out RVM report] OK. And then this	

	one is the important one. Critical. OK. What it looks to me like is that the reason this is complaining is that BugFixCoda isn't hoarded, but should be, I think. Why would that be beeping at me? [Reads Help Screen.] Let's make sure I understand this event. Oh, I can't 'cause I'm in Help. Alright so this event was... [Checks Control Panel.]	261 13
	:	
	[HoardWalk goes off while still completing TaskUnavailable report.]	
1:01:50	[missed a file while formatting thesis.] Space problem? It might be... So the reason I put this space up is that one of the things I'm concerned about is the fact that latex2e got kicked out of my cache. It might be because there's not enough space in the cache to hold everything I need.	28 29
1:06:50	[TokensExpired event causes system to beep three times] Oh, I'm about to lose my tokens. Ah OK. That's a warning. My tokens have expired, but I would need a new password, which I don't actually have. Although I'll	
1:07:20	see if it's the usual. Oh– that's bad. That shouldn't be readable.	30

D.2.3 Debriefing

Time	Transcript	ID
	[Explains why the tutorial didn't cover Reintegration.]	239
	[<Tab> vs. <Enter>] That's, that's actually the one area where I found the interface getting in my way. Because I don't think I could explain to you how entering information on the, on the task forms works. ... I couldn't necessarily tell you the difference between <Tab> and <Enter>, and I'm, and I'm sure I can't tell you what each one of them would do in any given situation on the task. It, it just feels – I don't know how I would simplify it, but it feels too... like, when I'm there, I just hit some random combination of <Tabs> and <Enters> until the right thing happens. **[Suggests Control keys for meta-information.]** Make life easier. Perhaps more mousing. : The one thing I couldn't understand how to use <Tab> and <Enter> for was how to get a next box. So I would almost create a next box when you start typing the first. I mean, sometimes if I hit save the new box popped up, and so the rules for getting the next box when I wanted to enter a next thing weren't at all clear, and so that... The <Tab> vs. <Enter>, if it weren't also tied in to both entering this field's worth of information as well as setting up things so I could enter a new field, it might have been easier.	3
	To delete things in the task interface, I would have liked a delete button and would have eventually remembered the keyboard shortcuts... **but at the beginning you needed the delete button.** Right.	6
	The interface is clearly designed for people who don't use the mouse often. Unsurprisingly.	
	[Uncertain about which indicator to use to start a hoard walk– Task or Hoard Walk.] With a week's use, that would be less of a deal. Because I would be able to go straight to Hoard Walk and just do it.	31
	[User indicates he would probably use "hoard walk" in an xterm.]	

[Questions about confidence in the preparation for the blizzard.]
And that's just experience. The experience of using hoard tells me I've probably left something out. And this might. I think this is easier, but you know, there's always that fear that 'cause... And usually I'll pull the cable out and try to do the things I want to do anyway before I go home just to make sure I can. And I would probably continue to do that with this at least until I did it often enough that I could pull the cable out and everything would work that it felt like a waste of time to do.

Occasionally I miss the dashboard indicators.

[Indicates that he likes ability to configure events.]
I don't need to see the pop-up window every time I miss something disconnected.
That's the one thing you can't turn off because...

[User might leave network window up all the time in a roomed window manager.]

One of the things I really don't like, currently, is not really sure whether my updates are making it back or not. And not really sure whether, you know, for misses if they are really supposed to be misses. It would be nice to see what the state of...

And the other... One other thing that would be nice to have in, which I don't know how this works with the brave new world. Sometimes, I forget, after I write reconnect, I forget to write disconnect again. And, on my home machine, this is extremely painful because I'm only ever connected via phone line. I never want it to be write reconnected while I'm actually working. And so having some little something saying– "Hey, dummy". Usually, I discover immediately because all of a sudden my machine stops.

[Dislikes single-click copy] 23
It might be, because that's the common case, I would eventually learn to just live with it. I don't know. But it feels, it doesn't feel consistent with all of the other interfaces I've– it doesn't– it's not the behavior I would expect given my experience with other interfaces. It doesn't– I mean I might actually learn to think all the other interfaces should do this.

[Reproduces Bug: Double-click on prog1, then double-click on prog2; prog2 shows up in contains list twice.] 22

So, I don't know if this is a window manager artifact or an interface artifact, but the one other thing that I noticed was that everything shows up here sort of pinned to the top corner. 1
:
The model that I've used so far that I liked for autoplacement is... and if I didn't have autoplacement on this might not be a big deal– if I was not an autoplacement kind of person, but it increments by 10 in x and y until you bounce.
:
That's actually something that should be a window manager issues– NOT an interface issue...

I got sorta window overload at one point... There's lots of windows. 24

You were talking about a tree. 110
Oh, Yeah! So when I'm looking at the task definitions, it would be nice to have a tre– a, a, a task browser that looks just like, ah where did that thing go.
[Experimenter brings up Hoard Walk Advice request.]

	Yeah. If I had a browser like this for tasks, and again this might be substantial bits of work. But that, because one of the tings I found myself doing... **How do you represent Data vs. Programs?** Colors... or fonts, either one. **How do you represent meta– child, dir, none?** See that would be harder. That I don't know. : Even just the names... But, 'cause I wanted to look at subtasks to see which ones to include, 'cause I couldn't keep in my head exactly who included what... [Example: was editing included in XYZ?] And it would be nice to have some kind of browser representation of... or even a dag of the classes. **And you could even double-click on.... To show you the details of the definition.**	
	Expected "commit" to close window as well.	32

D.3 E2

D.3.1 Tutorial

Time	Transcript	ID
	[Introduction4 re: Cannot access AFS.] This is frustrating– Yes!	
4:35	[Introduction6 describes weak connectivity.] Hmm... That's interesting. I've heard of this, but I've never used it.	
6:40	[Interface to keep users informed and help them prepare for disconnected operation.] Yeah, that was, that was certainly a problem when I was using it. Sure.	
7:10	I was a little confused here. Codacon reminded– CodaConsole reminded me of codacon, which was just an output feature before it wasn't really an interface.	
	[Re: "For the record..." on UrgencyColors2] OK, I was just thinking about that.	
13:40	Hmm... I guess there was no feedback when the commit actually happened.	32
14:45	Huh, so it looks like, actually, more than one of them is at low bandwidth, or (maybe) That's all the bandwidth that's (there). Oh, and scarlatti is down, is the one that's really having trouble. OK.	
15:40	Ah, the network indicator is blinking... Hmm! Oops it stopped blinking. Now it's solid red.	33
19:00	When I read tasks there, I was a little bit confused between tasks and subtasks. :	211

19:15	Oh! OK. So a task can include other tasks.	
21:??	[immediate children vs. all descendants] That must be that it's recursive.	214
22:??	[empty entry widget] OK. I guess I guessed right.	
24:10	[User has trouble finding "black down arrow button." Thought it meant the scrollbar's down arrow.]	34
25:??	[Task10: "Yes, the system does check if the path you type is a directory".] How does it tell if I'm going to make up a name? :	35
27:20	[Types in non-existent directory– tries ending path with a /. Tries <Tab>. Just scrolls through things. Tries <Enter>.] OK. Then it goes to a new entry. Oh. OK. So it doesn't think it's a directory. That's OK. [Tries a real directory.]	36
	[System ignores program touching file system objects in user data area.] Hmm... I wonder why it does that.	
30:45	[Saves program definition.] It's a little bit disconcerting when you click on "Save" and it adds the extra blank entry. But that's OK.	37
33:??	[Change name and hit <Enter>] I hope that created a new one rather than just changing the name of the old one. : OK, writing with latex is still listed in that window so I guess it's still OK.	5
34:45	[TaskUnavailable causes system to beep three times.] It's beeping at me. Saying something at 90%. 46 MB. Oh, maybe there's not enough space in the cache. : [reads explanation of beeping.]	261
35:45	OK. So my interpretation wasn't right.	
36:35	[Notices three vs. two typo in Task20]	
	[Please double-click on the Advice indicator– Advice1.] Mmm... Looks like nothing's there yet. : [Notice that you are weakly connected– Advice2.] The network light was yellow, but there wasn't anything mentioned in the advice.	
	Yes, the Network light is still yellow.	
	[Hoard Walk Advice request arrives.] Lots of blinking red lights.	9
48:45	[Expands "All Tasks Needing Data" in HoardWalk4.] Hmm... There must be some overlap since the total cost is less than the sum.	
	[Requests that the hoard walk finish.] And the task indicator is flashing. :	
	OK. The task indicator stopped flashing.	33
	[Reads configuration for Tokens Acquired on EventConfiguration2.] Hmm. So how does it actually notify you if everything else is turned off?	38

D.3.2 Exercises

Time	Transcript	ID
	The defaults [colors] are fine.	
2:10	[Describe current network conditions.] So, I guess I have to go to the network panel for that. Um, they're all green at the moment so I guess that means they are all strongly connected.	
2:55	[Describe "One or More Tasks Unavailable" event and its configuration.] Hmm... Well, I think I can go to the Control Panel– it has help information about that. "Tasks Unavailable" should be under the Tasks indicator, and... [describes it.]	
4:40	Actually, something is critical in tasks– I wonder what it is. Um. I guess it's because the first three tasks aren't really hoarded yet.	
5:??	[Which tasks are hoarded?] Well, I'm not sure whether this means... are. OK. I expect this means which ones are supposed to be hoarded, rather than which ones currently are hoarded. Are actually in the cache.	219
8:??	[Request for Hoard Walk Advice arrives.] OK. So Advice and Hoard Walk are blinking. Ah. Presumably because these things things are not hoarded or aren't available yet.	263
	So, it looks like we have plenty of space available. Bandwidth is not great. Let's look at what files are, it's asking for. So, we don't need anything in the Darpa Grant.. Um... 'cause we don't want to work on that. So let's look at SOSP16 first. Uh. We don't– if we're just reading the paper... ah... it looks like the paper– the paper's list includes a .ps file. So we don't need to include the editing tools for generating that ps file. So, presumably, let's fetch. Actually, let's just fetch the .ps file. I don't think we need the rest. And, and under recommendation letters... Um. So, I guess we need both the letters and the editing stuff for that. I guess that's the same as getting everything in recommendation letters.	
9:30	I guess we might also need a viewer for that. Let me see if that's included here. Hmm... So it's not under either editing or papers. So it might– so we probably should also get the .tex file as well just in case.	
11:40	[Describe the Miss Events and their configuration.] [Goes straight to the Control Panel.] So, weakly connected cache miss advice. This is probably the Advice indicator.	
14:25	[Commits Event Configuration changes for Miss Events.] I hope that committed both of them. [Checks and discovers problem?]	39 {32}
18:50	[Needs sources.] Actually, let me see. Maybe I can just get that whole directory. : OK. There's lots of stuff under there so we don't want to get everything.	{57}
	[Defines a task, then changes name– what would happen if user hit return?]	
21:50	So, given that we want to set some priorities and priorities are set between tasks– that means we have to make a new task. Ah... [does so]	
23:35	OK. I haven't quite gotten the hang of how the <Tab> and <Enter> key interact here.	3

24:??	The pathnames look to be about right. [User has difficulty viewing full pathnames.]	{21}
29:??	Actually, I'm not quite sure how to get rid of them [old tasks already hoarded]. [Uses help successfully.]	102 279
30:??	[Help window grabs focus.] Oh, it probably doesn't respond with the Help window up.	13
30:30	[TaskUnavailable causes system to beep three times.] Ah– It's not completely hoarded– that's why it's beeping at me.	
32:15	[Hoard Walk requested.] So, it looks like it's running. Hoard Walk is yellow. I guess that means it's just running. Task is red. I guess that means because some of the tasks are not completely hoarded yet. So I guess I just need to wait for this.	
33:15	So it claims to have gotten all of the files. Although Task is blinking for some reason [blinking R → G to tell user all tasks available.] Hmm... not anymore. I guess that means, maybe it was blinking to inform me that it got everything. Hmm... That's interesting.	9
35:??	[Phase Three directions.] Of course, the problem is, I might concentrate on tetris and not notice the blinking lights. [User unsure if he should try to resolve the problem.]	41
41:??	[Moves the window next to the indicator lights so that he'll notice when things go off.] I'm not sure whether I'm supposed to fix the problem myself– I could do that. **Please don't.** Oops. So something happened. Let me pause the game. Hoard Walk turned yellow. Ah– that means the hoard walk is in progress. Warning. Hoard walk in progress. We don't really need to fix anything. It's fixing a problem itself. That looks like it's progressing, and hopefully this will clear up the task problem as well.	41
51:??	[Disconnected cache miss questionnaire.] Hmm. I thought it was trying to fetch this object before.	

D.3.3 Debriefing

Time	Transcript	ID
	[Asks user to clarify his answer to Evaluation Question 6b.]	
	Perhaps an indication of "this has changed since you last looked at it" would be useful.	41
	[User discussing hoard and spy utilities.] Yeah, somehow, I could never get that to quite work right. I just wound up using some fixed list and never changed it.	
	[User thinks he lost data in Control Panel. Experimenter looked at action log for the segment he was talking about, but found nothing unusual.]	39

	[Priorities.] When I had two tasks, and was going to put in the third one and I wanted to put it below the second one, but there wasn't really any space. There wasn't much space below the second one to drop it. And so.	42
	[User shows experimenter an example. It is unclear to the user where he can click to hoard a task either first or last]	
	[Scroll down to show most recently hoarded task.]	43
	[Unclear to user where you can click to hoard first or last.]	42
	It was fun!	

D.4 E3

D.4.1 Tutorial

Time	Transcript	ID
	Don't forget to think out loud.	
	I have to say I don't like cars with just idiot lights– I like gauges.	
4:50	Now I'm puzzled. Idiot lights are pretty simple. And, um, dynamic things sound pretty scary. It doesn't seem like a simple indicator. We'll see what this is all about.	
	Maybe not the font I would have chosen. [for indicator lights]	44
	Ew! Green-Yellow-Red. OK. Good.	
	OK. Good. So color blindness has been taken into account.	
5:40	How about flashing? Seems like flashing would be nice at least for red.	
6:05	There's a help in every screen– except for the indicator panel itself. Well, that certainly seems like a mistake. There should be a help button on the... OK.	45
6:30	Double-click!? Double-click. Alright. [on indicators]	46
7:20	So it's interesting this [UrgencyColors2] is covering up this [the Control Panel window]. That's a little sad. That's alright.	47
7:50	Oh!? At least one server at been severed?! See for me, red would mean, like, at least one volume is connected to no server.	256
8:10	Double-click on Network. OK. I don't like this double-click stuff.	46
	...as a percentage of maximum Ethernet bandwidth. Hey– what happens when we have fast Ethernet? Hmm...	50

8:30	I wonder at what point it switches from being green to red– maybe like little bars drawn vertically through here would let you know when it's about to happen.	51
8:55	Well it it's 0, how come there's this little blip here? That's not really very good.	8
	Network the indicator light flashes red. Yeah, but it stopped flashing. Hmm... OK.	33
10:10	If nothing is hoarded, why is 70% of my cache space full? And, is there a way that I can click on something to find out what's actually in the cache?	52 205
	These are ugly scrollbars.	53
	Frankly, this font is a little bit smaller than I would have liked.	44
	In fact, I don't know, I mean, it seems annoying. This is 1-2-3-4– a 6 entry list that needs to be scrolled. OK. Writing thesis. Ew. Lots of itsy-bitsy things that need to be scrolled.	54
	[Task definition] Yuck. Alright. Fine. Subtasks. Programs. And User Data. OK. Hmm. OK. Well, all these little lines [window resizers?] and so on are kind of amusing. User Data. Predefined. OK ... OK. Alright. So I wonder how you define one of these predefined things.	55
12:55	[Data Definition Window] Hmm... I'm not really sure what I would have wanted here instead of what's here, but I think it would have been something.	56
	Meta-information! I'm not sure why that's meta-information.	209
14:15	All descendants seems like an attractive thing to do. And it might be interesting if, like, it would actually tell you how much disk space or cache space that would be in this area.	57
	It chose "none"! I'm not sure that's the right default for a directory... Alright– I– In particular, I think all descendants would be the right thing.	58 113
	I wonder why we have "save as".	60
16:55	Ew! Well. I certainly would have expected "save" to make the window go away.	59
	OK. So we've partitioned anything that the program accesses into either user data or all other stuff. OK.	
	[Typing in programs in Program Definition.] So, it's not clear to me it's actually checking to make sure those files exist.	61
19:50	[Change name of "writing" task to "writing with latex"]	5
20:20	What? Continue to use it for other tasks. I'll do it– I'm not sure why– OK... Oh! I understand... So we've now created a new one, which is a copy of the old one.	
	Ah... Clever. [I have no idea what he's referring to...]	
	Fix the name and hit <Enter>. [Hits <Enter>.] I don't know why you hit <Enter>.	3
	Take away latex and xdvi. [Selects "latex" in predefined list.] latex. <Delete> or <Backspace>. Well, <Delete> isn't doing anything. Neither is <Backspace>. Oh!	62
	OK. Ah! So clicking on it automatically adds it?! Hmm... That's interesting. I wonder what happens if I click on it twice. Ah. OK. What happens if– OK. The right thing happens.	23 ~46
22:25	[Learning how to hoard a task.] And move the mouse down to hoarded tasks. Right-click [TaskUnavailable causes system to beep three times.] Whooo! Oh, no!	

	This is slightly– yeah. I wonder why just clicking on things here doesn't bring them over here like it did when we were sl- defining um tasks.	63
	[Task is unavailable] Is it <u>really</u> unavailable? Or– it's just not guaranteed to be available? Maybe yellow would be the right thing. Alright, so does this say this would use up– What does this mean? I have no idea what this means here. So, I know that my overall cache is 97 Meg. I guess. 90%. I'm totally lost. Alright. Fine. [goes back to reading, perhaps the help screen] What % is currently available. OK. Green or red only. Hoard	255 269
23:45	priority. Is this just the order that I put them in. The <u>location</u> at which you <u>right</u> click. Oh. And so I would need to, like, oh, no, and I can't, like, pick– drag them around in here. That would be useful.	64
	[Notice that you are weakly connected.] I guess 'cause this is yellow…	
25:50	[Since a process is waiting, it's important it gets immediate attention.] Well, does that mean I have to pop up– if it's needs immediate attention, why do I need to have a window already up on my screen and double-click or something?	66
	[15 second timeout] Well, 15 seconds seems like it'd be too long if fetching the file would take 10 seconds. 15 seconds doesn't seem like long enough if it would take 4 hours to fetch the file.	67
	[Weak miss query pops in NW corner of screen.] You know, popping this up in the center of the screen would probably been better idea. Also, having a count-down timer I mean if it's… Yeah, I think a count down timer	1
27:05	would have been good.	68
	[Not covering "Advice Offered"] That's too bad…	
28:15	[Requested immediate hoard walk.] I wonder why it turns yellow? Is there a problem?	258
	That's pretty weird. So the hoard walk thing is blinking yellow and red. Advice is blinking green and red. That's very interesting.	9
29:00	So. Hoard Walk. Yellow means operating and red needs help.	258
	[Hoard Walk Advice window] So, if we're having a problem, why don't we want to fetch any objects? And why is this zero? [Reads tutorial] Oh! Ignore it. That's why they're all zeros. Good. [Reads tutorial] List of tasks. Cost of each task. Well, I've never liked these little–	
30:30	little– thingys that are– these basically invisible things that change state in an– in an undetectable fashion. Is that selected or not selected? Only the shadow knows.	70
30:45	[Main section of Hoard Walk Advice Window.] Yeah…. This is…. I'm not sure I like this "All Tasks Needing Data" thing.	71
	Oh. OK. [can understand] Yeah, that's nice. Good.	
	[Notices an upper-case/lower-case problem in mockup.]	72
32:50	[Looking at partially expanded tree of Hoard Walk Advice window.] This is a little interesting. Why is it showing me some stuff here… if it's not expanded? That is– Oh! This is a peer of this. Alright, well, if it's a non-expandable thing, maybe, like, putting a dot here or something like that would have been better. 'cause really it looks like, since, looks, visually like this is indented further than this. And so that's a little confusing. OK. Fine.	73
33:10	[Fetch? Buttons on Hoard Walk Advice window.] Oh. These– These are horrible! What's the matter with a checkbox, is it patented by	70

somebody?

[Clicks "Finish Hoard Walk" button.]
Turns yellow. Yellow– alright. Fine.

Wonder why Task [indicator] is blinking red. I guess. And, now it's green. But how come it's green before the hoard walk is done? [mock up interface bug]. Alright. Fine. 9 74

33:50 [Space Error event.] 9
Yeah, you got it. I'm not sure why fla-fla flashing from green to red is the right idea here. That's a little confusing. Like, maybe red flashing on and off would be right.

[Click on Space Help button.] 75
Good. So this– these– little bit subtle. Like. It doesn't look like a big button– it looks like. It's sort of– alright. Fine.

[Clicks on Help button] 1
Hew... That popped up in a really convenient place. Look. Why don't these things pop-up in a slightly more convenient place. Alright.

[Reads help for RVM Full problem.] 76
Oh. Please contact your administrator. A little bit of expert help would not have been too bad. Like, you know, reinit with this flag.

84%– 8 is mushed up here [on the meter] 77

35:30 [Space Help re: cache filling] 76
How about, like, contact your system administrator and get a bigger cache?

[Directive not to click on destroy tokens and new tokens buttons.] 78
Well, okay, but, I mean, if I've never, I mean, how would I expect them to work if I'm a user of, I don't know, Novel Netware.

Hew, and the Help flies up in a totally different part of the screen. 1

[Coda tokens are similar to Kerberos tickets.] 216
Once again you're average Novel Netware user has no idea what any of this is.

I wonder if any events are good things.

Maybe "Control Panel" should be, like, I don't know, "Alerts" or something like that. 215

For maximal user brain niceness, instead of having this be a drop-down thing, having a copy of this little window [indicator lights] over here [on Event Configuration tab] would really be nice. 206

38:30 [Event Configuration Tab] 17
So, wa-wa-wa-wa-wait. Name of the indicator light. Name of an event. Ew... Well– yeah. I think there's probly some better way of laying this out. Like. Copy this over here. And have when you click on each one have a a little window up here, you know, list the– the various events. OK. Fine.

[Pop-Up– Beep– Flash]
How about, like, deliver electric shock to user?

[Tokens Acquired] 38
Tokens Acquired should notify me?! Well, that doesn't seem right.

[Reads urgency stuff on event config.] 38
I don't? Wait! Does this mean that– when I get tokens– that ch- this is telling the token light to turn green?

[Urgency Buttons on Event Config Tab] 79
These three buttons aren't really lined up– that's kind of cute.

	Um... Alright. But, if I just got tokens by typing in a password, then notifying me doesn't seem like it would be all that useful.	38
	[Reads explanation re: Read Disconnected Operation] OK. I wonder how that's different from how a disconnected cache miss. If I'm just– if I'm doing this as a simulation of this, why would I want the alerts to behave differently? It wouldn't be a very good simulation then would it.	80
41:05	[Examine "Operating Disconnected" event.] So, operating disconnected is not an event– it's a condition. Right. So like, "Transition to Disconnected Operation" would be a better name.	217
42:45	Oops! Well, that's a total lie. My tokens– Wait. I'm confused. So, if my tokens had expired, I would have hoped that this would turn red. Um. I'm not sure what possible yellow thing could mean. And you lied to me. You said my tokens wouldn't expire until tomorrow.	257

D.4.2 Exercises

Time	Transcript	ID
	During this portion of the study, I would ask that you not use parenthesis in the names of tasks, programs or data. Presumably, (hype) characters are also out, asterisks, questions marks? **Let's just use plain, old normal characters. Dashes are okay.** Okay. **OK.**	
3:15	[Finding "One or More Tasks Unavailable" event in Event Configuration Tab.] "One or more tasks unavailable." That's probably gonna be "Task". OK.	17
4:00	[Describe event notification– Popup– No, Beep– No, Flash– No.] So, what will the notification be? If it– That seems– Well, that's a little confusing. It won't pop anything up, beep, or flash so it's not clear exactly what the notification will be. Fine.	38
4:25	[Fixes the event configuration.] I guess I have to say "Commit", though we didn't talk about this before.	81
	[When do your tokens expire?] Mine expire at at, hello Mr. Window Manager, 10:50, and I assume a.m. May 2nd. OK.	195
5:30	[What Tasks are Defined?] [complains re: size of answer entry] And you know, let's see, let's just be curious. So, no matter how big I make, oh, yeah, it has this little dinky thing. [Slider to change sizes of subwindows.] Yeah. That's marvelous. Good. That's better.	55
6:30	[Suggests the numbers on the hoarded task entries should be labeled as priorities.]	82
	[Notices task names differ between the "Hoard Walk Advice" and "Task Information" windows and the Exercises interface.]	83
	[Providing Hoard Walk Advice.] So, the question I'm confronted with is which objects – not which tasks should be fetch...	218

12:00	[Please describe the "Weakly Connected Cache Miss Advice" event.] So, this is probably one of these– Control Panel things. Ah... Control Panel. Thank you very much. Event configuration. Indicator. Nnn... It's probably one of these– well, let's see. Tokens. Maybe it's a hoard event. Let's see. No. It's a Network. No. event. No. It's a Task. No. Weakly-connected cache miss. Well. There we go. OK. I didn't s... Oh. It does say "advice" in the name, doesn't it? So I guess I should have guessed that.	17
14:20	[Selects text in Exercises interface/window.] Oh. Now that– Those are good colors! That's. My that's amongst the– ah– really good color schemes I've ever seen. [blue highlight on pink background]	84
16:15	[Hits commit after fixing configuration of event.] You know, really, when I hit "commit," something should happen. I mean, like, this part should go away, or something should go away, or something like that. Having "com–", you know, hitting "commit" and nothing changes is a little disturbing.	32
	[Confused by typo in instructions.]	
	[Opens TaskInformation window.] You know. This little scroll box is pretty annoying. [moves around, makes taller]. [Note: Interface may have forgotten where the little slider box gizmo had been placed last.]	54 206
20:00	[Tries to edit program definition.] I wonder how I edit it? [clicks, then double clicks]	18 23
20:26	[Saves the modified program definition.] And now this thing should go away, but I'll make it go away myself.	59
21:00	[Resizes Task Definition window – not graceful.]	{4}
	[Tries to click on disabled– meta-information.]	20
22:00	[Types data definition pathname.] So, it would be nice if I had some confirmation here that there were, like, some number of files.	57
23:40	[Defining multi-path data definition.] And, I'm just going to skip down here, and come back and fill the flags [meta-information] in later, because that will be fewer mouse manipulations.	3
	Don't like having to hit <Tab> on that before I can click on the little buttons– that's counter intuitive.	20
24:25	Huh! That's interesting. This is above "New..." That's kinda cute. Not what I would have expected. Seems like "New..." should always be at the top.	7
	[Forgets how to delete a task, guesses backspace, gets it right.	62
28:??	[Defining Header Files task.] "Random Coda Header Files" contains– /coda/project/coda/alpha/include and /coda/project/coda/alpha/i38-6-mach. I guess I could have said @sys there, I don't know, I'm just following directions, /include-special, but if it were me I would actually put @sys in here, so let's do that, include-special, /coda/misc/c++/@sys/alpha/include and /coda/misc/tcl/common/beta/include. Good. N-A-Are these all actually include files? Include-special. Right. Now. I probably want to go back. Hello. Ah-I haven't ever hit <Tab> here is that the reason? [meta-information not enabled]. Um... Hmm... That's a little interesting. Maybe I have to... Ew... hidden modality! No thank you! How do I? Hello. OK. That one's all descendants. This one... You mean I can adjust the flags only of the last one!? That's gonna be a swift pain. Come on... Ah... Well. That's unpleasant, isn't it? OK. Good. Now let's– let's see where– let's	

see what if I come on– what if I sneak up on this? Kay... Good. NOOO! We've leapt right– OK. Thank you very much. I have no idea. Alright. Let's. You kn- Alright. Fine. So, let's just delete some of these puppies. Let's revert. There we go. OK. What is going on here? Thank you. /coda/project /coda/alpha/@sys/include-special. Ew... I don't know what <Tab> did that time. Hello. Um. Good. Alright. Let's um... revert. Cancel this puppy. And, try again. Playing by all the rules I can remember. /coda /project/coda/alpha/include. <Tab> All descendants. OK. /coda/project /coda/alpha/@sys/include-special. <Tab> And, we– Is this some, like,

29:30 secret way of telling me that there is no such file as @sys? i386_mach/ include-special. Yes! It's a secret way of telling me that file does not exist. OK. Now we're cooking. 36

[Data Definition Window.]
It'd be nice if this window were a little bit bigger, so I could see them all at the same time. [resizes ungracefully, but no comment made.] 54 4

30:00 [Saves definition.] 59
Alright, and let's save this puppy out. Okay. Good. No positive response. But, do we have, we have "Random Coda Header Files" [in Task Definition window.] OK.

Let's, let's see what the interface does when we're definitely too full [leaves old tasks hoarded].

[Chooses CV as 1st– goes to hoard it.]
Now, what's the– I cl- come down here, and click
[TaskUnavailable event causes system to beep three times.]
Ew. That was really interesting– whatever that was. Ew. So there's, like, sort of a way of rearranging the order of these things. But the interest– OK. Good. 262
:

35:00 I am <u>not</u> understanding what's going on here. I guess only the name is the part that you can click on? But, didn't I somehow get this. Who– Nelly! OK, so if I click on the name with the right button, it'll go down. And if I click on the graph with the graph with the right button, it'll go up. Of course, now it's not– who– that's, that's nice. Wouldn't have figured that out by myself. 42
:

35:45 Of course, it would be nice if the name hadn't been occluded by this, but that's OK. 85
:

Now, these things– so my cache space looks OK. I mean, do I have to make this update? I don't remember what it used to be. Oh maybe the– the, the cache space is still okay only because these are red? And these are green? Alright, anyway, I'll just leave these in, why not? I mean, 'cause after all they're lower priority. So these are the ones that should end up actually in my cache.

37:20 Hey! I wonder what this help thing does? OK. Good. Double-clicking. Ew... That's interesting. So. Is that true? It says double-clicking. Uh-oh. Alright. Well, how much have I crashed. [Help window steals focus.] OK. That's OK. Good. Fine. 13

I wonder why– Is this red just because some of it– some things are red. 259

[Watches hoard walk get performed to see things turn green, to watch the space meter, to see things get evicted.]

38:30 Well, great, everything fit in my cache and we're still 66% in use. So I'm not sure I believe in that. 86

[Lie: latex – bibtex – latex – latex, not just latex – latex]

Well if I'm supposed to report <u>all</u> state changes, then maybe I should go back and um... into the Control Panel, and make everything notify me. [goes into Event 87

	configuration.] Good Lord! There're gonna be stacks of these, aren't there [Sighs....] Well, I certainly don't want to go through and hack on all of them. Right? That seems like it would be unfair. There are gonna be hundreds of these things. So maybe I'll just be lazy and do only the ones that are– I mean– ar– aren't– for example, like repair– Oh! I guess notification seems to be on for a lot of these things. So maybe they're all going to notify me somehow. Space. Alright. Well, I guess we'll just go with it. I mean. This is an unusual task.	
	Excerpts from the tetris page. Alright. Yeah. Yeah. I know how to play tetris.	
48:26	[Fixing Coda Bugs unavailable.] I'm not really sure why it's now not hoarded because it was just a minute ago and something of lower priority is still hoarded. Maybe one of my colleagues has just updated files. Alright. So, I guess I will try a hoard walk. Why not. **Please don't.**	29
	So, the little hoard walk thing is yellow now. Ah... If it's yellow, I should look at it.	
49:35	[Tries help on progress-meter version of Hoard Walk window.] "This is the progress 'Help' text."	249
53:30	[Moves weak miss query window to center of screen.] This really would do better if it were here...	1
54:50	So, yeah, a button that would say, like, make every change... Whoa! Nelly! We have a pop-up here. Which would have been better off if it had been a pop-up here.	87 1
55:15	Everything's at 0%. I really don't like that 0% indication, but that'll do for now. : Well, alright, so we're not at least fakely disconnected.	8
56:50	[Determine urgency of Disconnected Miss query.] I guess this is critical because the Advice monitor [indicator] is red.	
58:30	Oh, wait! Wait! The network stopped being red. Great! So, it'd be sort– you know– [Sigh] I think that I would have– at least for me– I think I would have had a little bit more of a pop-up or maybe a happy sound for that.	88
1:00:10	Uh-oh. My Tokens are yellow. Well, I guess it's a warning... I'm not sure why this is only yellow. I mean, it seems pretty bad. I mean it seems like the worst possible thing that could happen on the Tokens front would be for my tokens to expire.	257
1:00:25	Oh– it's like little grayed out text. That's not what I would want. I would want... maybe nothing at all.	30
1:01:39	Oh! OK. So my Tokens are now red. Ah! OK. So this is bad.	257
	The interesting thing is that Reintegration is also red. Why is that? OK.	239

D.4.3 Debriefing

Time	Transcript	ID
	[User dislikes "All Tasks Needing Data"] So, I'm, I guess, the the– conflict there would be sort of why not, bring up, bring that up already expanded?...or at least have it expanded and on the screen at the same time.	71

So if it, if it had "All Tasks Needing Data" and then all of the children of that expanded, but not any further expanded than that.
That would be fine.

[Assignment– any time any thing happened– report it] 87
If that were really my assignment, in real life, then what I would want is a way to go into this window and say, you know, edit the class of all events to do something rather than having to go through and select each one of the 650 of them independently.

I wanted to use [parens]. 89

So, in general, there were a lot of things where I was given this three line window on four items. And, you know, that's just not right. 54
They're all sized for the IBM laptops. By default, all of those windows fit on the IBM laptop and they take up almost all of the screen.
... I would claim something a little different which is that, sort of, at any given moment, my focus is on only one of the three things that are on the screen all at the same time.
:

I mean, maybe I'd have to see it on a laptop to, to believe you. But, it just seems like... Yeah, I don't know.
So, would you rather have the window bigger and have to scroll through the window.
No, but like, for example, when I'm choosing... so, if th– if the claim is you can't have a– a– like, for the list of all so– several things... Two examples. One example is, like when I'm doing task definition and there are, like programs and files and subtasks. Well, I'm not really going to be doing more than one of those at a time, so putting a tab thing so that only one of those three would be actually visible on the screen at the same time would be fine. Right. My– I'm not dragging something from tasks to files. I don't <u>really</u> need to see both of those at the same time. Right? And so that gives three times the screen space to use for each one. OK. And then another example.
Then you can't see the whole definition all at once.
I don't need to see the whole definition all at once.
:

Then the other thing is for the list of files. I have this little thing which is trying to show me all the different sets of files that I've already described, like gdb or whatever, like a program. OK. Well, I mean, if the problem is that you ever really show me more than two out of the twelve that I have hoarded, then don't. Then have like a pop-up window. Right?
So is this in the program definition window?

This is some part of the task definition window. So I have, like, a task is a, some set of programs. Right. So if you're telling me that the set of predefined programs I already have, which is going to be, like, ten, you can never give me scroll bar room on the laptop for more than two of them. Then, that's not really useful. Then you should just have a button which says "Predefined..." And I hit that and up comes, on the screen, the full list of predefined ones. And I can like, checkbox the ones that I want and say "OK" and that goes away again.

Alert windows should pop-up in the middle of the screen. 1

[Tokens – yellow vs. red] 257
Is it really anything waiting? Like a read? Before a read would fail, you would tell me. Or the read would have failed when you told me... If you could stall the read, while you popped up the window to give me an opportunity to get the tokens, then, like, maybe that could be red and just not having tokens could be yellow. But, if as I

	suspect in real-life you would be failing reads quietly behind my back the whole time, then, you know, no that's not yellow.	
	Your configuration window should have that list of six little things. Right. And when you click on one of those, then its list of little friends should come up. That is, the configuration thing should look like the thing you're trying to configure.	162
	I think balloon help is a great win.	90
	[Discussion of building in Help automatically.] If one of the properties of every dialog box is the help text for it. Right. There's no reason why. You don't have to have any– what I'm saying is– it's stupid for you to have write any code having to do with the help system. The help system is just blobs of text. Right. And so when you're making the dialog box, you have two more fields– one is a little summary bubble help and then the other is the help text. Right. And it should automatically put up the help button for you and handle the event and so on. I mean. There's no excuse for not doing that. Um. So it shouldn't be any extra code for the interface designer. It should just be, like, a little bit of typing.	

D.5 N1

D.5.1 Tutorial

Time	Transcript	ID
8:45	Hey! I didn't want that to happen. Hmm... We have just requested the CodaConsole be started. Oooh... The initialization takes a few seconds so you will soon see a small window appear on the right hand side. That must be you.	
	[Color blindness/other cultures– until then please be patient. Hits <Enter> Key.] What happened to <Return>? Oh, it must be a– stupid windowing system. Piece of garbage.	
12:05	[Urgency Colors tab visible] Hmm... Why is this so complicated? If you click on the middle section. Well, why can't I just drag the color to the indicator light I want? That seems perfectly reasonable. Why is this thing in the middle... OK. I should just be able to drag the colors there– none of this pop-up box hookma.	91
13:00	**Will you mumble a little louder please?** Sure. **Thanks.**	
13:40	Alright. Commit your choices [for urgency colors]. I imagine that means ok... Is that what this stupid thing is. Eh. Commit. Eh. Eh. Commit. Commit. Commit. Great. Alright.	32

14:30 [Weak network]
Severed. At least one server. Wow! Multiple servers. Hmm...

Of the maximum Ethernet bandwidth?!? Well, why didn't, why isn't it the transport layer?

16:50 [double-clicked on Task indicator light] 1
All the windows come up in the left corner. Hmm...
[moves window and closes it and then reopens it]
Oh, good! Remembers where I left it.

18:00 [Task definition window]
Why do I need three subsections? Why can't I just have one? List of things. 54

Huh! Well I didn't want to do that! [clicks on predefined element] So it's moving them over. Let's look at the last main section – the one labeled user data. The list on the left shows... So I shouldn't ... 23

18:50 [Hits "Delete" button.] 6
Yikes! They all went away. Hmm... Okay.
:
I don't see a "writing thesis" here. OK. Task's name "New...". Hmm... This doesn't seem good. Wow! I have whacked the "writing thesis". Cool!
:
Hmm... OK. Too bad I destroyed that. OK. Where did writing thesis go? Hmm..
Bye-bye.
:

21:30 Hmm... "writing thesis" disappeared. Alright. Where's undo? Nan, nah, nah, nah, nah. No undo. 92
:
Let me just get you back to the Tutorial. So, it asked you to double-click on the–
"writing thesis" task
It asked you to look at the– double-click on the "thesis" element in either
Yeah, but it went away– I killed it somehow
I know. That's why I'm trying to get you back on–
How did I kill it?
I can't tell you that. All I can do is get you on topic. Please double-click on the "thesis" element in either the predefined user data list. Alright. So the last screen asked you to do this. OK. So then this is referring. You're back on track. Even though you don't have a "writing thesis" definition.
I just have a "New"
Oh. Actually. You know what? Let me. Let me just... Turn around. Don't look. But don't kill your mike.
[Experimenter regenerates "writing thesis" task].
I don't know what's in it.
There wasn't much.
You're not looking.
Oh
OK. This is not what I wanted. What happened? Oh. This just ended up on top. So. Your windows are in a different order.
Mnn.
This is what this is referring to.
This one up here.
Yeah.

26:00 [Reading meta-information explanation.]
Well, 's "immediate children" only mean the first-level set of directories afterwards?
Huh... [keeps reading] Alright.

	[Reading meta-information on all descendants.] What does it do with the links?	93
27:30	[Reading about bovik's choices of meta-information.] Well, how does he know that? How does he know he won't have any problems? Can you put, like, how much memory it takes next each of them? That way I can know. Oh, if I select this, it's going to take lots of memory. I don't have to be– have file system e.s.p. Man, I don't want to remember the state of my computer– that's what it's for.	57
28:25	[Complaining about where ctwm has focus associated with cursor]. Ah, you stupid piece of crapola.	
30:01	[Exploring difference between <Tab> and <Enter>] Ooh. Very good.	
30:40	[Saves definition] How do I know it's been saved? Oh, OK. It showed up down here. Down here. OK. Need something to say whether this was saved or not. If it's not saved, will it tell me when I try an clos– oops! wasn't supposed to do that. Come back. OK.	59
33:45	[Entering program definition] It's a sin that anyone has to type in any pathname whatsoever. They should be able to select it. This is just– typing– just– pathnames is not for humans.	94
	[Save] [User saves program definition multiple times].	59
35:35	[Saves task definition multiple times] Why is this just all automagically saved? The save button is a total pain.	59
36:50	[Modifying name of task.] This is... This is not so good.	5
37:45	[Removing latex and xdvi from "writing with Scribe" task] Mmm... OK. So I don't want to do "Delete Task". That would be bad... Oh, of course, hit either <Backspace> or <Delete>. Mmmm, which one, which one, OK.	6
	[Still removing element] Can I just drag it out?	6
	[Hoard Walk Advice request arrives– lots of blinking] Uh Oh! All hell's breaking loose.	9
54:38	[Reading Space Help Window] Ha! Ha! Poor guy. [Reference to contacting Josh Raiff].	
55:00	[RVM problem has been solved] Help button was visible but didn't update.	95
56:??	Yeah, but I want it to tell me before the expire.	260
	I was just going to say mumble louder when you started talking.	
59:??	So, is there a button to turn off all the sound, all at once, or do I have to go through each thing? Hmm...	87
1:00:??	[Token Expiry event] But they said they were going to expire tomorrow.	

D.5.2 Exercises

Time	Transcript	ID
4:15	[Searches for "One or More Tasks Unavailable" event]	17
6:??	[Hoarding Details] Hmm... Tasks currently defined. Well, there's a lot of them defined. Which of these are hoarded? Editing. Tasks. Eh. Alright. Which of these tasks has highest priority? Better get that SOSP paper out. SOSP16. The lowest? I was just there, is editing.	
6:45	Which, if any, are completely available? Oh. Which of these is hoarded? Well, OK, so editing, OK so editing is completely available. Things being hoarded are editing, letters, SOSP16, and Darpa Grant. OK.	219
8:50	[Hoard Walk Advice Request] Cost. 2070 seconds. Gee. That's, um... ah... There's 3600 seconds in an hour. So this is, uh, 3/4^{ths} of an hour. Alright, let's take a look here. Recommendation letters–	96
9:30	looks smallest, so maybe I want to suck those across first. So, editing is already in. So let's fetch the letters. [mumble] OK. So [mumble] let's look at SOSP16. So this'll take 83 seconds. So that's– a minute and a half. And the, ah, Darpa papers– oh– there's a bazillion of them. Nk. But I only need probably the tex file or the dvi. Let's see. He wants me to read it. A colleague asked you to review a submission for SOSP16. So let's see. Either postscript or dvi. DVI is the smaller. So if I just want to read it, I'll just get the dvi and then the letters of recommendation. Those seem to be.. the easiest way to go. OK... OK. I'm trying to fetch the smallest amount of, um, data	
	to do my task. I selected the SOSP dvi file because I just needed to review it and it is small and I selected the letters, um, because they are small. What about the programs? So, I think the programs are already all selected. I mean, that's what the hoarding cache shows. Let's go take a look. Um... Tasks. Yeah, see editing is already all the way in the, um, in the, in the thing.	228
	So we won't, we won't select any editing. So it looks like I can just suck down the letters, and the ah, and the ah, SOSP dvi file. Ah. I wonder if I have the thing for viewing. Am I smar– Am I that smart? Let's see. Um... Hoard. No, I don't want a hoard walk. I want Tasks. Tasks. Um. Recommendation letters. Well. Let's take a look here. Um. Xdvi. Oh good! So I have an xdvi viewer. Oh. It's in– in a.. In writing I have an xdvi viewer. Mmm... Well, so it looks like I need to get writing down. Let's see what editing has in it. Editing only has gnu-emacs. Hmmm... OK. So that, that's OK. So it, it looks like I should also set up to get the writing across.	
13:30	**Don't puzzle too much about the xdvi viewer.** OK. **OK.** So, I won't puzzle too much, but I'm going to change my choices now. I'm just going to get the tex file, 'cause I can edit that with, uh, emacs. Alright. So, let's ah, SOSP16. And let's not get that one– let's get the tex file. Alright.	128
16:15	[Searches for Miss Events] The "Weakly Connected Cache Miss Advice", um, is I think a thing that the hoarder brings up. Is a prompt that the hoarder brings up to figure out what to download. OK. I don't know, "Disconnected Cache Miss Advice". And their configurations. Tasks. Ah. OK. Repair. Reintegration. Hoard walk. Ah– Advice. Oh, well, there's a nice thing. Oh– OK. And it is set to critical... :	17

	Alright. Therefore I must have to save this. Commit. Commit. Commit. [Fix Weak Miss Event.] Commit. Commit. Commit	32
21:40	Hey! Where's, ah, programs you need to build coda will be defined as "Building Coda". Hmm... Well, look at all these things are sucked up all the way. How nice. Well, I don't see a "Building Coda." That doesn't seem right. Must be a subtask.	18
22:00	Let's go see. Ah... "Building Coda"! Ha. Ha. There he is.	
22:30	Programs. "Building Coda". Who– Who– Twice. That's clever.	22
	[Typing pathname] And w– want to add /coda/misc/gnu-comp/i386_mach/omega– Man! The Marquis de Sade came up with pathname– /gdb. Save.	94
	[Creating user data] So, let's, um, create some user data here. So let's call this, um, bug task. I assume they don't have a bug task here. Yeah. It doesn't look like they do. OK. I want. I want to select this pathname. Bad, bad, bad. Alright. /coda/usr/bovik/src/venus. Do you know if you're a directory? Why aren't you a directory? I want you to suck it– Hmm... It must not be a directory. Mmm, that's too bad. /coda/usr/bovik/	94 36
24:??	**Check for typos.** Coda– code– coda. OK. Let's look at this guy...	
	I wish I knew how big these things were. I wonder if I'm going to have problems with my cache now.	57
	This is called our "storm work". We save him. Save him a couple more times just to make sure.	59
27:25	"Storm work"– we want to– #1 [beep] priority [beep-beep]. Alright– You're all mad at us now.	
	Uh, oh– I think I did the priority too early.	
29:00	Uh. I wish I had pathname completion. I wonder if we do? 368 <Tab> Ha... Yeah. No pathname completion. Alright.	94
	Alright. Save. Save. Save. I guess he's saved.	59
34:00	"Writing with Scribe". Go away. Up. Well. Ah. Ah get out of here. Let's change your name. Scribe. OK. Uh! Oh, go away. Ah... Delete. Ah... Well, we'll just do that one. You go away. How do I get rid of him? Eh,.. we won't do that. We'll just play it safe [Tempted to use "Delete Task" button]	6
35:00	[Defines data definition named "New..."] Mmmm. Save. OK. So now– I need, um... [Looks for definition] Hey. Where'd it go? Interesting. Did I not name it anything? I must've not named it anything. OK [User saved definition under name "New..." and system did not complain in any way!]	97
37:45	Oh, that's right. So I gotta make sure I gotta run these things a couple times so that these guys know what's going on.	
39:15	Oh... well, it got everything. D'I get everything? Well, it got everything.	
47:45	[Looks at Help window to determine why Task beeped and CV not available] Well, that help isn't helping.	252
55:??	[weak miss query] Told it to fetch, 'cause it needed to tex the intro report OK. But now we have advice went green. It must have gotten the report.	253

55:50	Network is red. I must have– [mumble] tripped and hauled out the network cable. Reconnect the network. Help! [Clicks Help button] Great. It doesn't tell me what's goin' on. Well, what happened to the network? Geez. Come on. Tell me what happened to the network.	250

D.5.3 Debriefing

Time	Transcript	ID
	Under Control Panel, right, to figure out what events are going on. **So, when I ask you about the weak cache miss advice event.** I'd rather it just have all the events in one list.	17
	[cache management for hoarding] Um. So this right here. I go to Tasks. Right. And it tells me how my hoarding is going and what hoarding is happening. Alright. So I can tell whether– you know, if I have all my tasks set up right– I can tell whether I can leave.	
	So the biggest one [comment] I have is that when I go in here and define stuff. Um. When I push <Tab>, right, it takes me somewhere else. To me, it should do filename completion. Alright. So, I hate typing in filenames. And if I can't drag a file directly in there from the interface, so on the MacIntosh, I'd be able to just drag a file from the finder into here in a normal, kind of, Mac application. But in Unix, you don't do that, so I'd rather, I'd rather have file completion here, filename completion? Cause that's how I live my life. Right? I even set up my directories and everything so at least the first two characters are usually unique. [Experimenter discuss plans to use PATH environment variable] [Agrees that would be nice.] [Filename completion is orthogonal to PATH environment variable.]	3 94

D.6 N2

D.6.1 Tutorial

Time	Transcript	ID
5:45	[Reviews Intro7 screen]	
7:10	[CodaConsole pops up] We have just requested that the CodaConsole be started. Alright. I just saw something	

	pop up. Control Panel, Tokens, Space, Network, Advice, Hoard Walk, Reintegration, Repair, Task. And there's an indicators window bar at the top. [Not sure what is meant by a "window bar at the top". There is no title bar on the indicator lights window.]	
8:45	[Note that there is a "Help" button on every screen] OK. (So, I) don't really have the console interface up right now.	45
11:15	[Hit "Commit" button for urgency colors tab] And then I hit "Commit". Oh. OK. And then I have to close it. [Seemed to expect the "commit" button to do the "close."]	32
13:30	OK. So let's see. A task is a set of files and directories. So there must be some way to group things into a task and then name it.	
13:50	And I notice that when I move my, uh, cursor around on the, uh, Control Panel here– er, not the Control Panel but this indicator panel, there's a little gray bar that appears to show which one I'm selecting– before I click.	
14:30	[Task2 describes bottom third of Task Information window as showing those tasks that have been hoarded by the user] And nothing's been hoarded by the user right now.	
16:00	[window has 3 main sections– description of Task Definition window] OK. So Tasks and Subtasks are, um, the same thing I guess. Or any Task can be a Subtask of any other Task, or something like that.	211
17:20	[The definition for "writing thesis" contains...] Yeah. I guess at this point I'm still a little bit confused about the relation between Tasks and Subtasks since there's, uh, um, some of the same labels are in each. I guess it means that, um, a Task can be a Subtask or something as well.	211
19:00	[Looking at Data Definition – before meta-information explanation] This directory, and I guess the "immediate children" would be the files in that directory.	
19:45	[Resizes the Data Definition window] [User makes no comment about the ungraceful expansion...]	{4}
20:20	[Meta-information explanation– of "none"] So I guess you'd have the listing available and that's it. :[continues reading] And is that recursive?... [continues reading] Ok, so it's the entire, ah, recursively the entire tree.	
23:00	[In Data Definition Window] I see there's a [navy] blue highlight around, ah, the path I'm going to define now.	
	[User attempts to use xterm running tutorial] **Can you iconify that one again? I'll give you an xterm. Just not that one.**	
	[User types path and hits <Return>] Hmm... Is /bin not? It created a new one. Oh, I see. So why did it? Yeah, let's see. Then when you hit either <Tab> or <Enter>, the meta-information are will become enabled allowing you to choose the appropriate meta-information for this entry. No, but it also gave me a new one. But I guess, yeah, it's not gray anymore. OK. I thought it would maybe just pop over here and not give me the new one, but I guess I might want to use "none" as a default.	3
27:10	[Tried <Tab> and <Enter> to move around the meta-information area.] Does <Shift><Tab> work in reverse? No.	3
27:30	[Saves Data Definition]	59

I guess that window doesn't go away– it's just saved it somewhere under that name– "grant proposal."

Alright. Yeah, I guess I was, ah, expecting the "save" would make it go away. 59

29:00 [Re: automatic program spying]
Oh. Interesting. So it has to, um, watch it run for awhile. "By making this distinction between a program data and user data, we can let you share program definitions among different tasks. Because this–" OK. So. I guess if a program uses, a, file that's just a user data file and I put it in a user data set, then, um, it will ignore it. If I hadn't put it in a user data set, I guess, ah, it, ah, wouldn't ignore it and would figure out that that's something that I need, ta–, that needs to be included, ah, in the program definition.

30:35 [Defining a program]
/coda/misc/bin/scribe, and if I hit <Enter>, I should get a new one, and I do.

32:45 [While trying to delete "latex" program from "writing with Scribe" task, user hits "Delete Task" button.] 6
Oh, man, I think I just deleted the task. (Don't) hit <Delete> or <En, ah, <Enter> key. Oh, I'm sorry. <Backspace> or <Delete> key. Alright. OK. Yeah, I guess, yeah,
33:00 maybe I was expecting a button or something for deleting programs.

33:45 OK. So I didn't have to hit "Save As" even though I started with that and changed its 60
name. Hmm... OK 5

34:40 [TaskUnavailable causes system to beep three times.] 261
Hmm... Something just beeped at me– sounded like from behind me. Um. And there's a red thing. And task is– I guess bad things are now happening. Things are beeping and things are turning red. Um. [Reading] Now move on, move the mouse down to the hoarded task list– when you right click, OK, repeat for "hacking advice" and "hacking venus" tasks. Alright. [mouth noise] So, click or select and it's blue and I come down here and right click. And I added that guy. One. Writing thesis.
35:15 Two. Hacking advice. Oh– it's just, ah, counting the number of hoarded tasks I guess. 82
Right click. There's three. And each one of them needs that much space or something. "You've probably noticed that your machine beeped three times and the task indicator turned red. The interface has just told you that a task is unavailable. This is because your newly selected tasks have not been hoarded yet." Ah-ha. OK. "We'll get that shortly." And so, I'm not sure what this means yet. This is the total– I'm using 70% out of my total cache space and here I'm using 90% out of something. I guess it knows that "writing thesis" is a task that needs 46 Meg worth of stuff or something. And I've got 90% of that 46 Meg hoarded already. I don't know. Let's 268
see. [Goes back to reading explanation].

[Reads explanation of priorities.] 82
Ah! That's the one, two, thing

[Reprioritizing explanation]
So, I guess, like, if I wanted to change, ah, "hacking venus" to higher up, I would just selected it again and drop it. And I guess this one would go away and it would appear in some new place up here.

39:00 Ew... Everything's all flashing now. [Advice blinks red.]

39.35 So right now. I'm getting no advice and venus needs some advice.

39:45 So right now. The network is weakly connected, so it's yellow. Advice is red because 261
there's some urgent advice. And task is red, um, I guess because this task, ah. I don't know, some task is getting screwed up because it can't grab this file– or something. Or is worried about grabbing that file over a low bandwidth.

41:30	So, yeah, if you're away from your disk for an hour, then it might as well go ahead and grab it.	
42:30	[Weak miss timeout prod] Yeah. I guess the two "No's" in a row momentarily confused me 'cause [mouth] it seemed like two things both saying "No" but then I saw that they– they say different things.	98
44:30	Ahhh....! Yep, I'm getting advice [Hoard Walk Advice request arrived]	
44:55	Oh. And it stopped flashing. It was flashing and it just stopped flashing. So I guess maybe the flash is just to attract some immediate attention and then it just goes solid.	33
45:30	So, I guess I can get advice from both Hoard Walk or Advice in the event that the advice has something to do with the Hoard Walk.	10
47:20	[Hoard Walk Advice window] And it looks like it's gonna be a tree– it looks like, like a little tree widget gizmo.	
48:30	[Note that the costs do not add up.] That's true. There must be some overlap.	
49:55	[Expands each of the nodes; sees Editing/Building Coda in multiple tasks.] I wonder why these are in, ah, italics and these are, ah, not italics. I guess presumably italics means shared perhaps. There's "Building Coda" and if I selected on– yeah, I guess that must mean anything shared and the non-italics are the unique ones.	
50:40	And task is now flashing red. Oh, and the Hoard Walk Informat– this thing came up from– I didn't notice this come up, but it popped up somewhere.	
51:00	I wonder why Task is flashing red and then stopped. Huh! So, I guess– ah– 'cause it was red before 'cause not all the tasks had all the files they needed	33 261
51:45	[Space Information] Now let's see. Is that available or used? ... Ah... "The window shows the status of the available cache. The available disk space on the partition of the local disk where the Venus cache resides. And the available RVM space, an internal data structure." OK. "Should any number exceed a reasonable threshold, the instructions will tell you what corrective action to take." So I guess we're using 16% of the local disk, is how to interpret that not that there is 16% left. Yeah, that's because it– as it fills up, the bar fills.	271 280
	["Notice the Space Information window disappear and reappear."] I didn't know– I guess it just went dung-dung, flashed really fast or something. Excellent.	
53:20	OK. The cache is filling. Be careful not to... [Note: User went straight to the bottom half of the Help window on his second exposure to it.]	280
54:40	I guess, ah, maybe Tokens would start becoming yellow when it's getting close to expiration and red when it has expired.	260 257
55:15	Let's see. Is there an events thing [indicator]? [Looks at indicators]... So I guess we're going to be bringing up Control Panel. What's a configuration? Yeah it says that.	
56:40	[Discovers trigger descriptions.] OK. That's pretty nice. I like that.	
58:15	Read disconnected... So I guess that means that I can write back to the server, but I can't read or something. Oh yeah OK. It's going to tell me. [Reads trigger description.]	

D.6.2 Exercises

Time	Transcript	ID
3:30	["One or More Tasks Unavailable" event] Let's see. Under what conditions will the "One or More Tasks Unavailable" event occur? Uh... Let's go to, ah, Tasks. Um... Ah. Let's see. Where was that dang thing? Was it under Control Panel? Event information– yeah, OK. Event configuration. Um. One or more tasks unavailable. Um. I guess that's tasks [Finds it]	147 17
5:40	[User uses the sliders widgets to see everything.]	
6:40	So that's why this thing is red– task indicator. 'Cause three of them are out.	
7:15	Oh. I'm getting Advice and Hoard Walk flashing at me.	
8:15	OK. So, Hoard Walk. Ah. I think I can get Advice from clicking on this guy as well. Yep. Hoard Walk Advice. So, um... Let's see. So what happens when I click on Advice– just for the hell of it. Hoard Walk Pending Advice. That probably brings me to the same screen. Both recommendation letters and papers, I assume, are going to need editing– need the editing guy. Yep– So no matter what I do I need that stuff. So, um, if I'm going to do anything, I need those. Then, and that's going to take me 7 minutes. : [User chooses biggest file (.ps) because he knows how to read it.] I don't remember all the tex and dvi stuff. So, even though this is, like, the biggest file, it's the only one I happen to know how to read. Ha! Ha! Ha! So I would pull that guy down and look at it. Maybe I don't care about the diagrams. I don't know. Do I need the diagrams? Eh... I don't know. :[letters of recommendation] Um. Eh... How does that change my time? Oh– hardly anything. I mean, really, ah, everything else is small... it's like the editing guy is the big guy. It takes a big hit. [Discovers tex-mode and decides he must be able to read paper as .tex file so he doesn't need ps.]	
13:15	Number of objects, -2. I don't know. Some weird thing happening. I'll ignore it for now.	99 124
15:45	[Describe Miss Events] So I brought up Control Panel to look at the events. Uhm. So, ah, "Weakly Connected Cache Miss Advice". Is that all? I think that's Network stuff. No. So what else? Space? Advice. Oh OK. Makes sense.	17
18:43	[i386_mach box] Ah... second. OK. I've never used a Mach (box?), so I'm not really sure what that means. But 386 is a relatively weak processor, so I guess a not very powerful machine.	
20:20	Let's see. We don't have a fixing coda bugs thing yet so I guess we'll make a new task.	
20:45	[Looks for "Building Coda" under tasks] Do I have a "Building Coda"? Oh– That would be programs. Right. "Building Coda" is programs, not a subtask. Editing contains gnu-emacs. OK. So, let's close that guy. So that would be the same thing as just including gnu-emacs as a program would be to include editing as a	18

subtask, but I'll include editing as a subtask seems, ah, slightly cleaner.

23:05 [Creates Coda source, data definition]
/coda/usr/bovik/src/venus, tab over, and I want the immediate children. I guess I'll check to see if there's like subdirectories I need, but... {57}

24:20 [Attempts to click on meta-information without having hit <Tab> or <Enter> first.] 20

25:40 Coda bugs. I'll save. Did I save? I don't know if I saved. Um. I guess I'll click save anyways. 59

26:40 [Interesting header files]
So I wonder. Let's see. Source files. Would I want to include them in my "Building Coda" programs? I think I– I thought I would want to put them in my, ah, user data section. Hmm... So I would actually want to make, like, a, ah, "Coda includes" user data. Mmm... Coda headers. Building Coda programs. I'm a little confused about 100
the notion of putting these header files in with the programs instead of putting them in user data. No. I'm going to go ahead and, so when I was looking from home, so when I fix Coda bugs, I want to have those header files around too. So I put the bugs in the source. So now let's, ah, make Coda headers and I only want these few.
:

27:50 [User attempts to use arrow keys to move around the meta-information area of the screen.] 101

29:15 [User adds headers to bug fixing task.]

32:20 [Attempts to delete selected object from contains list using "Delete Task"]
Ew! Dang it! I deleted, ah, the writing tasks I guess. Rats! I wonder if I– I was 6
meaning to delete this, ah, recommendation letters that I stuck in fixing Coda bugs. 62
Oh well. I wonder if I say– if I just don't say "save". Oh– what if I say "Revert"?
Probably not. I think it's gone. What if I don't say "Save"– I just go boomp. No. {92}
Poo. Alright.

35:50 [Prioritize these tasks.]
OK. So I see. If I separated the header files– Right now they're stuck in the fix bugs thing. So what I really want to do maybe is put them in a, ah, separate thing that I can give a low priority to.

36:?? [Deletes selected task– remembers not to use "Delete Task" button.] 6
I hit the <Delete> key. Hey, Ha. Ha. 62

37:00 [unhoarding tasks] 102
Yep. Selecting them and deleting works like I expect.

37:20 [TaskUnavailable event causes system to beep three times.]
OK. It's beeping at me because it hasn't, uh, all those things I need to do a cache walk, or whatever that's called, a hoard walk in order to get everything in there. 220

And "writing CV" is the highest of all. I wonder if I can slip it in there. No. OK. 42
Hmm... I wanted to stick it above the top one, but ah. So I'm going to delete "fixing coda bugs" and try to stick it between these two.

38:00 And I guess all of them, it's telling me, are 37 Meg. Uh. All of these tasks take 37 Meg. And I probably have about 30– less than 33 left 'cause I'm at 66% of 97. So, something's gonna get left behind– that's for sure.
:

38:25 Ah, except I'm at 64% of each of the 37– Ah! I don't know. What I'd like to see, I 268
guess, maybe is something telling me, um, for all of the hoarded tasks I have now how 273
much more cache space would I need to have all of them in there. So that you can, 15
you know, see exactly where– who's gonna start getting left behind– like what task

	isn't gonna end up having enough stuff in it.	
45:05	[User remembers RVM problem from tutorial.]	
48:40	[TaskUnavailable event causes system to beep three times.]	261
	Basically, it doesn't have any of the stuff for "fixing coda bugs." That's weird... I guess that's all that means. Let me look at, ah, um, Control Panel and see what, ah, is indicated under Tasks. "One or More Tasks Unavailable". So, yeah, it must be the	207
	"One More Tasks Unavailable"– beep and critical. OK. Well, I guess I would have to do, um, one or more tasks that have been selected for hoarding become unavailable. Now why would they become unavailable? OK. Let's see. If it was available, what	29
	would make it become unavailable? ... So, I guess I would just do a hoard walk. **Actually, please don't do a hoard walk.**	
50:00	Then I saw another change, which is this hoard walk guy flashing up. See it's yellow. So yellow means it's doing one right now I guess. Starting one. So it's like a warning level thing. Ah– No help on the hoard walk information– Oh! OK. That's where it is. Down here. "This is the progress help text." [Hoard Walk finishes.]	249
52:30	Ew... The Network just went to pot.	
	:	
	[Tries Network Help to determine what to do to fix the problem– help doesn't suggest anything.]	250
54:??	[Weak miss advice.]	
	Let's see. I don't remember what it is I'm trying to work on right now– oh yeah I'm working on my, ah, thesis. I'm processing it through latex.	149
55:??	Ah... we just went off the air– zero! Alright. What– What can you do in the event of a– is it, like, do I already have, like, the stuff that I need, ah, let's see. Yeah. If these are what I'm working on, except what I really said I was doing was working on my thesis so I guess I don't have those hoarded right now. So, if I had them hoarded, it	
	wouldn't matter that the network just dropped off. And I could ask it to hoard it, and then, ah, Do I already have a, ah, "working on my thesis" task? No, I don't. Pooh. So what would I do in this event? I guess I could ask it to hoard it, and hope that when the network came back on, that I could then grab it and then if the network's being flaky. Neh. I don't know what I would do in this situation. Alright.	191
58:20	[Disconnected cache miss event.]	103
	So that seems to indicate that even though I haven't hoarded it, um. Coda is going to, ah, try to keep track that of that was an important file to me and grab it for me in the future. So that I don't have to explicitly define a, um, explicitly define a file, I mean, explicitly define a task.	
	[User doesn't know bovik's password.]	
	[Submitting password takes excessive amounts of time to return.]	104
	[Doesn't know abount reintegration.]	239

D.6.3 Debriefing

Time	Transcript	ID
	[What to do if you find yourself weakly connected? **Define a task, hoard and disconnected. Call the operator.** You define tasks even when you're strongly connected. **Not in the past– pmax binaries weren't in Coda. Perhaps in the future.]**	

D.7 N4

D.7.1 Tutorial

Time	Transcript	ID
	When you name things, do not use funky characters. Stick to the alphabet. There is a bug that causes a catastrophic failure if you use a certain funky character. And by funky character, I mean anything like this [points to numeric keys]. Okay. Just use the alphabet. Dashes are okay. Like single dash. This one. [points] This is okay. Thank you.	
	[AFS outages.] Yes, that's frustrating.	
	Would you mind reading a little bit more out loud? OK. **Thanks.**	
8:10	[Choosing urgency colors.] Mmm... Uh Too Dark! Hello. Maybe dark for normal is good.	264
	: I guess that's as light as red gets... Hmm. But now the warning is much lighter than the red. Well... I guess that's OK. Hee. Hee. Maybe if I make it blue, it would be better. No I think I'll remember yellow a lot better than blue.	266
	[Notice Network indicator is now yellow.] What if it wasn't yellow that I changed it to?	106
15:20	"If the pathname is a directory, the definition also contains some meta-information: none, immediate, or all descendants. If the meta-information is set to none, then only directory will be included." OK– contents, but not the children– children directories. OK– Great. Um. "Immediate children, then the directory and its immediate children will be included." ... "Look at the directory contents and examine any children." Oh. OK. So, that is children– [mumble] You can list it, but not read anything. OK. So	

	there you can list it and read anything.	
	Yeah, if you had links in there, that could be really big.	93
18:40	[User successfully uses keyboard accelerators.]	
19:55	[Automatic program hoarding – rereading instructions in Task13.] I guess because they are already there.	
21:40	[Types in pathname. Hits something (<Backspace> or other key near <Return>). The entire pathname then disappears.] /coda/misc/bin/scribe. Can't type today. Ew... Where'd it go?	107
24:20	[TaskUnavailable event causes system to beep three times.] Oh! Now what was the beeping?	261
26:00	[Reprioritizes a task.] Hey! It worked.	
27:00	[Note that you are weakly connected to the Coda servers.] That must be the Network thing.	
27:35	[Two main sections in Advice Information window.] Ah! So the advice goes two ways.	{115}
28:30	[Go ahead and fetch if user doesn't answer weak miss query.] Unless it costs money.	
31:05	[Hoard walk advice request.] Oh, there's things blinking there.	{9}
32:05	[Hoard walk Advice window.] You can also see an area labeled network status. This area shows you the average network bandwidth as a percentage of the max– maximum Ethernet bandwidth. OK. Eh. That almost needs a log scale.	50
	You can compress the node in a similar fashion. What? Oh! Nnn... OK. I see.	
34:30	[Expanding nodes to see what's there.] Boy, these things go down forever, don't they? : I like the minus sign. Cool! Ok. The minus sign is nice.	
35:15	[Space indicator will soon indicate an urgent problem.] Ha... I feel like I'm on Apollo 13 here. [Reads Space Help overview. Then RVM explanation.]	
36:30	[Space problem solved. Space indicator will indicate a potential problem. Rereads the Help window overview.] I already read that part, didn't I? [Moves on to explanation section.]	
38:10	[Notification of events.] It could be just like AFS and give you an error when you try to Save when your tokens are gone and you have no idea what causes it.	
	You should now be looking at the Event Configuration tab of the Control Panel. Did I forget to do something? Hmm... I must have.	
39:15	[...whether or not the bell should be used...] Oh, I hate bells.	
	Under the Advice indicator, please select the– wha? Hee. Hee! OK. Advice indicator. There.	108

D.7.2 Exercises

Time	Transcript	ID
	[Experimenter reminded user not to use "funky characters".]	
2:25	Uh. There's no interface up. Something's wrong. Let me. Let me... Hmm. **[mumble] There's now an interface [it showed up in the wrong "room"], but you missed that first question.** Hmm... **Did you miss a question? Let me check if you missed a question. The– so the first one was "please choose colors for the urgency levels** Oh, colors, OK **Um.. pick colors you can see and remember. If you are happy with the default colors, you do not need to pick new ones. OK?**	
7:20	[Hoard Walk Advice Request] [The Advice light turns green as soon as user opens hoard walk window.] Cost in seconds... Seconds is kind of a small unit for this um. Editing. Don't we already have all of editing?	120 96
	[User attempts to use xcalc unsuccessfully.]	
	[User has no difficulty finding Miss Events.]	
	A blizzard is presently hitting Ohio. Oh... It's May...	
24:05	And you also need gdb. Ah... I need to make a new one. gdb. Contains, ah, /coda/misc/gnu-comp/i386_mach/omega/bin/gdb	
	[User Data for Coda Sources.] And I think I probably want the "immediate children", because otherwise I can't modify anything. OK. Although it's possible I would want "all descendants".	
28:30	[Header Files Task] Let's fix this task. Ummm... Um. Hmm. I don't know whether to put them in programs or user data, but I guess that, ah, maybe they go in programs. OK. Which means that I would, ah, ... yeah, they must go in programs because that way it'll sniff out everything they need. But that's only if they run. Well whatever, I'll put in	100
29:30	anyway. Well, they're not executables though. Um. I'll put'em in user data. That sounds good. Yeah. OK.	
	[Creates header files data.] And, some include directories have subtrees, but I'll assume these don't.	
31:30	[User adds header files to bug fixing task.] OK. Save that. And done. You need to write your CV before Tuesday. Hmm. I wonder if headers should have been a different task. Oh well. Too late now.	
36:30	[Prioritize these tasks.] Yeah, I think header files should have been a different task. Let's see...Well that's not too hard to change.	
37:25	OK. Coda bug fixing is gonna... I guess I should take out these old ones, too. I wonder how I do that. There we go. [Figures out how to unhoard without consulting Help.]	102

[TaskUnavailable event causes system to beep three times.]
Oh– Now it's complaining that I don't have all of bug fixing ready. Right.

Please request that a hoard walk be performed immediately. OK... Hmm. Well, whatever. Let's see. Finish Hoard Walk. Hoard Walk Advice here. Hmm. OK. [User never completed the prevous hoard walk advice request. Thus, the request remained when she attempted to start the hoard walk. She did the most reasonable thing– clicked the Finish Hoard Walk button. She seemed somewhat puzzled, but not terribly so.]

[User reads incident report directions while playing.]

[User closes incident report with ☒ rather than "OK".]

I wonder if space should still be red like that.
:
Oh, that went off. I wonder if that's an event.

52:20 [TaskUnavailable event causes system to beep three times.]
Mmm. Something beeping. OK. Uh. So, what is the problem here? Probably that you can't write the CV. Ah. Let's see, the "writing CV" task is not in the cache. Ah. Ah– I guess you could fix this by doing a hoard walk 'cause there's still some cache space. Although, not enough.

Mmm... Hoard Walk is yellow. Why are we yellow? [User took a long time to answer the Task Unavailable report so the Hoard Walk Begin and End occurred almost simultaneously.]

D.7.3 Debriefing

Time	Transcript	ID
	[Questioned about pathname disappearance which occurred at 21:40 of Tutorial.]	
	[Questioned if she had verified @sys!=i386_mach. She had not– just checking to see if /coda/misc/bin existed.]	

D.8 P1

Pilot user P1 was not recorded on video or audio tape. For this reason, there is no transcript of her session. Her primary purpose was to critique the tutorial. Her comments were reflected in the version of the tutorial used in subsequent runs.

D.9 P2

D.9.1 Tutorial

Time	Transcript	ID
	[Playing with Event Configuration] Ew, Cool!	
	[Examining Event Configuration for "Operating Disconnected"] ...Operating Disconnected. So I get notified. [mumble] urgency critical, which means red. No pop-up. No beep. But it flashes... Now flashes happe... I don't know what flashing means. Flashes... The network in.... it must, must turn this on and off. You know, so like blink. Maybe. Attract your attention.	221
	[Space indicator changes to red] "...an urgent problem. If you open the 'Space' indicator window..." Ew, it does, how 'bout that– cool. "...you'll notice..." That's what flash must mean. OK. Flashes between green and red. OK. Wonder why green? Anyway [continues with tutorial].	9
	[Description of Meta-information] "...be able to look at the directory contents, but not at any of the children." Child directories must be. So I can see the files there, but not the children. Alright. "If the meta-information is set to 'immediate children', then the directory and its immediate children will be included. This means that you'd be able to look at the directory contents and examine any children." Maybe I was just wrong.	
	[User clicks "Save As" button for window below the one he thought he was saving.]	

D.9.2 Exercises

Time	Transcript	ID
	[Please describe the "Hoard Walk Pending Advice" event and its configuration.] [Reads question] [rereads questions] So I have to get to that I guess. So let's take a look at it. No, that's the advice– that's the wrong one. It was in... No, it wasn't Tokens– what was advice? No, it's not advice. It was. It must have been Control Panel. And-it-takes-a-long-time– Event configuration. So we're looking for hoard walk... pending advice. OK. Please describe this event and its configuration. OK, con-copy. No– too bad. This event is triggered when Venus has a hoard walk temporarily waiting for advice...[describes configuration].	108
	[Please describe Miss Events] So, weakly-connected and disconnected... I think are both Space. Is that true? No, they're probably Network. Operating strongly connected. No. It's... weakly-	17

connected cache miss is in advice. So, I remember that– but not exactly.

[Please describe our connectivity to each of the servers.]
Must be strongly-connected; otherwise, we wouldn't have a, um, little thing there.
[double-clicks on Network indicator]

[Hoard Walk Pending Advice] 155
Uh. Hoard Walk. OK. It doesn't say, what, um..., it's not asking for advice. That comes under the advice thing and nothing comes up. So, wait a minute. Let me. I'll close all these things. OK. Has requested advice. Please examine the request. Decide.
[double-clicks advice indicator]
There's no advice. [clicking] I need help. There's no advice here. [double-clicks on Hoard Walk indicator] Hello. Hello. Hello? 156
16:00 **Harry?**
Yes, I need help.
Experimenter is on the phone with lab administrator trying to figure out why the sound is not audible from monitor in the equipment room.
What's up?
It says to um.. uh... Just a minute ago you requested a perform has requested advice. And there's no advice being requested.

Well, but this is the, um, hoard walk. So the hoard walk advice comes in under the hoard walk line.
Oh! OK.
OK? Make sense? 155
Except I don't see any advice being requested here. OK. I did look here, and I didn't see any either. But maybe I'm looking in the wrong spot.
Try, try hitting Help.
OK, Hoard Walk advice... [reading from Help window].

[Mobile 95 Paper Task] 157
That seems silly to create a task for one thing.

[Save Confusion...] 158
Oops! I need to go back. I didn't save it. [mumble] Writing and thesis. So, it saves automatically if I close it. That's interesting. OK... Without even telling me. That's, I guess, a feature.

[Hoards a task that's unavailable– task indicator beeps.]
I tell you I would turn that beep off really fast.

[Transcription stops as the user begins setting up for Phase Three.]
The user was unable to complete Phase Three of the Exercises because the experimenter had set the timings such that they were extremely short for testing purposes. Thus, all of the events occurred within about 10 seconds! The experimenter attempted to recover from this failure, but was unable to do so within the user's time constraints.

D.9.3 Debriefing

Pilot 2 was not debriefed on tape.

D.10 P3

D.10.1 Tutorial

Time	Transcript	ID
	[Introduction5: "hoarding"] Uh. Hoarding. Um... OK, so user specifies and prioritizes tasks– not files– that he or she plans to work on during an upcoming disconnection. Now that's interesting. Um... I wonder what the mapping from tasks to files is like.	
	[Introduction9: Introduce Dashboard Indicator Lights.] ...dashboard indicator lights of an automobile... Well, maybe that's a reasonably successful interface... [reads screen out loud]. OK, familiarity – I'd buy that.	
	[First impression of CodaConsole indicators.] Up, there it is... Um... This window contains our version of indicator lights. There are nine different indicators. OK... I guess indicators equals text here. The names of the indicators are: Control Panel, Tokens, right, right, right, right, right. Ok, they're all there. Uh, we've purposely made the indicator window as small as possible so users will be willing to participate. Right, well thank God you haven't used little bizarre pictures whose mapping I'd have to figure out. Um...	
	[User cannot figure out where to double-click.] **Please double-click on the "Control Panel" indicator light.** Oh, that thing. OK... **I will add the word "light".** Right. **Would that help?**	292
	Right. I also thought it was a label, not a button so I was a little confused. I thought it was just a passive indicator of state, not an active thing that I could click on. **OK.** OK. Now I see what we're talking about.	159
	[UrgencyColors2] Interesting choice... So I'm getting all the color names twice. Once as illustration; and once to actually let me choose it. I wonder why they're not the same list. [*We added the "For the record..." comment to address this confusion.*] : My current choices are hideous. Let me put it back.	
	[EventConfiguration1] Boy, I wish this thing told me, were actually in the colors of Normal, Warning, and Critical so I don't have to remember that that's what this association is.	160
	: This is yes-or-no. Boy, that would be better as a check box. So would that...	161
	: This is the only one for which I have a choice. I'd rather have that as a radiobox.	161

[Event Tab] 162
Let's see. Something strikes me funny about the layout too, because... These events are relative to this indicator and these configurations are relative to this event, but nothing about the layout shows me that. I'd almost rather have like nesting, where the configurations were nested inside the event stuff and the event stuff was nested inside the indicator. Although this part of the screen never changes, so that's kind of a little bit. Humm... Um...

[Control Panel]
These are indicator lights over here. So this is supposed to notify me when things are 163
happening with respect to Tokens, respect to Space, respect to Network, whatever. Except Control Panel seems to be different. Control Panel seems to be there merely to pop-up this. It doesn't seem to be an indicator– maybe it's an indicator light for something, but... [chuckle] It's an indicator that I need to interact with the Control Panel [chuckle].

[Event Configuration meanings] 164
Boy, you know I really wish. See, I had to puzzle out to figure out that... that this would mean, so if this event happened I had to puzzle through all this text to figure out that that would mean flashing red. If, in fact, I've gotten it right. Boy, I'd sure appreciate a little sample here that showed me something flashing red so I wouldn't have to fig– puzzle that out.

[Space indicator flashes to indicate RVM problem]
If you open the space indicator window, uh... You'll notice that the RVM space is critically full. Opp... There it is. It's flashing. OK. If you click on the hel... OK. So. If you open the space indicator window– I guess that's double-clicking again 46
[double-clicks]. There it is. OK. It [the Space indicator light] stopped flashing. 33
Uhm... That stopped flashing. There's nothing about this that's flashing. [chuckle] The impression I'm left with is that, by opening this window, somehow the danger has just gone away. [chuckle]

[Space Help Window]
Oh... So, the help button isn't just relative to the Control Panel. It's actually smart enough to know what's going on as well. That's pretty cool.
:
I see... OK. So, this part of the help seems like it probably always says the same thing, but the stuff below the line is relative to what's going on right now. OK. That's pretty cool.

[Network Light changes color]
It's a little strange having things happen, like, well, maybe not, I mean that's that's what this user interface is like. Things happen behind your back and so this thing changes all the time anyway so I guess having the tutorial change it is not so weird.

[Advice1]
"The job of defining these tasks is sufficiently difficult that the system needs to provide as much help as possible to the user." Difficult, difficult to do something I'm supposed to do all the time doesn't sound good. "Furthermore, the job of adapting to changes in network bandwidth of three or more orders of magnitude is sufficiently difficult that the system occasionally needs advice from the user." OK, I hope it's only occasionally. [Reads more of tutorial.] "Rather than automatically performing a fetch that will take a substantial amount of time..." I wonder who defines what substantial 245
is...

[Advice3]
"Since a process is waiting for this file request, it is important, ah, important that the request for advice receive immediate attention from the user." Mmm... So because some program wants it, I'm supposed to... Hmm... This seems like I'm the slave of

some program rather than the program being some slave of me, but presumably this is some program that I am currently using and so I care about whether it completes what it does or not.

[Configure an Event: Commit]
Huh... I wonder if I close without committing whether it'll tell me. Nope. Apparently that was abort. Or maybe it wasn't. Let me check what it says now... Um... Let's go back to advice. Weakly-connected cache miss. Nope, it's not a pop-up so apparently that was a cancel.

[Pop-Up Weak Miss] 201
Of course, the thing about this window is even though it popped-up, it's not especially attention getting– it looks just like all the other windows on the screen. So, if I weren't actively looking on my screen at the time that it popped up– 'cause I was on the phone maybe or talking to a colleague, I'd bet I wouldn't notice that.

[Begin Hoard Walk– Click on "Walk Now"] 258
And now it's yellow. Why is it yellow? Yellow means it's busy in this case maybe? I don't understand what yellow means here. And why is network yellow? Maybe that was the last exercise.
:
Now it's flashing red. Oh, boy.

[HoardWalk5]
I see... So, a task is sort of like a directory, but rather than having everything nested inside of it, presumably a task can consist of files and directories from all over the file system. Oh, like maybe the executable I'm using, like a compiler, plus the files I'm compiling. Ok, maybe that makes sense.

[Exploring Hoard Walk Advice Request]
Huh... That's interesting so I wonder if, instead of having it in my mind, that for sure I want to do activity X right now, I wonder how easy it would be to answer the question: Uh... What set of activities could I choose that would minimize the files that have to be brought over. I'm willing to do anything right now, but I can't afford to bring many files over so bring me over a few files that will let me do as much as possible. Maybe that's a weird thing to ask. I don't know.

[Hoard Walk indicator turns yellow as it performs Hoard Walk] 258
See. I'm totally baffled by what yellow means. I mean, yellow means busy, as far as I can tell. Not warning. 'Cause it isn't dangerous that it's, that it's fetching the files that I asked it to. It's good that it did it.

[Task3]
Ah..hmm... Subtasks, programs, user data. That's interesting. I wonder why the system cares about the distinction between programs and user data. Why isn't it just bits to it? Maybe the thought is that I care about the distinction somehow. I don't know. I'll keep reading.
:
Oh, I see. So I write. I see, so there's some specialization going on here. So writing might be just, maybe, my editor, but writing thesis might be writing, which contains my editor, plus the files that consist of my thesis or something. That seems pretty handy. Oh, and I'll bet you get a kind of multiple inheritance here because maybe I have writing plus latex or something. Huh! Cool. OK.

[Meta-Information] 57
"...he knows that this is a deep tree and that hoarding all of it could be a problem." Huh. It's kind of too bad that he had to know that rather than being told like how many bytes are contained, in the uh, for each of the choices or something.

1:12:00 Huh, I just thought of something. If I'm busily, oh, oh, oh, if I'm busily hacking my thesis or some code and I'm adding new source files all the time and I want to hoard them... OK, so, the thing I would do is name a directory here and pick immediate children. Right. So there's no, sort of like, problem where I have to constantly maintain this thing to add new source files in this dialog as I add them in my editor. 'Cause I can select a directory here plus its children. So. Good.

[Task13] {61}
Oh, maybe that's why you want to know the difference between programs and user data is because you're going to check the executable bit for me or something on these program definitions, whereas user data you don't care whether the thing is executable.
:
1:17:00 Oh, and the other thing is, here you know I just want a file so you don't have to ask me all those meta questions. Right. OK– this makes sense.

[Task14] 246
1:17:36 Really?! I don't know that I believe that. What if this were a shell script. Are you going to grep the shell script looking for the things it touches. I'm not sure I believe this claim.

[Click "Save As"]
1:19:15 Again, I click "Save As" and it gave me a new thingy here. I don't think I was supposed to get a new thingy. Now, I get the right thing. A program set named Scribe Tools is already defined... Hmm... 37
[User has clicked "Save As" twice in a row.] 59

[Task24]
1:25:00 The "Tasks" area contains... Uh. OK. The tasks... Hmm. The "Tasks" area contains a list of all existing tasks. What was this other box? [double-clicks on Hoard Walk indicator] Was it Hoard Walk? There's some box that, that I was just working with that did something remarkably like this. It had the hierarchical kind of thing. Oh, well. 31

[Task24: Bizarre form of drag and drop.]
I see. OK. You're right. This is totally bizarre drag-n-drop. OK.
:
OK. Well. It's weird, but it works.

[Task25: Prioritize]
So, toward the top apparently means... Well... a smaller priority number. I don't know if that means higher or... oh, you just told me. The item numbered 1 will have the highest priority. OK. So lowest number equals highest priority. 82
:
I see. The priority here is really top to bottom. The numbering is sort of irrelevant. And, if I want to rechange the priorities, I have to change their order in this list.
:
I wonder what the priority of the tasks up here [all tasks list] that don't appear down here [hoarded task list] are. I guess maybe they have, like, no priority because they won't be hoarded or something. 242

[Token Expiry]
Okay, my tokens expired, which is what that beeping was. Now it's yellow. Wait. If it was important enough to beep. Oh wait! All of this is sort of my fault. I could go into the Contr– Control Panel and, and, and change the color if I wanted. So I guess, yellow plus beeping was my choice. Other than red plus beeping seems like a smarter choice, but... 276

D.10.2 Exercises

Time	Transcript	ID
	[Please describe weakly connected and disconnected cache miss events and their configurations.]	
	Umm... Weakly connected cache miss... That's not here...[Hoard Walk indicator]	17
	So... Uh, maybe Network [indicator]. Um, well that has operating weakly connected, so that's no good. Um, maybe Space...since it has to do with caches. Nope. Not there either. Where is it? Uh, how about Network? Oh I already tried here. Um, Reintegration, Repair, and Task. Well, to heck with that, I'm just going to try one at a time then. Um, there we go. Weakly connected cache miss. I don't know why this is under Advice, but it is.	
	[What is our connectivity to each of the servers?]	
	Oh, let's see. Um, well, they're all green so I'll type down that they are all green so I guess we're strongly connected to each of them.	
	[Which, if any, are available?]	268
	Uh, well let's see... Editing is definitely available 'cause it's there and it's green. I don't know if we're supposed to count SOSP or not. It's there, but it's red. Oh, well, I'll put that SOSP16 is there...but barely.	219
	[Please examine hoard walk advice request. Decide what objects should be fetched at this time and describe your reasoning below.]	
	So, let me look at hoard walk, by double-clicking on it. There it is. Um. So what am I doing again... Hoarded two tasks... Um... Am I in the right place? Let's see. What did this say? We've selected two tasks to be hoarded but you've recently made a connection. Just a minute ago you requested that a hoard walk be performed. OK. Now you notice that the hoard daemon has requested advice. Uh... Hum. Maybe I want advice instead [double-clicks advice indicator]. No, there's nothing here in advice. Oops. Um, which make sense cause it wasn't blinking or anything. Um... Um... Just a minute ago, you requested...right. Now you notice that the hoard daemon has requested advice. Please examine the request. Decide what objects should be fetched this time at this time. Describe your reasoning below. Hm... [Double-clicks on task indicator] So, I clicked on. I'm basically confused... So I'm trying to puzzle my way out of this. Umm. I'm confused because... Well, let's see. This mentions... Let's see. This mentions the hoard walk being involved so I brought up– oh, rats, I hate this window manager– I brought up, rrr, it's window, but I don't see anything sort of relevant. Um... It seems to be requesting, I've been told it's requesting advice, but the advice panel doesn't have anything. Doesn't say anything about advice here. There's no, like, nothing in the "Advice Needed" window. Um.. Hoard Walk Information. I don't see anything really relevant there. Um... [click on help.] I just asked for help in the hoard walk information box. Hoard Walk Advice Window– oh, OK– allows you to give advice to the hoard daemon about what tasks should be fetched. The upper left corner contains information about the status of the cache. Right. The upper right corner contains information about the network status. OK. I remember that. Um. Information about the number of objects to be fetched during this hoard walk and the amount of time those fetches are expected to take. Humm... OK. The main body of this window shows tasks for which one or more objects need to be fetched. These tasks are shown hierarchically. OK. Well, what did the tutorial thingy say again? It said you have selected two tasks to be hoarded. You recently made a connection. Please examine the request. Decide what objects should be fetched. Describe your reasoning below. Hmm. HmHmHmmm. So I don't think I want the task information 'cause that's just for saying what files are associated with	155

what tasks so I don't care about that right now. Um... Let's see... Maybe I'm supposed to start the hoard walk. Let's see what saying fetch does. [Clicks on "Fetch" button in Hoard Walk Information window.] So, let's see. Well....
Experimenter stopped this exercise because once the "Fetch" button has been clicked, it is impossible to complete the exercise. 236

[Please define a task for writing your CV.]
Well, since I don't know how, it's my advisor's directory, I don't know how big it is so I'll do the conservative thing and pick immediate children. I still think it'd be nice if immediate children vs. all descendants gave me, like parenthetically, how much data that was off to the side. 57

[User hoards newly defined task. Task indicator blinks and turns red.]
OK, well, now it's beeping at me. Um... I wonder why it is beeping at me. Um... OK. I double-click on it and it's giving me its task definition, which doesn't tell me anything about why. Um... Oh, maybe I'm running out of... maybe the problem is I've over-committed myself in terms of cache space and that's why it's beeping. Huh. Oh, well. I'm going to ignore that for now. 261

[Please define task for reading paper.] 157
This seems like an awful lot of trouble just to get one file onto my machine.

[Hoards reading paper task.]
So I'll put reading paper in here. And, it too is red so... Chances are, the red in fact it's... Oh, I see what these little indicators are. OK. No, I guess I don't see what these indicators are. Um... Let me resize the window. Huh, so these 3 say 37– oh, these first 4 say 37 Meg. This last one says 34 Meg. Hoarded Tasks. Oh here's the cache space up here. That says 97 Meg. Huh. OK, I now have no clue what these little picture over here mean. Maybe I was shown in the tutorial and I've now forgotten. Um. And I certainly don't know why these are red. 269 268

[Hoards thesis task.] 268
Huh, it's red like the others are. So there's definitely something going on here, but I don't know what it is.

For Phase Three, we switch to a different screen format and a different method of interaction. The transition to this segment of the Exercises confused the user. The user's comments helped us improve this transition such that none of the six actual users expressed confusion over what was happening. 167
[Dialog between the user and the experimenter related to this confusion is omitted.]

[User fills out incident report after we fix the RVM problem.]
Oh, but wait.... Oh, I see, I see, I see, OK. So now the network's doing something goofy... So I wasn't supposed to fill out an incident report for that last thing.

[Incident Report button does not work while weak miss and discomiss query is visible.] 168
We fixed this problem for the final version.

[Reintegration Pending Event.] 239
Well, it's red, so I'll figure it's critical.

D.10.3 Debriefing

Pilot 3 was not debriefed on tape.

D.11 P4

D.11.1 Tutorial

Time	Transcript	ID
5:10	[Introduction7] Okay, that's a problem on a one-directional link, but not that much of a problem on a two-directional link.	
6:05	[Introduction9] That's good. Metaphors are good.	
6:35	**Can I ask you to be a little bit more [?] thinking aloud?** Oh. Ok. **Thanks.**	
6:50	[Introduction9] The metaphor may only go so far...[reading]...or like the battery indicator on a laptop	
7:45	[Introduction10] Hoard Walk?	231
8:05	[Introduction10] Though putting the Control Panel [indicator lights] on the side assumes that you're more interested in vertical space than horizontal, which isn't necessarily true, especially when I use the system, but that's okay. (eh?), live with it.	169
8:30	[Introduction10] Would it change to yellow on black, or would it change to black on a yellow background?	170
10:15	[UrgencyColors2] ...The default yellow is a lot brighter than green... so it's easy to see. Hmm... But to be honest I would like, I would like it if...critical were black letters on a red background.	171
	[UrgencyColors2: User agrees with the "For the record" comment.]	
12:25	So I was thinking warning might be nice if it was a little brighter. And...I'd really love to change the background color as well as the foreground color. Oh...Red is too dark, but orange is pretty good. Dark orange is brighter, which is good, but it's also less red, which isn't so good. Orange is worse still. I guess we'll do OrangeRed.	265 171 266
13:50	[Network1] Weak?	229
14:20	[Network1 – percent of maximum Ethernet bandwidth] Okay, that's interesting, except that bandwidth isn't used linearly on Ethernet so, this might be good, 17% on rossini might be good, but it might also be, kind of, marginal, and I just have to trust that the designers have selected thresholds that are reasonable, which is actually a problem if I upgrade to a 100 Mb, for instance, on part of it, possibly, ok, for instance.	172

15:25	[Network2] Ah. Ok. So now network is flashing. That's a good way to get attention. Very good.	
16:00	[Task1] I'm tempted to wonder why these [indicator lights] are not alphabetized? It means, when I, if I'm unfamiliar with it, and it says the Task window, I have to do a linear search through all of them to find it. That's a little slow.	143
17:15	[Task1] Where program in this sense means more than just the binary.	
20:30	Can I see multiple ones at once if I move this sucker over here? Yes, I can see multiples ones at once. OK. That's nice.	~24
21:30	[Explores the task definitions.] I'm thinking that "writing" might contain configuration setup for the programs here, but that hopefully is defined in the appropriate programs. So, for instance, editing would use the program emacs which would contain the user data .emacs in my home directory. [Explores the task definitions some more.]	173
23:35	[Task6: In Data Definition segment...] How do I get more of this? It would be nice if there was some easy way to see more of the text of the thing.	21
24:05	And since dissertation is a subdirectory of thesis, presumably you don't just suck in the whole subordinate tree, but you have to specify the leaves. OK... Meta-information. None... children or all descendants... OK. That makes some sense. That's not too bad.	
26:30	[Task7] "...he know is shallow tree...". Not so much shallow in terms of how many. Hmm... So shallow in this sense means not very much data, not, not very deep. 'cause it could be deep, but have not a lot of stuff in it.	{57}
28:05	[Task10] [Types in name of data set into pathname area.]	137
28:25	[Realizes mistake.]	
28:55	[Task11] I can see how typing /coda/usr/bovik would be really tedious in this, like tilde [~] would be a nice thing...	94
29:20	[Task11] Ok, and I've typed this, and it's not letting me select any of "None" or whatever, so this is slightly confusing.	
29:35	**So the, when you type a path here...** When I type a path here... **it actually checks to see** Right **if that's a directory.** OK, so this is, this is a–. **That directory doesn't exist** So, right. Right so some poor port of this doesn't exist, which is why I'm not allowed to do, do that. Hmm... **Exactly, so one of the things you could do–** Is make it up. There's an idea. Ok. It would be nice if there was more feedback here that... **this is not a directory** Yeah, that–	36

	OK like, some box that says doesn't exist or something.	
32:3?	[User hits "Save As" twice.]	37 184
33:00	Ah. In the data definition window and in file dialogs like that, on a MacIntosh these days, it's fairly common to find a "Make New Folder" for it, so in this case, "Make New Directory" button and that might be helpful here so that I don't– didn't have to bring up an, an xterm and just make the stuff by hand.	175
34:30	[Task14] Ok, how does it track?	246
36:00	[User hits "Save As"; interface inserts an empty entry widget; user re-clicks "Save As"; interface brings up "already exists" dialog; user mildly confused.]	37 184
37:45	[Task18] Hmm... Okay, "...and hit <Return>." I typed <Return>, and the only evidence that I have that it did anything at all is I might have heard the disk chatter. Hmm... Interesting. *User encountered an interface bug. The experimenter got the interface and the user back on track, so that the experiment could continue.*	
	[User discovers that <Backspace> doesn't work on `telos`.]	176
40:55	["Save As" dialog after the first "Save As" – this is an artifact of the way the experimenter worked around the interface bug, but since the user encountered this dialog box naturally at 32:3? above, the comments are included here.] "'Writing with Scribe' is already defined." Okay. That's confusing. Perhaps I wanted to click "Save" rather than "Save As". How 'bout that? Yeah. Okay.	184
	It has squawked at me about... [reads last paragraph of Task20]	
42:40	[Task21] And I guess there is no sense that it could be possibly be yellow – either it's there or it's not there. Okay.	
43:??	[Task21] Maybe it should be "1: writing thesis"? Stylistic.	177
43:??	[Task21] Oh, I can't just reorder the list?	64
47:00	[Advice3] It could also start servicing the, the re–, the request as it's asking you whether or not you want to do it, though that might interfere with other things you're doing, especially if you're trying to interactively use, say, a low bandwidth link – like say, a modem.	178
48:20	[Advice4 – weak miss time out series.] Neat. That was, that was pretty, uh, good.	
52:22	[Hoard Walk Advice Window.] "All Tasks Needing Data" – Not very informative.	71
52:48	[HoardWalk4] So is that like, "Fetch Now" "Don't Fetch Now"?	232
53:20	[HoardWalk4] It's becoming more common to have the, sort of, indicators that point ▼ or ▶ . [Note: User indicated the arrows with gestures.]	142

54:25	[HoardWalk5] It's interesting that you have to select it and then hit the button. Editing... and... Right. And then you have to select it and then double-click. So it's really three clicks.	179
55:00	[User hits "Fetch?" button] "Then click on the 'Fetch' button. Notice that a hoard walk– Indicator turns yellow." Mmm... Red actually. [Looks back and forth for about 20 seconds.] Okay. Let's see. Hoard Walk Information is Hoard Walk Advice. Or should I do that step again? **What step do you think you are on? [mumbles]** Let's see. Um. So I requested that all tasks be fetched. Then click on the "Fetch" button and it didn't seem to do anything, though this updated when I selected that. Oh! Aha! There's this and then there's that. **Right. I'll try to make that clearer.** Perhaps if you say the "Fetch" button in the lower right-hand corner.	237
	[Hits "Fetch?" button and believes fetch to have started when, in fact, this just selects the object for fetch, and the hoard walk won't actually continue until the user clicks on the "Fetch" button in the lower right corner.]	283
	[Space Help window – "Be careful not to hoard more data than will fit in your cache."] I wonder what happens if you do?	247
59:25	I suppose that's a good dialog. I only think it trite since I'm reading the tutorial.	180
59:45	[Tokens] I wonder when Tokens turn yellow, like how many hours.	248
1:02:45	OK. So, read disconnected, so I've got my, say, laptop. I am, say, not connected to any network and I go try to read something that's not there. And, it's configuration is to silently... now I don't know... presumably that, that does not mean silently fail, that presumably means silently hang... pending, pending network. Though I don't know.	38
	: [Reads description of "Operating Disconnected" event configuration. Looks at configuration of "Read Disconnected Cache Miss". User is somewhat confused.]	124
	:	
1:03:50	Is that true? I have notify "No." So it always notifies by flashing. That's not clear.	38
	: [Fixes configuration of the "Read Disconnected Cache Miss Advice" event to be consistent with the description of the "Operating Disconnected" event provided in the Tutorial.]	

D.11.2 Exercises

Time	Transcript	ID
	The numbers are good 'cause the bars are all about the same for the 15 and 17%.	
3:25	[Exercise #3] Task is red... So I pulled up information.. and what I see is... In the hoarded task... I see the cache space is OK... the task list... the list of tasks I mean... they're not... I haven't hoarded any of them yet. In the hoarded tasks, um, SOSP16 is only got 15% of it in the cache. (2) Recommendation letters has 2%, all of nothing in the cache... and (3) Darpa Grants has 64% in the cache. But editing is fine.	
	[Exercise #4] [Immediately pulls up Control Panel. Immediately chooses task indicator.]	

	Unfetch Help... Editing... Unselecting... duplicated in another task... unselecting this element will mean it is not fetched for other... But I need editing to do both of these things. OK. So this is a slightly confusing ... message because editing, which is, which I know to be a task, has ca– popped up even though it is not explicitly listed under task names. Do I want to unselect editing from all these tasks? No, I don't want to unselect. It's a double negative. It's a little odd, but there it is... OK.	140
	The current space status is OK. I don't even need to pull up the box for, but I did since I'm predicting from the dialog that I need to answer more questions. :	
	These are good dialogs. They're clear. Well, they're almost clear. This 66 is overlapping a little to the left, so stylistically it could be off on that side. Um... But these are pretty good.	77
17:30	So I want the events. Um... So I want the configuration of the events so I'm pulling up the Control Panel. Event configuration for... Let's see... The cache misses should be under space, I hope. No. No, that's just running out of space. OK. So I look under Network, and I only see operating strongly, weakly or disconnected. Ah... Advice. OK. Hmm... It's slightly odd to me that the we-, the we-, the cache misses would be under Advice rather than Space or Network since I consider the cache to be part of Space. Fine.	17
21:20	Actually, I'm not sure, thinking about it, I'm not sure. I'm not sure if the weakly connected means my link or the current server link.	233
22:30	...First, I'm going to try to disambiguate read disconnected from a weakly connected.	80
22:55	Further investigation shows that "weak" refers to my link.	233
23:00	OK. Disconnected cache miss. Now, not clear, it's not clear to me... what's the difference is between a disconnected and a read disconnected, though I believe this is a write disconnected. What I tried to update. Not... I imagine... since this is... different from a read disconnected miss that this refers to a.. an update, update to, to, to change data on a server and... [inaudible, but his typewritten answer would complete this sentence with "cannot reach it (or anything else)".]	80
29:20	I'm going to look in my...xterm here to see whether I can pick just immediate children or all descendants of /coda/usr/bovik/src/venus... So I'm changing directory to /coda/usr/bovik/src and I'm going to look in venus. And, not see any directories. I'll look under advice. OK. And I'm going to look in console. OK. These all appear to be, ah... OK. Console uses tcl. Hmm... [clears throat] Make a note to myself that console uses tcl, which means that I'm going to need to pick up tcl. So, good, fine. Ah... Well, it's safe to do all descendants on this. Good.	{57} {185}
	I'm not quite sure... what... that underline means. It's just sorta, there. I'm not sure if that means anything in particular or not.	181
	Hmm... I wonder why CodaHacking is unde... before New... I guess it's an alphabetizing thing. It would be convenient for New... to be at the top. So I'm going to turn that to lower case... and save it again. Good. Oh. How interesting. Fine. Go away. You, yo, alright, well. I tried to delete this and it didn't seem to do anything. Ah. Delete task. Aha. That makes some sense. OK. And now I'm conveniently at New... task so that's good.	7 138
	And that's going to require CodaHacking. Oops! Just once, thanks.	22
	The bugs are compressed. So... I'm going to need ga-zip or something. But I'll do that in a moment.	{185}
	[Experimenter explains @sys (i386_mach vs. i486_nbsd1) issue.]	
	Pretend any binary is in /coda/misc/bin.	

	Well, I'm just going to bring up... the CodaBugFixing task definition 'cause it's the most convenient way to get to the user data definition... this is so that I can make a new one of them.	183
	Oh... I hit <Tab> and it doesn't send me to the bar so I'm wondering what I've... ha ha... I have made a typo.	36
	So, I've defined user data that's "useful headers" and now I'm defining a task that's "looking at headers," that contains the "useful headers" as data and gnu-emacs as a program and I don't need any subtasks. This will let me put the "looking at headers" task in the hoarding area, but at a different priority than "CodaBugFixing." OK.	
	[Save As on Scribe twice] Set named Scribe. Oh... So I didn't have to do one. Well, good. So there already was a Scribe. Oh really? ... Well, I'll just look at it. And it's just fine. But there isn't a gv. [Defines gv] OK. Boy. I just hit them. So I move this over here and I'm going to look... ah... there's it. Good. Alright. Fine. [Finds 2 gv's now?]	184
	[Removes "gnu-emacs" program from all task definitions and adds "editing" subtask instead. Likes it better this way because it's more abstract.]	
	Fine. It started yelling at me about how it doesn't have everything to "write cv", which is not surprising.	
	Ah... Tasks was flashing because it was turning green. Good. So the previous stuff should be all set.	{9}
	Poof! And a magic fairy turns RVM to okay.	
	I seem to actually be using occasional cognitive cycles, glancing over at the network display, er, the, ah, the, ah, summary panel actually to make sure that everything is still ducky. Tetris is a good test of this because it tends to require actually a lot of thought. So... Huh! What a score! ... Oops, missed a block.	41
	Now I'm not sure I can decide whether tokens expiring is a warning problem or a critical problem, though frankly, I would classify this as critical since, if I don't type in tokens I don't... I'm not able to do anything more.	257

D.11.3 Debriefing

Time	Transcript	ID
	["The interface helped deal with the actual complexity"] Right. So, suppose I didn't have this interface and I was trying to set up to hoard. Um. You know. I've still got this problem of I have a bunch of data that I want to work with, a bunch of programs that I want to work with and I need some way of specifying, well, shove all these things in the cache. You know, go get them now. And I can easily see that without, without some decent tool to help package them up that could be, um, become a mighty big pain. Um. I actually foresee– so for common ones just writing little shell scripts that did essentially the same thing as this, but this is a nice, general, easy-to-use interface. I really. I thought that was actually one of the, one of the best, the best sort of, interesting parts of the, hard parts of the interface is to be able ta–, to abstract away from individual programs and data to tasks and tasks that use general pieces of data as their thing.	
	[Concerned about tracking references – discussion.]	

	One of the problems with the abstraction of tasks is that, you know, if you buy into it and it looks like a really good idea to me, you know, to do that and you want to set up a nice abstract set of classes. Then that's a great except... there's still things you have to discover for anything, like, oh, I want to work on fixing Coda Bugs during the snow storm. Oh, I've gone and compressed all of my bug files so now I have to go...	185
	[Likes gauges and lights: lights to get attention and gauge to see more detail. : Tri-state: OK– warning– error better than good/bad.]	186
	[Discuss needs of Speech Recognition systems.]	
	The ability to specify pop-up dialogs for most any situation arbitrarily is great. Um... [having the system able to pop-up and user able to configure it to pop-up or not] 'cause pop-ups are annoying and I want to be annoyed in some situations and I want to be left alone in others.	
	The point of that is that the situation changes, sometimes. So, it's not like you can figure out should this pop-up or not once and for all and get it right. Um. So the flip side of that is that, I think what I'm suggesting for an expansion of it is to sorta have working sets of settings. Um, so that you can have these macro descriptions of "I'm disconnected," or "I'm at CMU," or that sort of thing. So where when you think you're plugged in and it loses the network you'd like to know about it right away. But if you're disconnected, you know, who cares and selecting amongst those would change a whole lot of individual configuration settings. : [Seen something like that for Power Macs for "plugged in" vs. "on battery".]	187
	[User offers a hypothesis as to why users hit "Save As" twice. Users may expect "Save As" to bring up a dialog asking for a new name.]	188
	[Discuss why user didn't fill out form when state changed from bad to good.]	

D.12 P5

D.12.1 Tutorial

Time	Transcript	ID
	Well, I clicked "Commit" and nothing happened. I'll click "Commit" again. OK. So that's kinda funny. Ah. Oh. OK. So have to close it separately from committing.	32
	(There should) be something so all the windows didn't come on top of each other and obscure the earlier ones.	1
	If I keep going like this, I'm going to have too many windows to handle.	24

Whoa! Why won't it let me click "all descendants"? That's really funny... Oh, I guess I have to hit <Enter> and now I can click "all descendants".	20 3
So, for executables, it would be nice if could follow the path. I wonder if it does that– 'cause I don't remember where these programs are.	94
[close all windows] Close. Oh I forgot to click the "Save" button. Can I pull that window up again. Task Definition. Task Definition. I guess I lost it. OK. So, I need to... OK. Duh! Oh well. That's Okay. : [Redefining a new task definition.] Click the "Save" button– don't forget to do that this time.	127
Click on the "Scribe Tools" in the program predefined list. I did that. OK. Well, it di–, it didn't put "Scribe tools" in here. Oh. OK. So, it underlined it in some ways. So, hit the save button– Oh! I double-clicked. I should have clicked. This double-click/ click thing is such a pain. OK. "Scribe Tools". Excellent.	23
So it doesn't tell me how much is unavailable. Does it? Oh. OK. 90% is unavailable. [Actually, the meter is indicating that 90% is available.] I guess. That's sort of confusing.	268
I guess it was green before.. so that means that a hoard walk isn't happening I suppose. OK. Let's... let's close this. And I suppose it'll do it in the background, except that it's doing this funny flashing thing now. Which I'm not quite sure what this means. Now they both went red. OK. When weakly connected to the servers [mumbling] may request advice. By default the Hoard Walk indicator changes from yellow to red. Hmm. I guess now the different colors just mean too many different things. Maybe it should pop-up windows instead.	9 262
Please select the "Read Disconnected Cache Miss"... under the Advice... And, read disconnected, oh I guess that's "Write Disconnected Cache Miss". Oops!	80

D.12.2 Exercises

Time	Transcript	ID
	"What is our connectivity to each of the servers? Strongly connected, weakly connected, or disconnected?"	
	:	
	OK. Let's pretend we're weakly connected. And, uh, I don't know, and scarlatti 'cause that always sucks.	274
	:	
	"Please describe the "One or More Unavailable Ta–" event. Does the interface... notify... what urgency level? Oh, oh, OK. Oh. Whoops! I thought I was supposed to make stuff up. Oops! OK, so I clearly. OK so, let's, let's, ah. [Sighs.] One or more tasks unavailable. So where would that be? That would be under... task... [User double-clicks on Tasks indicator, looks around, then double-clicks on Control Panel.]	147
	:	
	Right. OK. Tokens, Network, Advice, Reintegration, Repair. Wait. So where did this? Space maybe. Network, Advice, Reintegration, Repair. Task. Ah. There it is.	17
	Lower numbers is higher priority, right? I think that's what it means. Let's click on help and make sure... That's right.	82

OK. It doesn't seem to be doing anything...yet. I suppose I have to... I don't see the hoard daemon asking me for advice. Maybe I should ask for a hoard walk, so let's go click on hoard walk. OK. Selected two tasks. Is that right? Well, it looks like we're at three tasks, with Darpa Grant. So, fetch, no, fetch, fetch, OK... Hmm... That's really strange. The tutorial seems to say that the hoard walk was started automatically. But, ah, wel– whatever... Well, it walked the whole hoard. .... OK. 155 235

[Searches through indicators for the "Weakly Connected Cache Miss Advice" event.] So maybe there should be a description of each indicator since there're so many of them. 17

Whoa! I keep flipping yes and no. Damn! Alright so I'm... I need to get yes and no straight in my head. {161} {70}

Wait. Where's my "Building Coda"? Don't see it. Oops. OK. [searching] Event configuration. Whatever, whatever. Pair. OK. So this is not it. How the hell do I make a new task? Tokens, Space, Network, Advice, Hoard Walk... Oh, OK, here we go. And, oh, "Building Coda" is a program. So, how do I get a list of all the programs task I've defined without going through, like, something that uses one of them? That's very weird. 18 183

Oh! Whoa! Why won't it let me click on all descendants? Oh! Come on! Hmm! That's interesting. That's really weird. So. Won't let me do that. Hmm. Ah OK. I typed C-I-D-A that makes sense. Oh. It still won't let me do that. That's kinda weird. That is weird. Oh well. We'll just assume it let me did that. OK. 20 36

Hmm. It would be nice if it, like, beeped or something to tell me a directory didn't exist. 36

OK. So I'll make a low priority task called "Coda includes" or "Coda unimportant includes".

Where's Scribe? Ah, in, aw... don't have Scribe, uh? Well. Eee, no don't click on that, don't click on that, don't... Double-click, huh! Well my Scribe tools just went away . [Thought this was defined from before during the tutorial?] {23}

Don't forget to think out loud.

[Hoarding stuff] 127
And, then, ok, it looks like I did didn't save, oh man! I didn't save that "Coda bug fix" task. Oh! What a pain! OK. So let's go make that again. Ah... Oh, bummer! OK. So. It needs to prevent me, from ah, from closing a window if I don't save something because I just do that <u>way</u> too often. And, now, my right, my "coda bug fix" task has just disappeared. Oops!

Please remember to fill out incident reports even when things turn from bad to good.
Oh, OK. So one like one for when the RVM thing went away.
Yeah.
OK

[TaskUnavailable event causes system to beep three times.] 29
Task. Oh, it beeped. Right. What happened? Right. So, what's wrong? So, it couldn't get anything on the CV. It beeped. Not exactly sure what it wants. Cache space is fine. Hoarded tasks– red. So why did the CV thing go red? I guess that's critical.
:
I don't quite understand this. Why would it go red? Well, let's look at space. Space is fine. What about hoard walk? Well, what if I tell it to walk? Okay. Okay, so it's doing something funny with the hoard walk. So why does it need the CV fer– huh? OK. So, so, I'm disconnected [User is not disconnected!] so that's why it's doing

funny things with the hoard walk, but why did the task thing go red? Because, it couldn't get some file? Don't know. That's really strange... Okay, so it went green. OK. And it filled up my CV thing. So this.. I... don't quite get this because I was compiling my thesis I thought.

OK. So, what happened to the network now. Oh, I need to do an incident report. So I've become weakly connected.

[TokenExpiry event causes system to beep three times.]
OK. What happened? I ran out of tokens. Did I? Oh no... So, submit. No password provided. Whoops! OK. So this is critical, because I need all this stuff.

Oh, I'm disconnected! No, (I'm not) disconnected anymore.

Why is it (changing) OBJS when I was doing a latex?

D.12.3 Debriefing

Time	Transcript	ID
	One thing I had trouble with was, sort of, the general instructions were all mixed up with the "do this thing now." Or seemed to be. So, if they were sort of separate and bold font or something, it might help. **OK. So, so you mean the arrow things.** Oh! Those were the, ah, OK, so I guess, right. Um. [?] Maybe it was just like, my learning how to use the whole thing because that sort of thing went away after awhile.	189
	I thought it was funny that the tutorial didn't have any of the reintegration stuff and then it popped up during the exercises. **So the reason I did that, is just to see how people would react and whether they'd be able to figure it out.** So that was intentional. **That's why it was the very last thing and that way if people got totally confused it didn't hurt anything else... The disconnected cache miss was the same way. You hadn't seen that in the tutorial either. You'd heard about it, but I hadn't actually showed you what it looked like.** Right. **That didn't bother you as much.** No, I guess, ah... So, the funny thing was you would expect for a disconnected cache miss that, like, your latex would be running somewhere and it would just stop and keel over, but that part of it wasn't in there at all...	239
	The other thing with the tasks is that, I think, programs, there should be a way to put in stuff that Coda doesn't infer as well 'cause with latex, like, if you, if, like, it if you just cache the stuff that latex used even on 10 runs, on the 11th runs it might decide it needs a new font file. I mean, 'cause this happened to me all the time. So, I mean, I should be able to say, for this program cache all the **this whole directory** or these set of font files. Something like that. **So you'd like the programs to also let you include, and I know you need this, like, so definitions kind of like the, in the data definitions only part of the program definition.**	190
	[(2e) Clarify: Little bar that shows how much of my cache I'm using up. If a hoard walk is going on, show meter on screen. Or click and have them come up without all	

	the motif stuff surrounding it.]	
	[Cars that talk to you. Something's wrong with the headlamp and then you go investigate to figure out what.]	
	[Evaluation Questionnaire: User most likes the task support.] **You like the task definitions, huh?** Yeah, that, I mean, that's the most painful part. Is constructing those hoard files so if there was some easy way to do that, that's what I'd use the most.	
	[Evaluation Questionnaire: User most dislikes number of indicator lights.] **Too many indicator lights– which ones do you think could be combined?** That's a good question... [User suggests having one indicator just that means "something is wrong".] [User suggests having the green ones not visible, to only have the indicator show up if something goes wrong.] [User expects to be able to configure which indicators are visible "in a final system."]	192 193
	[Discuss problem with "Yes"/"No".]	
	Does anything in the study throw you because of what you know about Coda coming in? [Confusion because no mention of spy; discuss "autospy" tool.] *Should be fine for experts.*	

D.13 S1

Participant S1 is a member of the secretarial support staff. She volunteered to participant in my usability test to better understand Coda. Her background includes some computer science undergraduate courses. Once again, since the goal of our work was to familiarize her with Coda, I tended to interact with this user somewhat more than would normally be appropriate. For example, about 28 minutes into the Tutorial as she was struggling with a known problem with the directions in Task10, I stopped her to better explain what was meant.

D.13.1 Tutorial

Time	Transcript	ID
	Hmm. **Don't forget to think out loud.** Hmm? **Don't forget to think out loud.** OK **Thanks.**	

	[User explores Task Definitions.]	
19:??	[User does not double-click on "thesis" element, as instructed, but she is able to recover on her own.]	124
25:??	[Reading <Tab> vs. <Enter> description.]	3
	And if I go back [to first path: /coda/project/coda/slides], it does not allow me to do that. Probably because that thing really isn't a directory. No. Because– Hmm. [Rereads description.] So, another pathname. How 'bout /coda/project/coda/(proven).ppt. [sigh]	36
28:??	**It is checking those pathnames to check to see if they're directories. So if they– if the directory's not– if it's not a directory, the meta-information will not be enabled. Make sense?**	
	Mm. Hmm.	
	OK.	
	Wait! If it's not a directory, meta-information won't be disabled?	
	Right. Will not be enabled.	
	Will not be enabled.	
	Right so.	
	If it's a file though, such as	
	If it is a directory, it will enable the meta-information.	
	OK.	
	OK.	
	So, this is a directory. Why can't I choose (the meta-information). [Mumbles] [sound of typing].	
	You sure it's not /coda/usr/satya?	
	It is. But I thought we were supposed to make them up.	
	Well, you can. Use actual direct- So, so, This. It's actually looking in the file system to verify that that's a directory.	
	Mmm. Ok.	
	Does that make sense? So, you can– you don't have to make up– fake– you don't have to make up fake filenames and pathname. It was just use whatever filenames– pathnames come to mind is what I meant by that...	
34:15	[User types "/coda/misc/bin/scribe and /coda/misc/bin/gv" as a program pathname.]	137
55:??	Oops. Back one. A window titled "Authentication Information" should now be visible on your screen. I skipped one.	194
	Let me read it to you.	
	OK.	
	So you have a window title. OK. Now let's explore. Let's now explore the Tokens indicator. For the purposes of this study... [Experimenter reads Tokens1 to the user.]	
	Thank you.	
	You're welcome.	
57:30	[User struggles with the first paragraph of EventConfiguration2 (selecting a different indicator).]	226 {206}

D.13.2 Exercises

Time	Transcript	ID
	Hmm. Perhaps I'm not allowed to have that open at the same time. : **Click "OK" on the Help window and things'll come back.**	13
6:??	Hmm. Critical. That's the term I'm looking for– critical. Saying critical. Nno. Well, let's see. Will it beep at the user? Ah. If we have Control Panels, then we have– well, wait. I have Task up. So, Darpa Grant. Nmm... No. That's not the information we're looking for. Or, that I am looking for. I am looking for inf-information about that task. Annn... not Space, and it's not Tokens, and in Control Panel. Hmm... How 'bout Task? Ah. "Under what conditions will the "One or More Task Unavailable" event occur?" Three, Darpa Grant– that's an example. <u>But</u>, it <u>really</u> is condition, task, um, is a multiple task and that's why the red-critical is shown. I believe.	234 147 17
	[When do Tokens expire?] April 24. At 10:25, I believe in the "am", I'll assume it would be military time...	195
9:??	[What Tasks are currently defined?] There are four. Hmm. Defined. Darpa Grant, Editing, Recommendation Letters, SOSP16, and Writing. 1 2 3 Wait 1, 2, 3, 4, 5. Which of these are hoarded? 4.	
11:??	[User does not explore beyond "All Tasks Needing Data" in Hoard Walk Advice.] So. Recommendation Letters and SOSP16. All Tasks. So really what I want them to do is to finish the hoard walk and to go ahead and <u>fetch</u> the information. So these tasks now separately are both at 100% availability. The hoard walk is in progress.	236
14:30	[Please describe Miss Events] No, I don't want Network Information such as that. Nnn. Nor. Nor this I believe. Operating Weakly Connected. Nnn. No. Operating Weakly Connected. (...urgency colors). Really, this has <u>nothing</u> to do with it. Hmm... The Advice Monitor isn't going to tell me that either. Task Definitions aren't either. So basically just describe	17
17:00	Operating Weakly Connected. The advice. What will the Advice monitor provide. Well, weakly connected, will, will notify the user with a yellow warning, <u>not</u> critical, but will not pop-up, oop, beep, or flash. The other question is "<u>Dis</u>connected Cache Miss Advice" will notify the user with, I should really include that, cache miss advice [added to typewritten answer about the "Weakly Connected Cache Miss Advice" event], will not, so it will notify the user with a red-critical warning, but [mumble] will pop-up but will not beep or flash. [This is the correct configuration for the "Operating Disconnected" event.] *User modifies the settings for the "Operating Disconnected" and "Operating Weakly Connected" events. The experimenter then spent quite a bit of timing figuring out how to undo these modifications before the beginning of Phase Three. (We had anticipated that the user might incorrectly modify the actual events and had written the Exercises to ensure that these events were configured properly before the beginning of Phase Three. We had not, however, anticipated that the user might modify other events. In any case, the experimenter successfully fixed the damage before Phase Three began. The user apparently remained unaware of the goings on, though could have noticed a mysterious window appear and then disappear very quickly – if she did, she made no comment about it.)*	

["Building Coda"]
Is that defined somewhere? Or is that something I need to define. Hmm. What is predefined? Not letters of recommendation. Not papers. Not proposals. Darpa 97. No. "Create a task for fixing Coda bugs." What I need to do is to contain these executables. OK, that makes sense. "Building Coda". (Well) I also need editing because it's going to be reports. So, writing. Probably gnu-emacs since it's going to be writing and debugging. Only what application or what predefined debug coda. gnu-comp/@sys/bin/gcc. That is not. Source code. Well. How do I– the sources to the code that you must modify are located– Hmm. Nope. Contains the following pathnames. No. So, I need to debug coda and these are the sources to the code. So. /coda/usr/bovik/src/venus. Hmm. And, of it, I think that I should include "all descendants" because I need all this information. I need it to pull in from there. And again /coda/usr/bovik/advice and again "all descendants."

I believe that's not an executable– I believe that's a file. No, that is programs, so I also need to find it. Here. Backspace. Control-U doesn't work. Oh no. Oh no. 196
/coda/misc/gnu-comp/omega/bin/gdb. [Note: User added gdb to the gnu-emacs definition rather than the "building coda" definition.]

[User looks for bug reports. It seems that user thinks the reports are formatted so user includes the "writing" task in the "bug fixing" task.] 197

33:30 [User adds header files to "bug fixing" task– in data definition with the source directories.]
"So you don't want to just add them to your 'Building Coda' programs." So you don't want to add them to programs, but you could add them to user data. "Because it could be hoarded in preference to really important source files". Hmm. So why don't we just add them to... user data. Um. Perhaps we'll just add for immediate children and add them as user data. 100

38:45 [Puts a single program in each definition.] 198

39:45 [Single click adds unintended element.] 23

41:30 [Please hoard these tasks. First tries Hoard Walk indicator.]
Now I need to hoard them. OK. Debug coda. Whoops. That's not what I meant to do. No I wish to bring it here. I have 66% of cache space available. Various header files. So basically writing...

43:45 I think debugging should be first because that calls for the most resources so... walk now for the Hoard Walk. 199
[See also: user's written comments re: reasoning]
[Verifies resources requirements in Task Information. Requested walk before instructed to do so.]

I can't [bring up "tetris&"].
You can't put the, um, the ampersand doesn't work because it takes over the xterm.
OK. And I have another question for you– what is supposed to be happening, though, while I'm doing this?
I can't answer that. But now that you've started playing the game, you can click "OK".
OK.

55:05 [User does own hoard walk to solve TaskUnavailable problem.]

57:45	Oop! [User and Experimenter chuckle.] Sorry **Um. You killed ctwm– [laughter].** Sorry. **That's okay. It didn't– I don't think it hurt.** My hoard walk is still going. **I wish I could see that. [mumble]. Hello?! OK. All better except that things are appearing in the wrong– you're just going to have to ignore some screens–** That's on pause. **I need that. Raise window. Ignore this stuff down here.** OK. That's fine.	
1:00:40	Oh. Network. Pause. Wow! All resources are low. Requesting program: /mach/alpha/bin/latex2e. Estimated Fetch Time: 23 minutes, (?)7 seconds. So, we're weakly connected with our miss cache. Advice is critical. Weak Miss. "Weak Miss questionnaire running already. Please try again later." "There is a weak miss questionnaire running already. Please try again later." Sign... Um. No, please fetch.	200
1:01:40	"The bandwidth to each server is expressed as a percentage of the maximum Ethernet speed. Servers whose names are written in light gray have had their network connections artificially manipulated through the Control Panel. Servers whose bandwidth estimate is unknown have a bar shown in dark gray." OK (I) understand that, but I'm just weakly connected. Advice monitor is not helping me right now. I'm sort of in fetch mode right now. Hmm... And what it's indicating for weakly connected is a warning. Notify. Yes. Uh. OK. Now the network is really bad. Um. It's telling me I'm disconnected. So. [Sigh.] I need to connect to all (of 'em). These aren't things I usually have, ah, control over. So the incident report is this...	
1:05:00	Now. A disconnected cache miss has occurred on the object. /bovik/thesis/dissertation/intro.tex. The object was referenced by– How much do you believe this cache miss will affect your current work? Invalid. Not at all. This is a practice disconnection. I am not currently working on my thesis. Don't even have a B.S. degree. This will not affect me all. "Please choose a value from 1 to 6."	149
1:05:??	OK. The network is back up. Wonderful. Tokens. Tokens have expired. Um. I would "submit", but I don't know what the password is. I am not familiar with reintegration.	239

D.13.3 Debriefing

The user was debriefed on tape, but nothing is audible on the tape.

D.14 T1

As Participant T1 began our experiment as Participant N3. However, as this user began reading the Tutorial's introductory screens, he realized that he actually knew more about Coda than he thought – as evidenced by the comments at Tutorial times 2:06, 3:10, and 20:30. For this reason, we did not include him as a novice user.

D.14.1 Tutorial

Time	Transcript	ID
	[Pink ≅ the walls] The walls? Like this red color?	285
	[Skips second half of paragraph 2 of Introduction4]	124
2:06	"A consequence of this is that we can voluntarily disconnect..." I know that.	
3:10	[Use hoarding to combat this problem.] Oh, by the way, I know about, a little about hoarding too. I didn't put it down in the questionnaire.	
	[Appears to have skipped 4th sentence of paragraph 2 in Introduction6.]	124
	[Questions whether it's really 4 orders of magnitude.]	
	[Appears to have skipped 2nd and 3rd sentences of paragraph 2 of Introduction7.]	124
6:20	[Dashboard indicators– example of having to pull over to side of road five minutes ago or risk serious engine damage.] Hmm. I don't understand that.	286
	[User has some difficulty identifying the indicator lights window.]	111
7:00	I wish the color's brighter I guess. The green's kinda, um, low– ah, not, dim.	264
7:21	[...the green means everything is normal.] Maybe bright green's better. Um. If something were busy. Well, I guess it would catch his attention too much when you're doing work so dim green's good I guess.	264
8:15	[Note that there is a Help button on every screen in the interface.] Uh, where's the Help button? [Looking at indicator lights]	45
8:30	Ah! This is annoying. I'm just moving the, ah, window [tutorial] to the middle so it's not so low.	1
9:40	Sigh... Why does it always go to (that) position? This is annoying.	1
	[Skips much of 1st three paragraphs of UrgencyColors2.]	124
11:00	[User is not sure what the "For the record" comment in UrgencyColors2 means.] So, I wish these things are in color, so maybe I'll click on the– the or– the critical and then I'll choose the color from these buttons would be a lot easier for me to use.	287 91

	[On Network1] I wonder what "weak" means.	229
13:05	Oh! It's flashing! Network goes down automatically? But, how do I know which files in which server? This is strange. I wish somebody would tell me that.	129
	[Please close.] What if I don't close this? See what happens. It closes itself! Yeah. Huh. That's cheating. I didn't close it myself.	
	[Skims sentence 2 of Paragraph 1 of Task1.]	124
14:10	Yeah, but latex is a hard one because you have to get all the fonts and styles. This is not clear this is an easy thing to do. Um. Even emacs needs a lot of lisp files.	
	Oh. This didn't go away. I'm sorry. This is weird.	121
	[User skips Paragraph 2 of Task2.]	124
16:40	[User would like to highlight things in tutorial to mark spot.]	130
17:30	[Task4 is very confusing to user.]	288
18:50	[User misses scrollbar on Data Definition window.]	131
19:40	[User attempts to resize window– ugly.] Why is there all this empty spaces here? Why doesn't the space get used here. This is kind of strange.	4
20:30	[User skips meta-information description Paragraphs 2 and 3, in Task7.] I know that; I'll skip.	124
21:00	If you resize sideways, I wonder if this shows the whole thing. It still doesn't show the whole thing. That's weird. ... slight annoyance... [User wants to see full pathname in data definition, also wants the interface to either resize properly or forbid resizing completely.]	4 21
	[Difficulty finding black down arrow.]	34
	Hit <Enter> to move the cursor down. I wonder if I can just click. Yeah. It works. Cool.	
	[Type in pathnames, feel free to make some up.] Sigh. It's being– not very creative.	
24:10	[Trying out <Tab> in Data Definition.] So, can you do <Back-Tab>? <Back-Tab> doesn't work, but <Tab> works.	132
24:50	Oh, by the way, if I'm typing a path like this random thing, maybe this I want this to be a path but how do I know what's in the file system? I guess. Only if you're a Unix user, you know there are certain directory that has to be there in order to test out whether something's a directory.	94
	[Iconifies Task Information window.]	
28:00	[Program definition– click on black down arrow button.] That's good. This is consistent.	
	X cut and paste is not working. That's annoying.	133
29:00	[Use ghostview rather than Scribe?!?]	
29:10	[Please close all windows titled XXX.] Why does it keep on saying "close all windows" instead of "close the window"? There's only one of them, right?	134

30:15	[Task16 instructs user to add "with latex" after "writing" and save.] Type "with latex". So this is changing. Ah! That makes sense. Click the Save button. OK. OK. Eh, it changed! Oh! It created a new one. This is bizarre.	5
30:35	I don't see a difference between these two, but the window seems to move. That's weird.	135
31:30	[Fix the name and hit <Enter>.] I hit <Enter>, but it doesn't seem to do anything.	136
31:55	[Click on name and hit <Backspace> or <Delete>.] That's good. That's easy.	
32:30	[Realizes mistake in Program Definition.] This is weird. I have a path here, rather than these funny names. I must have done something wrong.	137
32:?? 33:15	[User tries to delete program definition by clicking on program name in "predefined" list and hitting <Delete>.]	138
33:30	Where the heck is Scribe? Writing with Scribe. I'm very confused. Uh.. Scribe Tools... will start in the program predefined list– program predefined list. Did I kill Scribe by accident? I don't see Scribe in here... Maria, I can't find Scribe Tools. Did I delete it somehow? **There is a known bug. It is possible you deleted it without realizing it. So why don't you go ahead and redefine it?** How do I– oh– So how do I do that? I don't remember. I go here. I go "New..." And I just say Scribe Tool. But I don't know the path. **You may assume everything is in /coda/misc/bin.** OK... And is it just called Scribe? **Yes.** Great. Thank you. **Ah-huh. ... Sorry about that.** Nothing ... sorry about that. I must have done something– weird. **No, no, no, no. There is a <u>known</u> bug.** Oh. OK. **That didn't crop up until after we started the study. So I'm not allowed to fix it.**	139
	[User gets hit with double-select bug?] Oh, it just showed two of them.	22
35:20	I'm supposed to be able to delete it like this, but why is it not deleting? That's really strange. Um. I expected to– so I don't know why I can't delete it. I mean. I wish I could click and delete– click.	138
35:50	[Save] I wish there's a indication that actually the Save is successfully done somewhere. For example, in emacs, there's a button that shows something is modified. So that would be a good thing to have, because I kept on clicking with Save and it just depressed and came out. And God knows if it actually did anything. OK. Just something that would help.	59
	[Should now be looking at Task Information.] **You iconified that window.** I did? **Yeah.** I didn't kill it. Thank you. **Yup.** :	208
36:55	Yeah, but if I click on it, it should deiconify or start a new one. Anyhow...	

37:40	[Use a bizarre form of drag-n-drop.] What's a bizarre form– ? : Oh, you don't hold the mouse– you click and let go. So, this is not really click and drag, right? Or drag and drop so you shouldn't say that. It's confusing.	289
38:10	[TaskUnavailable event causes system to beep three times.] Eek. It's having problems. Bad connection. No good. How do I repeat? What do I do? What do I do? [Reads last paragraph] ... Yeah, but how do I do this with hacking device? Eh– I'll do it anyway.	261
38:45	[Meter showing availability of Task(s).] But, what good is it to get 80 or 90 percent? I mean, it's all or nothing, right?	268 114
	[Tries reprioritizing.] This is weird. It doesn't seem to work completely. [User seemed to be reprioritizing into same priority position. I think it may be confusion as to where you click to put the task in a specific priority.]	42
42:50	[Advice blinking red.] It's not blinking anymore. Wait a min– it will change to red. So it only blinks a little bit I guess.	33
	[User answers weak miss questionnaire. Then, reads instructions telling him to do so.] What questionnaire? Oops. Clicked too quickly. Missed something.	
44:??	**You just did exactly what the directions told you to do.** Yeah, but I missed the questionnaire part. What questionnaire? **What you saw <u>was</u> the questionnaire.** Oh! Oh! That's just a question. OK. I thought a question– that's my English understanding– I thought a questionnaire was more like a big form with lots of questions. **Nah– just a little one.**	230
	[Sighs.] This is a lot of reading.	
45:40	I wonder what's the time out for the second question. [of weak miss series].	
	Hey! It's blinking again. It's gone again. Cool.	33
	[Skips all paragraph prose on HoardWalk1.]	124
47:55	It's actually hoarding?! The disk is not spinning. But anyway.	
48:15	Why? [...does the hoard walk indicator change to yellow?]	258
48:50	Everything's blinking now. Why is it blinking red color and then green and yellow? Shouldn't it be staying at red? Oh, it's changing the color to red.	9
51:35	Why is this a toggle? [Fetch and Stop Asking]	141
52:20	["+" and "-" on tree widget.] I guess the convention for these signs is a downward triangle. I mean this in windows that's what it looks like so maybe it's nice to use the same conventions. But "+" and "-" are also very intuitive. But if you put a triangle here, I would need instructions. I	142
52:50	would know I can click on it. I no– wouldn't know– I probably would try clicking on it anyway. For example, I can click on these guys too– That's very nice! Ah. This is cool. What if I do this? Ah. This is nice. So if I expand it again, does it remember? Yeah. It remembers. This is a very nice interface. Cool.	

	[Skips all paragraph prose on HoardWalk5.]	124
53:20	[User is presented with unselect question.] That's silly. You somehow select these two and it asks this stupid question.	140
	I'm not understanding this very well... This is annoying. This is a <u>single</u> file. Why is it asking me those questions? I mean I'll understand if it's, like, under a subdirectory, like this, and the questions should probably, should be around here somewhere so I don't have to move my pointer all the time. This is a funny place to place the questions.	140 1
	Task is blinking red again.	
55:??	Wow! This 70% is not in the box. Bad. Bad. Bad. Bad. Bad.	77
56:??	[RVM full Help window.] That's useless. Well, I guess it's useful– at least I know who to talk to, but I can't fix it myself. That's not good.	76
56:45	[Cache filling Help window] [User skips top part] Be careful not to hoard more data -blah, blah, blah than will fit in the cache. I can make my cache bigger. How do I make my cache bigger? You're not telling me that.	76
57:15	Why is my network still yellow? I don't understand. These guys are not yellow. Well, I guess, when it's 1% if it's yellow, I would like to see the fonts in yellow. I don't understand this. This is strange. They're all black.	8
	Why not list this [indicator lights] in order?	143
58:40	I would rather use klist.	144
1:01:40	So maybe we need a system default just in case I screw up so many times and then I'd like the original.	145
	[User is looking at Advice Information window. Asks for help.] **Right. You can close this window. Try reading the help.** This Control Panel allow you to configure events notification. The left half of the screen allow you to select the event you would like to configure... **Oh, this didn't get updated. Go ahead and click OK. Let me rewrite a new help window. Suppose the help window said. Um... In the upper left corner of the screen, you can select the indicator under which the event is notified... Then, in the lower left corner, you can select the event that is indicated.**	251
	Sigh... I'm not very good at instructions. [Not sure what he's referring to here... something about "commit".]	
	The user apparently changed the configuration of the "Read Disconnected Cache Miss Advice" event while exploring on his own. It no longer showed that it was disabled. User was confused. Experimenter restored it to the "default" configuration.	
	[User does not understand "Under the 'Network'" indicator. Figures it out.]	290
	I wonder why all the windows open to the corner.	1

D.14.2 Exercises

Time	Transcript	ID
2:20	[Urgency colors] So, this windows. Hmm. OK. Just trying to see if it stays on. I guess it doesn't stay on if you start dragging– it only stays on if you click once. Oh, well.	146
	[One or More Task Unavailable event] Under what condition will the "One or More Task Unavailable" event occur. Under what condition would the "One or More Task Unav–" huh? Under what condition would the "One or More Task Unavailable" event– so that's event. Ah. [?] I don't remember. I don't remember where event is. Let me see. Control Panel. Tokens. Space? What's in Space? Oh this is how much space I have. Oops. Um. Advice. There's no advice. Hoard Walk. Let me see. Maybe it's in here. Nope, it's not in here (It's in) Tasks? This is the most complicated one. One or more tasks event would be unreadable. Don't know how to answer that question. Does the interface notify the user of this event? What urgency level? Maybe it's in here. Urgency Color. Event Configuration. That's it.	275 147
	[What tasks are currently defined?] There are many! What tasks are currently defined? I'm not gonna type all of 'em, right? What tasks are currently defined? Too lazy to type. I'll just type a few. : Which of these are hoarded? SOSP16. Recommendation letters. Oh. Sigh. Oh. Yicky highlighted color! (But, you) will do.	84
8:30	This 2% doesn't show up very well, so, because the color doesn't show up. It's almost– it's as– it's very misleading, I mean, I look at 2%, and it takes me awhile to realize– well, it's not [mumble].	8
9:50	[Hoard Walk Advice Request arrives] Ew. Something's flashing. Why is it flashing?	267
	[Hoard Walk window may have already been visible.] Hmm. What happened? I click on it and, uh, I don't see a window. Is this it? That's weird! Huh. Oh, well, I look at this instead. Um.	208
12:30	[Looking at Hoard Walk Advice request.] Hmm... I mean, of course I want a Darpa Grant. This is funny. Darpa Grant is not even one of those two choices. But also I want SOSP and I want recommendation letters. Um. : However, um, I'm still a little bit confused about– I'm still a little bit confused about when the task window, er I'm sorry, not the task window, the hoard walk comes up. Why is there, um, another– another one that says the Darpa Grant?	
13:05	[User double-clicks on the Hoard Walk indicator, causing the "Hoard Walk Information" window to appear. He then clicks on the "Help" button on this window, causing the a help window to appear.] Windows are jumping all over the place. That's really strange.	24
14:55	[Miss Event Configurations] [Read question twice.] Advice. [Looking under actual Advice indicator.] Hmm. Weakly connected. I didn't have anything under advice. "Weakly Connected Cache Miss Advice" and their configuration in the space below. Weakly connected. [Double-clicks advice, goes straight to the Advice indicator.] This is (really) strange. I'm not quite sure what this means, but, um, probably, "Weakly Connected Cache Miss Advice" and the	275 147

	"Disconnected Cache Advice" event. Oh– I have to type all this.	
	[How can we indicate that a window is already open?]	208
21:??	I don't need this. How come it does this? Save. This is weird. Oh! It finally showed up. I don't understand why it take so many clicks. Oh! It only takes one click and double-click means that pop up the window. Oops. Didn't understand that earlier. Oh well.	291
22:55	I want all descendants– all descendants. That's weird! Before I press return, I guess it doesn't know I want it so I can't click on these guys. Oh well.	20
	OK. I saved it. OK. Hm. That's weird. Task. Oh this is my window. Coda bug fixes. Oh I want this. Yes. OK. (?) this away.	
	[Header Files task] They are located in several places. Sign. So, am I supposed to do? Am I supposed to include them? Eh. Yicky.	{94}
	:	
25:25	You know something? You know, it might be helpful to have some sort a, of menu driven browse browsing like in Netscape so that I can actually be sure that I– like I didn't type in anything wrong, because this way it's kinda hard to tell whether or not, um, I'm typing something that's meaningful. Just a suggestion.	94
27:00	Another thing is, like, how do you know you need these header files, I mean, it's kinda hard to know. Is there a way that a program knows that the task– I guess it does because it suppose according to the manual he does. Um. Oh well.	185
28:??	I'm not quite understanding this. Um. Tasks and Subtasks. I guess I'm not understanding the distinction very well. Oh well. I'll just start from scratch.	211
29:15	[Save] How come this doesn't go away when I save? That's another interesting, um, thing.	59
	By the way, when you click "close", if something's modified, maybe it's good to have a warning saying that do you, are you sure, the thing has been modified, do you– are you sure you want to close this? I mean, this way I may not, you avoid me from accidently losing my changes, um, there, I meant to make.	127
	[Prioritizing Tasks] Wow! There's so many of them in here. So I guess I should delete 'em, right? How– How do I delete these guys? Would <Backspace> work? Oh, cool. It actually works.	102
	:	
31:??	So a question is that if I'm not hoarding a task, does this mean it'll be removed from my d-disk? Or will it stay there if it's already there? So that's another thing I'm not quite clear on playing with this.	242
	[What day is it today? How many days until Tuesday?]	
32:30	A true drag, it would be better. Um, drag-n-drop, rather than a– cl-cl– you know, the right button kind-of, ah, false things you have. It's slightly confusing without it.	42
	Catastrophic Failure caused by parenthesis in Task name. There was a relatively long break before we were able to continue. During this time frame, the experimenter recovered all of the data from before the failure, and then restarted the Exercises at the point of the original failure. This resulted in the loss of the time for the "PrioritizeTasks" exercise (the point of failure), which in turn meant that the two summary times that included this timing were also invalid.	89
	[Prioritizing Tasks] Wait a minute. CV. Hmm. I forgo– Oh! There we go. Finally. Took a while to move. [User reprioritized task into same position.] Um. Bug fixing. And, ah, Headers– parens. So, another thing is that it would be good if I add something that	42

51:30	thing would be shown on the screen rather than just shown on the scrollbar, I mean, otherwise it's kinda hard for me to see or actually add in something else.	43
54:00	There's something funny about the Task. Why is it blinking red? I've got 100% of everything. Why is it was it blinking? Oh– it's not blinking anymore.	9
	[Doesn't bother reading tetris manual excerpts] Yeah– I think I know how to play this game. I don't need this.	124
55:40	[Moves window]	
	Ah! Space is going cuckoo. Should I keep on playing?	
	OK– I'll send him mail. Where's my xterm? **You do not actually have to take the action– so whatever actions you describe here you do not actually have to take them. Just tell me what you would do.**	
	Please describe all incidents– even those going from bad to good.	
1:01:30	[Does a Hoard Walk– interface does the flip out on the hoard walk progress meter.]	
1:07:10	Advice. Weak miss. We missed the thesis dissertation. Estimated fetch time. My dissertation. Of course. Do I want to fetch? Hmm. I don't remember this. Do I– Yes– I need more time. I don't remember this part. OK. Bad network connection. I never asked to fetch the thesis. Why is it fetching the thesis? I don't want to fetch. That's silly. OK. It's a warning light. I think. I forgot what color it is. Oh. It's red. Got to be red.	149
	[Disconnected Cache Miss Questionnaire: How much will this affect your work?] Not at all; it won't. This is a new one. Network is bad.	149
1:11:20	[Please choose 1-6 (on Disconnected Cache Miss Questionnaire)] Please choose– I did. I choose 0. Please choose a level 1-6 by clicking on your selection. I don't understand– that's a 0. I don't care about it. Well fine. I'll do 1. That is weird. I should be able to say 1.	150
1:14:30	It got stuck. It's not moving. Hey! What's going on? [Submit button is "frozen"] **It'll come back. You don't know the password. It's okay.**	104
1:15:26	Sigh. Reintegration is red. So what the heck does that mean? Token is also bad. Oh I can't fix either one. Reintegrate Now. I don't know what this means. Well, I guess I can read Help. Fine. I'll read help. [Reads] Oh. So I need to reintegrate. Why do I need to reintegrate? I guess something must have changed. OK.	239
	What the heck! You have now completed exercises?! I'm not finished yet. [User was still filling out the final incident report when the final screen popped up.]	151
	[No feedback from "Reintegrate Now".]	

D.14.3 Debriefing

Time	Transcript	ID
	[Difficulty understanding the questions that asked about event configurations.]	275
	[Interested in using fast Coda laptop but not 486-based.]	

[What do you know about hoarding?]
I heard the term before, and then I understand it as the– you select a bunch of things you want and then you hoard it for you. But it's still not quite clear that, at this point, it's not quite clear to me that it actually knows to, for example in emacs, it knows to hoard all the lisp files I need. Or in latex, it hoards all the fonts and the style files. And, it's not clear, you know, that it's doing that.
Even now?

I read something that says something in the tutorial that says you do it a couple times, observing when you run, but that's not enough because what if I'm doing something that's– from– what if I'm starting something, for example all of a sudden I start in lisp mode, and else it can't find certain files. I mean, that's not, not good enough. I need something to be able to let me say "I want them all." (chuckles) I guess I do have that, yeah, in a sense I do, but then I have to search to get the subdirectories myself. So I have to be some sort of expert Unix user or I have to know how this, ah, program is looking at things. I have to understand this program well to know what to hoard. So, in order to do the hoarding right, I guess– guess is already the hard problem. And the thing is that part is never quite, um, emphasized or anything later on– so it's just a vague one line that say that, I had to remember. And so, for example, if I'm clicking on hoarding now– I'm not seeing the same– getting this file, getting this file, getting this file. You know. It's like. Sometimes you could like display something really quickly and then I know what's going on. Or whether it's just hoarding this one thing. 244 243

So you'd like to be able to go down and see what that definition contains? 153
Yes, if I want to.

Yeah, so it's still not clear, right? In the Task, you say get "all descendants" or get one level descendants. But there's still nothing says that when I'm, when I'm running this, it's actually getting things I want automatically. I don't see that– I guess I'm missing that part. Where in the interface is it allowing you to do that?
The programs will do that, but the user data doesn't.
[Explanation]

[Menu-driven browsing– N3 describe what he means– very similar to gnu-emacs– intended to avoid typing pathnames.] 94

[Would like to see indication that window needs to be saved.] 59

[Likes to be able to close without saving to abort.] 154

[Show user how you can click before and after. Finds 'no-man's land' where it doesn't recognize. User's intuition is to drop anywhere on 1st element to drop into 1st spot, as opposed to top half of 1st element!!] 42

[Suggests flash Green → Almost Black.] 9

E Analysis

In this Appendix you will find an analysis of the data and comments found in Appendices C (Experimental Data) and D (Transcripts). This analysis begins by examining the amount of time each user spent on individual screens of the tutorial to determine which aspects of the tutorial (and of the interface) were the most confusing. The analysis continues by looking at user performance on the exercises. It concludes by examining the qualitative data collected—both the user responses to a questionnaire completed at the end of the study and spontaneous comments collecting during the course of the tutorial and exercises. In each section, I present the analysis of the data, a list of findings, and a summary of those findings. I conclude this Appendix with a hierarchical listing of all the findings.

E.1 Identifying Tutorial Difficulties

In this section, I analyze the amount of time each user spent on each screen of the tutorial. I begin the section by presenting a Keystroke-Level Model (KLM) analysis [7] of the amount of time necessary to complete the required activities (such as clicking on this button or double-clicking on another) of each screen. After deducting this time estimate from the amount of time each user spent on the screen, I calculate the user's reading rate on that screen. I then average each user's reading rate over all screens of the tutorial to estimate that user's overall reading rate. Using this reading rate, I identify screens that were either easier or harder for a user to comprehend. I conclude this section with a list of Tutorial Results and a table summarize which users contributed to each result.

E.1.1 Keystroke-Level Model

Each screen of the tutorial required the user to read some text and perform some actions. At a minimum, the user was required to click on an OK button to continue the tutorial. For each screen, we present a KLM model of the required actions. The model includes only those actions that *every* user was expected to take. The same model is applied to

every user; thus, one source of error occurs when the user deviates from the expected actions. These models do not account for reading time.

Mental operators were added to the models according to the rules of Card, Moran, and Newell [7]. Those mental operators appearing in light gray were eliminated according to Rule #1 of their algorithm. I make one additional assumption regarding the use of the **M** operator. That is, an **M** operator does not precede any action the user performs in response to explicit tutorial instructions unless:

1. The instructions leave a decision to the user.
2. The user must search a list of two or more elements to find the specified item in the instructions.
3. Users in the study had difficulty finding the specified item according to the directions given in the tutorial. (Note that further use of the specified item, or similar ones on other screens, do not require additional **M** operators.)

The intuition for this assumption is that the mental preparation for any action taken based upon explicit instruction is overlapped with the reading of that instruction. The exceptions listed above represent situations in which the user's mental preparation during the reading of the instruction is insufficient for performing the action. Any **M** operators appearing in dark gray were eliminated by this assumption.

As part of the models, I include **W** operators. Such an operator is added whenever users might have been required to wait because of the design of the instructions. Because it is impossible to tell how much of that delay overlapped with reading, I ignore all wait times (as indicated by their dark gray color). At worst, the analysis will produce false-positives: screens identified as difficult that are not. The models specify these wait times only to ease *later* identification of these false positives.

Screen ID	Screen Name	Required Actions	KLM Operators
11	Introduction10	Indeterminate wait (ignored)	**W**
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]
13	UrgencyColors1	Point to "Control Panel" indicator light	**MP**[indicator]
		Double-click left button	**MBBBB**[left]
		System Response Time	**R**
		Point to "Urgency Colors" tab	**MP**[command]
		Click left button	**MBB**[left]
		System Response Time	**R**
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]

14	Urgency Colors2	Indeterminate number of color choices:	(Assume 1)
		Point to down arrow button	**MP**[command]
		Click left button	**MBB**[left]
		Point to new color name	**MP**[command]
		Click left button on new color name	**MBB**[left]
		Point to commit button	**MP**[command]
		Click left button	**MBB**[left]
		Point to close button	**MP**[command]
		Click left button	**MBB**[left]
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]
16	Network2	Indeterminate wait (minimal – ignored)	**W**[1.5sec]
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]
19	Task2	Point to scrollbar	**MP**[command]
		Click left button	**MBB**[left]
		Click left button	**MBB**[left]
		Point to "writing thesis" entry	**MP**[command]
		Double-click left button	**MBBBB**[left]
		System Response Time	**R**
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]
20	Task3	Search for three sections	**M**
		Search for two subsections	**M**
		Look at User Data's left and right lists	**M**
		Look at "writing thesis" definition	**M**
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]
21	Task4	Search for choices	**M**
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]
22	Task5 (See Open Window)	Search for choices	**M**
23	Task6	Point to scrollbar	**MP**[command]
		Click left button	**MBB**[left]
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]
24	Task7	Point to scrollbar	**MP**[command]
		Click left button	**MBB**[left]

		Point to scrollbar	**MP**[command]
		Click left button	**MBB**[left]
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]
25	Task8	Point to down arrow button	**MP**[command]
		Click left button	**MBB**[left]
		Point to "New..."	**MP**[command]
		Click left button	**MBB**[left]
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]
26	Task9	Home to keyboard	**H**[keyboard]
		Type new name (15 characters)	**M15K**[name]
		Type <Return>	**MK**[<Return>]
		Home to mouse	**H**[mouse]
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]
27	Task10	Home to keyboard	**H**[keyboard]
		Indeterminate number of paths entered:	(Assume 2)
		Type path (~15 characters)	**M15K**[path]
		Type <Tab>	**MK**[<Tab>]
		Press <Tab> twice	**M2K**[<Tab>]
		Type <Return>	**M1K**[<Return>]
		Home to mouse	**H**[mouse]
		Point to SAVE button	**MP**[command]
		Click left button	**MBB**[left]
		Point to OK button	**MP**[command]
		Click left button	**MBB**[left]
29	Task12 (See Open Window)	Find "screen capture"	**M**
32	Task15	Point to down arrow button	**MP**[command]
		Click left button	**MBB**[left]
		Point to "New..."	**MP**[command]
		Click left button	**MBB**[left]
		Home to keyboard	**H**[keyboard]
		Type name and hit <Return> (13 chars)	**M13K**[name]
		Type path1 and hit <Return> (22 chars)	**M22K**[path]
		Type path2 and hit <Return> (18 chars)	**M18K**[path]
		Home to mouse	**H**[mouse]
		Point to save button	**MP**[command]
		Click left button	**MBB**[left]
		Point to close button	**MP**[command]

		Click left button	MBB[left]
		Point to OK button	MP[command]
		Click left button	MBB[left]
33	Task16	Point to down arrow button	MP[command]
		Click left button	MBB[left]
		Point to "writing"	MP[command]
		Click left button	MBB[left]
		Home to keyboard	H[keyboard]
		Type " with latex" (11 chars)	M11K[name]
		Home to mouse	H[mouse]
		Point to save button	MP[command]
		Click left button	MBB[left]
		Point to OK button	MP[command]
		Click left button	MBB[left]
34	Task17	Point to name entry area	MP[command]
		Click left button	MBB[left]
		Home to keyboard	H[keyboard]
		Delete "latex" (5 chars)	M5K[delete]
		Type "scribe" and <Return> (7 chars)	M7K[name]
		Home to mouse	H[mouse]
		Point to "latex" program	MP[command]
		Click left button	MBB[left]
		Home to keyboard	H[keyboard]
		Hit <Delete> key	MK[delete]
		Home to mouse	H[mouse]
		Point to "xdvi" program	MP[command]
		Click left button	MBB[left]
		Home to keyboard	H[keyboard]
		Hit <Delete> key	MK[delete]
		Home to mouse	H[mouse]
		Point to "scribe tools" program	MP[command]
		Click left button	MBB[left]
		Point to save button	MP[command]
		Click left button	MBB[left]
		Point to OK button	MP[command]
		Click left button	MBB[left]
36	Task19	Select three tasks for hoarding:	Requires 3 each of:
		Point to task name	MP[command]
		Click left button	MBB[left]
		Point to hoarded tasks list	MP[command]
		Click right button	MBB[right]
		Point to OK button	MP[command]
		Click left button	MBB[left]

40	Advice2	Indeterminate wait (ignored)	W[10sec]
		Point to weak miss entry	MP[command]
		Double-click left button	MBBBB[left]
		System Response Time	R
		Point to DON'T FETCH button	MP[command]
		Click left button	MBB[left]
		Point to OK button	MP[command]
		Click left button	MBB[left]
42	Advice4	15 second wait (reading may overlap)	W[13sec]+W[15sec]
		Point to YES, PLEASE WAIT...	MP[command]
		Click left button	MBB[left]
		Point to chosen button	MP[command]
		Click left button	MBB[left]
		Point to OK button	MP[command]
		Click left button	MBB[left]
44	HoardWalk1	Point to "Hoard Walk" indicator light	MP[indicator]
		Double-click left button	MBBBB[left]
		System Response Time	R
		Point to WALK NOW button	MP[command]
		Click left button	MBB[left]
		Indeterminate wait	W
		Point to close button	MP[command]
		Click left button	MBB[left]
		Point to OK button	MP[command]
		Click left button	MBB[left]
45	HoardWalk2 (See Open Window)	Wait for hoard walk advice request	W[4sec]
48	HoardWalk5	Indeterminate number, N, of node expansion/contraction pairs:	(Assume 4)
		Point to node	MP[command]
		Double-click left button	MBBBB[left]
		Point to OK button	MP[command]
		Click left button	MBB[left]
49	HoardWalk6	Point to FETCH? Button	MP[command]
		Click left button	MBB[left]
		Point to FINISH HOARD WALK button	MP[command]
		Click left button	MBB[left]
		Point to close button	MP[command]
		Click left button	MBB[left]
		Point to OK button	MP[command]
		Click left button	MBB[left]

50	Space1	Event delay	W[3sec]
		Point to "Space" indicator light	MP[indicator]
		Double-click left button	MBBBB[left]
		System Response Time	R
		Point to HELP button	MP[command]
		Click left button	MBB[left]
		Point to help window's OK button	MP[command]
		Click left button	MBB[left]
		Point to OK button	MP[command]
		Click left button	MBB[left]
52	Space3	Event delay	W[3sec]
		Point to help button	MP[command]
		Click left button	MBB[left]
		Point to help window's ok button	MP[command]
		Click left button	MBB[left]
		Point to OK button	MP[command]
		Click left button	MBB[left]
53	Space4 (See Close Window)	Event delay	W[1.5sec]
59	EventConfiguration3	Point to indicator listbox	MP[command]
		Click left button	MBB[left]
		Point to "Advice"	MP[command]
		Click left button	MBB[left]
		Point to event listbox	MP[command]
		Click left button	MBB[left]
		Point to "Read Disconnected Cache Miss"	MP[command]
		Click left button	MBB[left]
		Point to indicator listbox	MP[command]
		Click left button	MBB[left]
		Point to "Network"	MP[command]
		Click left button	MBB[left]
		Point to event listbox	MP[command]
		Click left button	MBB[left]
		Point to "Operating Disconnected"	MP[command]
		Click left button	MBB[left]
		Point to OK button	MP[command]
		Click left button	MBB[left]

15	Open Window	Point to choice	**MP**[choice]
18		Double-click left button	**MBBBB**[left]
22		System Response Time	**R**
29		Point to OK button	**MP**[command]
39		Click left button	**MBB**[left]
45			
47			
54			
57			
17	Close Window	Point to close button	**MP**[command]
28		Click left button	**MBB**[left]
35		Point to OK button	**MP**[command]
38		Click left button	**MBB**[left]
41			
53			
56			
61			
1-10	Continue Tutorial	Point to OK button	**MP**[command]
12		Click left button	**MBB**[left]
30-1			
37			
43			
46			
51			
55			
58			
60			
62			

Figure E.1: GOMS Analysis of Tutorial Screens

This table contains a GOMS analysis of each screen of the tutorial that requires the user to perform one or more actions. Any screens not explicitly listed in the table are assumed to require the user to click on the OK button, as analyzed in the final entry of the table. The GOMS operators shown in the right-most column of this table, with the exception of the **W** operator, are defined by Card, Moran, and Newell [7]. The **W** operator represents the maximum amount of time the user could have to wait for an event to occur. Because these wait times largely overlap with reading time, they are ignored for the remaining analysis.

Screen ID	Word Count	GOMS Operators W	K	B	P	H	M	R
Initial	??	0	0	0	0	0	0	0
Introduction1	86	0	0	2	1	0	1	0
Introduction2	65	0	0	2	1	0	1	0
Introduction3	68	0	0	2	1	0	1	0
Introduction4	206	0	0	2	1	0	1	0
Introduction5	226	0	0	2	1	0	1	0
Introduction6	162	0	0	2	1	0	1	0
Introduction7	257	0	0	2	1	0	1	0
Introduction8	50	0	0	2	1	0	1	0
Introduction9	219	0	0	2	1	0	1	0
Introduction10	242	0	0	2	1	0	1	0
Introduction11	109	0	0	2	1	0	1	0
UrgencyColors1	81	0	0	8	3	0	1	2
UrgencyColors2	341	0	0	10	5	0	4	0
Network1	150	0	0	6	2	0	1	1
Network2	31	0	0	2	1	0	1	0
Network3	27	0	0	4	2	0	1	0
Task1	177	0	0	6	2	0	1	1
Task2	157	0	0	10	3	0	4	1
Task3	188	0	0	2	1	0	5	0
Task4	31	0	0	2	1	0	2	0
Task5	35	0	0	6	2	0	2	1
Task6	110	0	0	4	2	0	2	0
Task7	303	0	0	6	3	0	3	0
Task8	54	0	0	6	3	0	3	0
Task9	57	0	16	2	1	2	3	0
Task10	165	0	38	4	2	2	7	0
Task11	19	0	0	4	2	0	1	0
Task12	85	0	0	6	2	0	2	1
Task13	135	0	0	2	1	0	1	0
Task14	49	0	0	2	1	0	1	0
Task15	138	0	53	10	5	2	2	0
Task16	116	0	11	8	4	2	2	0
Task17	154	0	14	12	6	6	3	0
Task18	62	0	0	4	2	0	1	0
Task19	147	0	0	14	7	0	7	0
Task20	226	0	0	2	1	0	1	0
Task21	38	0	0	4	2	0	1	0
Advice1	199	0	0	6	2	0	1	1
Advice2	115	0	0	8	3	0	1	1
Advice3	255	0	0	4	2	0	1	0
Advice4	105	0	0	6	3	0	2	0
Advice5	95	0	0	2	1	0	1	0
HoardWalk1	157	0	0	10	4	0	1	1
HoardWalk2	141	0	0	6	2	0	1	1
HoardWalk3	261	0	0	2	1	0	1	0
HoardWalk4	150	0	0	6	2	0	1	1
HoardWalk5	166	0	0	18	5	0	5	0
HoardWalk6	70	0	0	8	4	0	1	0
Space1	161	0	0	10	4	0	2	1
Space2	30	0	0	2	1	0	1	0
Space3	75	0	0	6	3	0	2	0
Space4	17	0	0	4	2	0	1	0
Tokens1	54	0	0	6	2	0	1	1
Tokens2	34	0	0	2	1	0	1	0
Tokens3	17	0	0	4	2	0	1	0
EventConfig1	133	0	0	6	2	0	1	1
EventConfig2	255	0	0	2	1	0	1	0
EventConfig3	67	0	0	18	9	0	5	0
EventConfig4	29	0	0	2	1	0	1	0
EventConfig5	35	0	0	4	2	0	1	0
TokenExpiry	111	0	0	2	1	0	1	0

Figure E.2: Count of GOMS Operators per Tutorial Screen

For each screen in the Tutorial, this table shows the number of words the users had to read (including words shown in Help screens users were directed to read) as well as the number of actions users were required to take to complete the screen of instructions. The actions come in seven forms. User could be required to wait (**W**), though all wait times are ignored; they could be required to type some characters (**K**); they could be required to click the mouse (**B**); they could be required to point with the mouse (**P**); they could be required to move their hands from the mouse area to the keyboard area (**H**); they could be required to make a decision to perform some action (**M**); or they could be required to wait for the system to display a window of the interface (**R**). For a detailed listing of the actions required for each screen, please refer to Section E.1.1.

Operator	Estimated Time Required
K	0.20
B[button]	0.10
P[location]	1.10
H[location]	0.40
M	1.20
R	0.10

Figure E.3: Operators of the Keystroke-Level Model

The **K**, **B**, **P**, **H**, and **M** operators estimate the amount of time necessary to type a single key (both the up-stroke and the down-stroke), click a single mouse button (either the up-stroke or the down-stroke), point to a location on the screen with the mouse, move the hands to either the mouse or the keyboard, and mentally prepare for some action respectively [7,21]. The **R** operator represents system response time and is assumed to be 0.1 seconds for all instances appearing in the models. (Actual display times for individual windows appear in Figure C.15)

E.1.2 Screen Difficulty Analysis

Using the GOMS models presented in Section E.1.1 above, I calculate the amount of time each user spent comprehending each screen of the Tutorial. Given the total amount of time the user spent on a given screen, I subtract the time necessary to perform all required actions. This later time is calculated by the following equation:

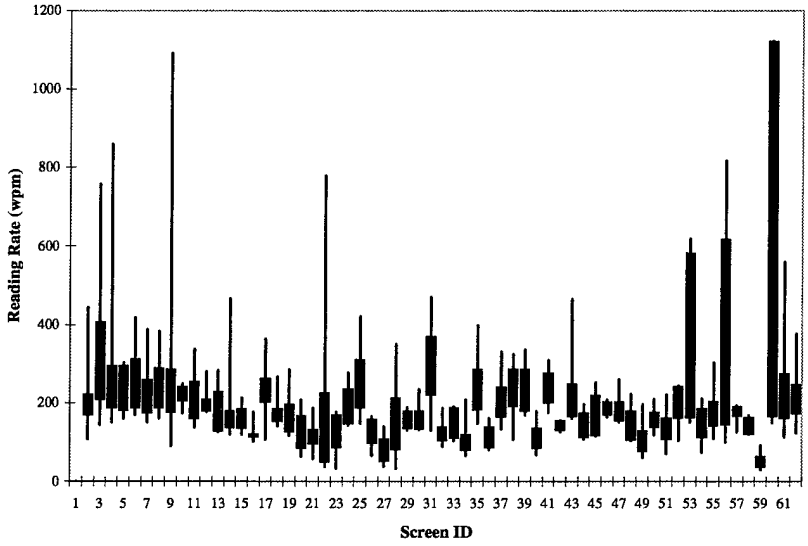
$$ActionTime = N_K T_K + N_B T_B + N_P T_P + N_H T_H + N_M T_M + N_R T_R$$

where N_X = the number of X operators, and

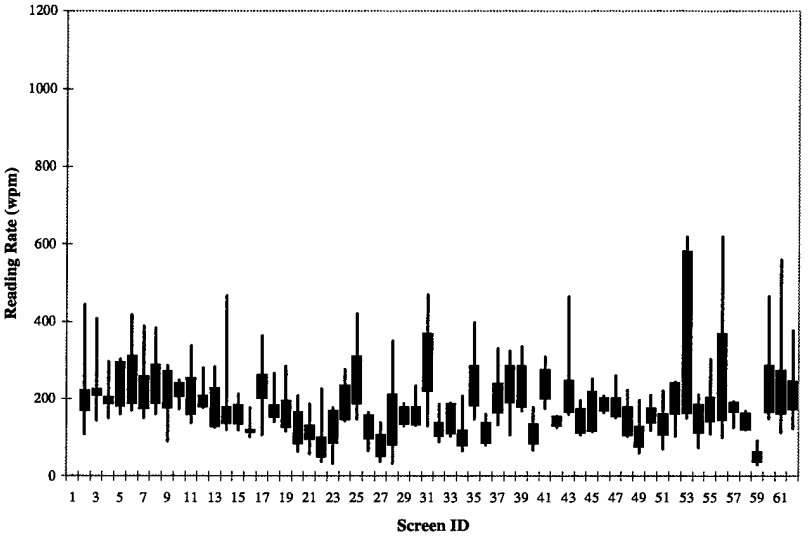
T_X = the time required for an X operator.

The values for N_X for each screen appear in Figure E.2. The values of the T_X appear in Figure E.3

After calculating the total amount of time a user spent understanding a given screen of the tutorial, I can use the total word count of that screen to calculate the user's screen reading rate. Figure E.4(a) shows the range of reading rates on each screen of the tutorial. The line identifies the high and low reading rates; the bar represents the range excluding the highest and lowest values. Participants whose reading rates were much above 600 wpm could not have read the entire tutorial; they had to have been skimming [7]. For this reason, I eliminated any reading rates above 652 wpm (a total of seven screens from four users) and recalculated each user's average and standard deviation. The resulting graph is shown in Figure E.4(b).



(a) Original Reading Rates

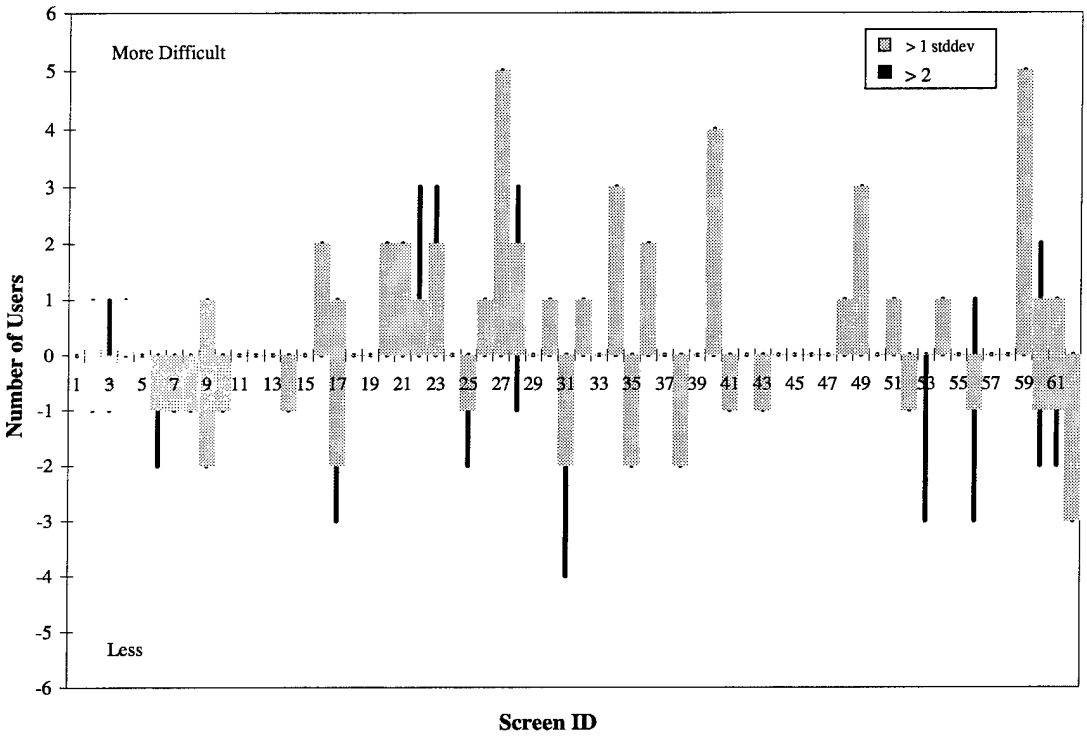


(b) Reading Rates (sans skimming)

Figure E.4: Range of Reading Rates

These graphs show the range of reading rates observed from the six novice and expert users. View (a) shows the original reading rates; view(b) shows the reading rates after elimination of those rates above the expected skimming rate. Each screen of the Tutorial is shown along the x-axis (Screen ID numbers correspond to those of Appendix B). The line shown for each screen represents the range. The bar represents the range after excluding the minimum and maximum reading rates.

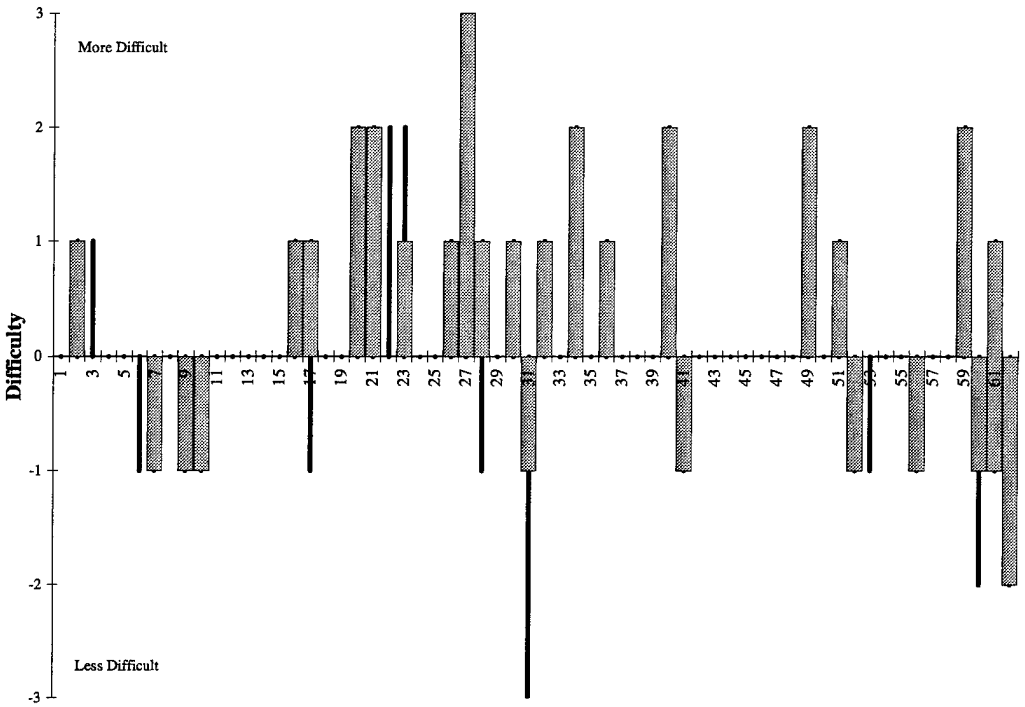
By averaging a user's screen reading rates over all screens of the tutorial, I estimate the user's overall reading rate and standard deviation. Using these values, I can identify as difficult those screens that slowed the user down well below their average reading rate. Figure E.5 shows three figures. The first one identifies screen difficulty as measured across all users. The second two identifies screen difficulty as measured across only the novice users and only the expert users.



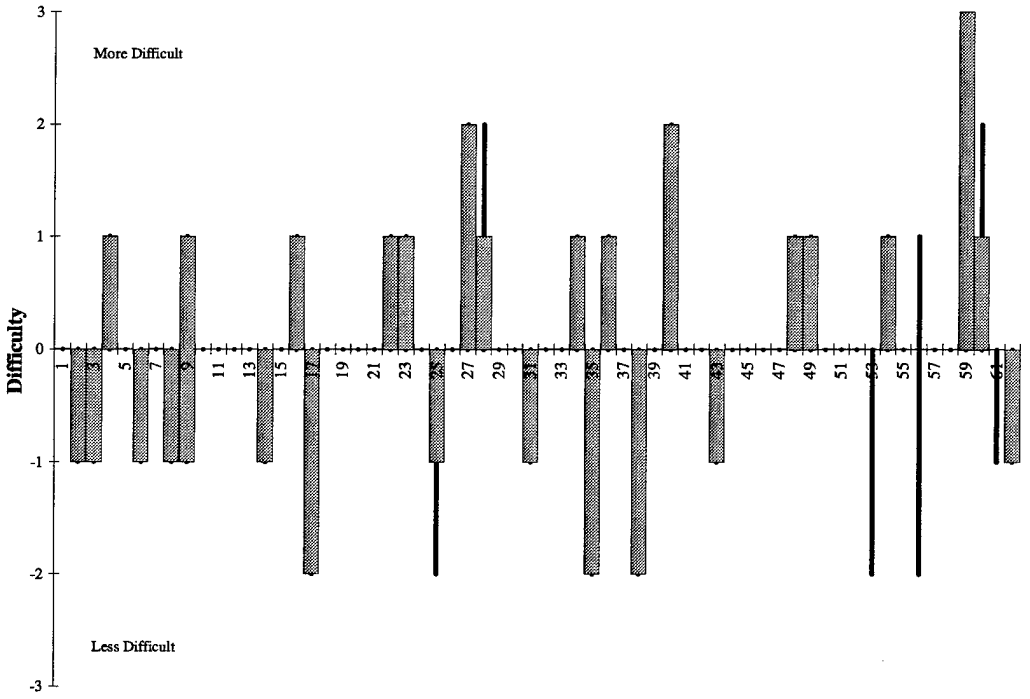
(a) All

Figure E.5: Screen Difficulty Analysis

These three graphs show the results of the screen difficulty analysis. Along the x-axis we see each screen of the Tutorial, as identified in Appendix B . Along the y-axis we see the number of users finding each screen more or less difficult. Each bar represents the total number of users whose reading rate fell outside of one standard deviation of their average reading rate. Each line represents the total number of users whose reading rate fell outside two standard deviations. View (a) presents the results of including both the novice and expert users. View (b) examines just the novice users. View (c) examines just the expert users.



(b) Novice Users Only



(c) Expert Users Only

E.1.3 List of Tutorial Results

Tutorial Result #1: (Screen 16) Two users had difficulty with the "Operating Disconnected" event.

Tutorial Result #2: (Screen 20) Two novice users experienced difficulty orienting to the "Task Definition" window.

Tutorial Result #3: (Screen 21) Two novice users had difficulty understanding explanation of the different ways to view task definitions.

Tutorial Result #4: (Screen 22) Three users had difficulty finding the "writing thesis" task in the "Task Definition" window (under either the "predefined list" or the "contains" list).

Tutorial Result #5: (Screen 23) Three users experienced difficulty orienting to the "Data Definition" window.

Tutorial Result #6: (Screen 27) Five users had difficulty entering data into the "Data Definition" window.

Tutorial Result #7: (Screen 28) Three users had difficulty closing "Data Definition" window.

Tutorial Result #8: (Screen 34) Three users had difficulty modifying a task definition.

Tutorial Result #9: (Screen 36) Two users had difficulty prioritizing tasks or understanding the "Task Unavailable" event notification.

Tutorial Result #10: (Screen 40) Four users experienced difficulty orienting to the "Advice Information" window.

Tutorial Result #11: (Screen 49) Three users experienced difficulty completing hoard walk advice request.

Tutorial Result #12: (Screen 59) Five users had difficulty examining event configurations.

Tutorial Result #13: (Screen 60) Two expert users had difficulty confirming their answer regarding event configuration.

E.1.4 Summary of Tutorial Results

Tutorial	Novice Users			Expert Users		
Screen ID	N1	N2	N4	E1	E2	E3
16: Network2			✓			✓
20: Task3	✓	✓				
21: Task4		✓	✓			
22: Task5	✓		✓		✓	
23: Task6	✓	✓				✓
27: Task10	✓	✓	✓	✓	✓	
28: Task11		✓		✓	✓	
34: Task17	✓	✓		✓		
36: Task19		✓		✓		
40: Advice2		✓	✓	✓	✓	
49: HoardWalk6	✓	✓		✓		
59: EventConfiguration3		✓	✓	✓	✓	✓
60: EventConfiguration4				✓		✓

Figure E.6: Difficult Tutorial Screens by User

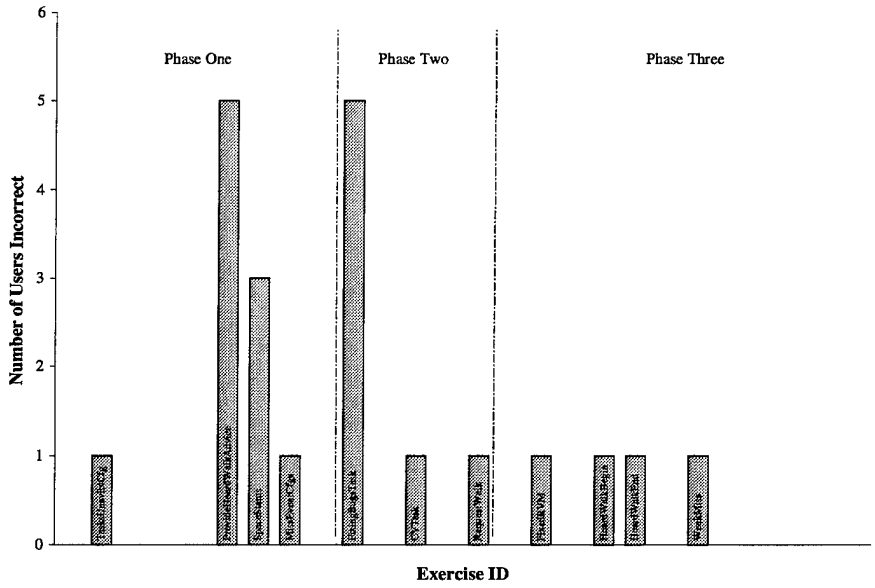
This table shows which user's reading performance suggests that an individual screen of the Tutorial was difficult. A checkmark indicates the user read the screen at a rate at least one standard deviation below their average reading rate.

E.2 Identifying Exercise Difficulties

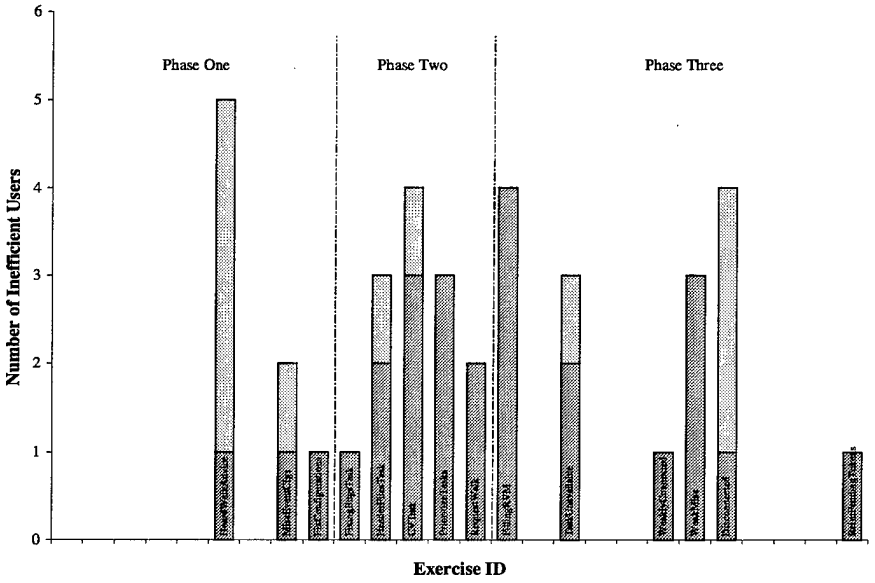
In this section, I analyze the user's responses to each of the exercise. My goal is to identify those exercises which users found difficult. I use two metrics in this analysis. Exercises are identified as difficult if more than one user responded incorrectly or if more than one user responded inefficiently. In this section, I begin by presenting the two metrics as well as a graphical representation of the data. I then list each of the findings. I conclude the section with a table summarizing which users provided the evidence for each finding.

E.2.1 Metrics

In this section, I describe the two metrics used in identifying difficult exercises. The first metric looks at the accuracy of each user's response. Accuracy is measured by comparing the user's response to a key. For each correct item, the user scores a point. These scores appear in Figure C.13. The second looks at the efficiency of each user's response. Efficiency is measured by counting the number of actions a user took to respond to an exercise and comparing that value to the number required for an expert user (expert Coda user *and* expert interface user) with foreknowledge of the question. These scores appear in Figure C.14. In Figures E.7(a) and E.7(b), you will find bar charts showing the number of users who responded incorrectly and the number responding inefficiently to each exercise.



(a) Number of Users Answering Incorrectly



(b) Number of Users Answer Inefficiently

Figure E.7: Exercise Analysis

These two figures show the results of the exercise difficulty analysis. In each figure, the X-axis shows Exercise ID and the Y-axis shows the number of users. View (a) shows the number of users who answered a given exercise incorrectly. The lack of a bar signifies that no users answered that exercise incorrectly. View (b) shows the number of users who answered a given exercise inefficiently. The dark gray segment shows the number of users exceeding 2*PAR on a given exercise while the light gray segments shows the number exceeding 3*PAR. The lack of a bar signifies that no users answered that exercise inefficiently.

E.2.2 List of Exercise Results

Exercise Result #1: Five users provided incorrect hoard walk advice.

Exercise Result #2: Three users could not accurately describe the current space status.

Exercise Result #3: Five users failed to completely define the "fixing bugs" task.

Exercise Result #4: Five users provided hoard walk advice inefficiently.

Exercise Result #5: Two users found the events to be configured inefficiently.

Exercise Result #6: Three users defined "Header Files" task inefficiently.

Exercise Result #7: Four users defined "CV" task inefficiently.

Exercise Result #8: Three users prioritized tasks inefficiently.

Exercise Result #9: Two users requested hoard walk inefficiently.

Exercise Result #10: Four users described "RVM Filling" event inefficiently.

Exercise Result #11: Three users described "Task Unavailable" event inefficiently.

Exercise Result #12: Three users provided "Weak Miss" advice inefficiently.

Exercise Result #13: Four users described "Operating Disconnected" event inefficiently.

Exercise Result #14: Two users reorganized task to hoard subtask at a different priority.

E.2.3 Summary of Exercise Results

	Novice Users			Expert Users		
ID	N1	N2	N4	E1	E2	E3
1		✓	✓	✓	✓	✓
2	✓			✓	✓	
3	✓	✓	✓		✓	✓
4	✓	✓	✓		✓	✓
5	✓					✓
6				✓	✓	✓
7	✓	✓	✓	✓		
8		✓	✓			✓
9			✓			✓
10		✓	✓		✓	✓
11	✓	✓		✓		
12	✓	✓		✓		
13	✓	✓	✓			✓
14		✓	✓			

Figure E.8: Exercise Findings by User

This table shows which users had difficulty with particular aspects of the exercises, including the accuracy and efficiency of their response.

E.3 Identifying Qualitative Opinions

In this section, I summarize the comments users made on the Evaluation Questionnaire. The raw data are presented in Appendix C.3. I begin by presenting a list of general comments made on the Evaluation Questionnaire. I then summarize that list to indicate which users made each of the comments.

E.3.1 List of Qualitative Results

Qualitative Result #1: Users did not appreciate question about reintegration after it had not been covered in the Tutorial.

Qualitative Result #2: Expert users had difficulty understanding distinction between <Tab> and <Enter>.

Qualitative Result #3: Expert users thought the Tutorial went into too much detail.

Qualitative Result #4: Expert user was confused by the distinction between the Hoard Walk indicator and the Task indicator.

Qualitative Result #5: Users had difficulty finding events in the Event Configuration tab of the Control Panel.

Qualitative Result #6: Novice user thought the "first level interface" provided too little detail to determine what was going on.

Qualitative Result #7: Expert user confused by the "no such file" indication.

Qualitative Result #8: Expert user finds he must explicitly think about how to structure his task definitions.

Qualitative Result #9: Expert user finds editing file/directory lists (in "Data Definition" windows) tricky

Qualitative Result #10: Expert user unsure whether or not a program was available.

Qualitative Result #11: Novice user dislikes having to enter pathnames.

Qualitative Result #12: Novice user accidentally deletes entire task rather than single element.

Qualitative Result #13: Novice user dislikes having fetch time expressed in seconds.

Qualitative Result #14: Expert user confused by behavior of "Save" buttons, in particular whether all tasks are saved at once.

Qualitative Result #15: Expert user confused by creating a new task vs. changing the name of an old one.

Qualitative Result #16: Expert user afraid of missing an event notification.

Qualitative Result #17: Expert user finds graphical user interfaces tedious.

Qualitative Result #18: Expert user would like to have the ability to stop long transfers over weak network connections.

Qualitative Result #19: Users like being able to see the status of the network and of hoard walks.

Qualitative Result #20: Users like being able to configure the event notifications.

Qualitative Result #21: Users like the "easy to define" tasks.

Qualitative Result #22: User likes that the system keeps track of the importance of missing objects.

Qualitative Result #23: Expert user likes that the system automatically tracks of file accesses for programs.

E.3.2 Summary of Qualitative Results

	Novice Users			Expert Users		
ID	N1	N2	N4	E1	E2	E3
1	✓			✓	✓	✓
2				✓	✓	
3				✓	✓	
4				✓		
5	✓					✓
6	✓					
7						✓
8				✓		
9					✓	
10					✓	
11	✓					
12		✓				
13			✓			
14					✓	
15					✓	
16					✓	
17						✓
18					✓	
19		✓			✓	✓
20			✓	✓		
21	✓	✓	✓		✓	
22		✓				
23					✓	

Figure E.9: Qualitative Findings by User

This table shows which users made each of the qualitative comments listed above.

E.4 Identifying Spontaneous Opinions

In this section, you will find a listing of the comments made by users during the course of the usability study. Each comment is given a unique number. The listing presents the comments in numerical order for easy reference. Following the listing, I present a table summarizing which users made each comment at what point(s) during the study.

E.4.1 List of Spontaneous Comments

Comment #1: Participants frustrated by window placement.

Comment #2: (See Comments #80 and #209-234).

Comment #3: Participants have difficulty entering data in Data Definition window.

Comment #4: Windows do not resize.

Comment #5: Participants confused that changing name and saving creates a new defn.

Comment #6: Participant expects a button to delete an element from the "contains" list.

Comment #7: Interface alphabetizes caps and uncaps.

Comment #8: Meters shown at 1% have no visible color.

Comment #9: User confused by two-color blinking.

Comment #10: User confused that two indicators open the same advice request.

Comment #11: Users confused by different windows appearing from one indicator light.

Comment #12: User expects <Enter> to make the help window disappear.

Comment #13: Users confused when help window grabs and maintains focus.

Comment #14: (See Comments #249-252.)

Comment #15: System should predict if hoarding a task will cause space problem.

Comment #16: (See Comments #268-272.)

Comment #17: User guesses location of event in indicator list.

Comment #18: Users have difficulty finding definitions in predefined lists.

Comment #19: User loses data after selecting task from listbox.

Comment #20: User wants to click on disabled meta-information.

Comment #21: User concerned that entire pathname is not visible.

Comment #22: System double-selects under certain circumstances.

Comment #23: User finds it strange that a single-click selects an item for "contains" list.

Comment #24: User experiences window overload.

Comment #25: User wants feedback regarding "health" of Venus.

Comment #26: User has difficulty finding task in Task Information window.

Comment #27: (See Comment #261.)

Comment #28: User confused about what object needs to be fetched.

Comment #29: User confused by file becoming unavailable.

Comment #30: Password visible on Authentication Information window.

Comment #31: User expresses confusion between Hoard Walk and Task indicators.

Comment #32: User expects "commit" to close Control Panel window (as feedback).

Comment #33: User confused because indicator stops blinking.

Comment #34: User has difficulty finding "black down-arrow button".

Comment #35: Users think they should literally make up pathnames.

Comment #36: User confused by lack of feedback when directory does not exist.

Comment #37: User bothered by addition of empty entry widget.

Comment #38: User confused about how system notifies if "everything" is turned off.

Comment #39: User confused about extent of "commit" button's action.

Comment #40: (See Comment #41.)

Comment #41: User concerned that he might miss an indicator state change.

Comment #42: User has difficulty prioritizing tasks.

Comment #43: Newly hoarded task may not be visible in list of hoarded tasks.

Comment #44: User dislikes font choice.

Comment #45: User expects "help" on the indicator lights window.

Comment #46: User surprised that a double-click is required to open indicator window.

Comment #47: Screen of tutorial partially covers a window of the interface.

Comment #48: (See Comment #1.)

Comment #49: (See Comments #253-262.)

Comment #50: Meters do not cover multiple orders of magnitude well.

Comment #51: Meters do not convey relative information.

Comment #52: Nothing is hoarded, yet 70% of cache is full.

Comment #53: User finds the scrollbars ugly.

Comment #54: User annoyed that a list with six elements must be scrolled.

Comment #55: User doesn't recognize slider lines as widgets controlling screen layout.

Comment #56: User dissatisfied with Data Definition window.

Comment #57: User wants Data Definition window to indicate the size of subtree.

Comment #58: User dislikes default meta-information for directories.

Comment #59: User expected "save" button to also close the window.

Comment #60: User unsure as to why there is a "save as" button.

Comment #61: User doesn't believe system ensures program executables exist.

Comment #62: User has difficulty deselecting a chosen element from "contains" list.

Comment #63: User confused about how to select tasks to be hoarded.

Comment #64: User would like to drag tasks within the list of hoarded tasks.

Comment #65: (See Comments #253-262.)

Comment #66: Requests needing immediate attention require too many steps.

Comment #67: User concern that 15 seconds is wrong default timeout.

Comment #68: User would like a count-down timer on advice requests.

Comment #69: (See Comment #253-262.)

Comment #70: User dislikes checkboxes, would prefer checkmarks.

Comment #71: User dislikes "All Tasks Needing Data" pseudo-task.

Comment #72: User notices an upper-case/lower-case inconsistency.

Comment #73: Peer mistaken for child in tree widget.

Comment #74: Task indicator turns green before hoard walk completes.

Comment #75: Space help button is too small.

Comment #76: User wants expert help for space problems.

Comment #77: Number unreadable on meter.

Comment #78: User dislikes directive to not use "Destroy Tokens" and "New Tokens".

Comment #79: Urgency labels on the Event Configuration Tab are not lined up.

Comment #80: Distinction between different modes of operation unclear.

Comment #81: User surprised he must commit changes to the event configurations.

Comment #82: Priorities need label.

Comment #83: User notices inconsistent task name between exercises and interface.

Comment #84: Color used to highlight in Exercises window bad with pink background.

Comment #85: End of task name is hidden behind meter.

Comment #86: User does not believe the Space meter (cache).

Comment #87: User wants to modify all event notifications at once.

Comment #88: User would prefer more obvious notification of strong connectivity.

Comment #89: Certain characters, used in task names, can crash the interface.

Comment #90: User would like "balloon" help.

Comment #91: User dislikes the Urgency Colors tab of the Control Panel.

Comment #92: User wants "undo".

Comment #93: User curious how the system copes with links.

Comment #94: User would prefer not to type in pathnames.

Comment #95: Space Help window did not update to show new information.

Comment #96: Seconds too fine-grained.

Comment #97: User "saves" definition under the name "New..." and it is lost.

Comment #98: User confused by options in the weak miss timeout prod.

Comment #99: User confused by bogus Fetch Statistics in Hoard Walk Advice window.

Comment #100: User confused by instruction to define header files as a program.

Comment #101: User expects arrow keys to work in the meta-information area.

Comment #102: Users have difficulty unhoarding tasks.

Comment #103: User expects system to track and automatically hoard missing objects.

Comment #104: Submitting a password takes excessively long.

Comment #105: (See Comments #264-266.)

Comment #106: Tutorial assumes green/yellow/red.

Comment #107: User types pathname, hits a key near <Return>, and it disappears.

Comment #108: User looks under indicator light rather than indicator listbox.

Comment #109: (See Comments #253-267.)

Comment #110: User would like a tree view of Tasks.

Comment #111: User has difficulty identifying the indicator lights.

Comment #112: (See Comments #281-292.)

Comment #113: User concerned that he would use "all descendants", "all the time."

Comment #114: User thinks it strange that we have a meter on the Tasks.

Comment #115: User thinks it strange that we have both kinds of advice under Advice.

Comment #116: (See Comments #235-237.)

Comment #117: User believes the Tutorial to be out-of-sequence.

Comment #118: User confused as to what exactly is going to "flash".

Comment #119: User does not recognize Event Configuration window.

Comment #120: Advice indicator goes green too soon.

Comment #121: User disturbed by window popping up.

Comment #122: User double-clicks in empty contains list and "New..." pops up.

Comment #123: User dislikes use of predefined and contains lists.

Comment #124: User failed to read instructions completely.

Comment #125: User expects meta-information for programs.

Comment #126: User unsure which window allows him to prioritize tasks.

Comment #127: User closes window without saving and loses data.

Comment #128: User notices omissions in the mock task definitions.

Comment #129: User wants interface to show what files are located on what servers.

Comment #130: User would like to be able to highlight in tutorial (as a bookmark).

Comment #131: User has difficulty finding scrollbar.

Comment #132: User expects <Back-Tab> to work in meta-information.

Comment #133: User wants cut & paste to work.

Comment #134: User disturbed by instruction to "close all windows" (only one open).

Comment #135: Visible shift when switching between definitions/configurations.

Comment #136: User confused by having to hit <Enter> after typing name of definition.

Comment #137: User types a pathname into the name of the definition.

Comment #138: User has difficulty deleting a definition.

Comment #139: User experienced an unexplained loss of a definition.

Comment #140: User confused by the "unselect" dialog box.

Comment #141: User confused by "Fetch" and "Stop Asking" toggling one another.

Comment #142: User wants the triangle icons rather than the "+" and "-" icons.

Comment #143: User would prefer the indicator lights to be alphabetized.

Comment #144: User prefers klist to Tokens indicator.

Comment #145: User would like to have a set of system defaults.

Comment #146: User would like to drag colors.

Comment #147: User searches for way to configure events.

Comment #148: (See Comments 238-248.)

Comment #149: User has forgotten primary "task".

Comment #150: User wants to be able to say "0" to Disconnected Miss Query.

Comment #151: User upset that Exercises tell him that he's done when he isn't.

Comment #152: (See Comments #219, 274 and 275.)

Comment #153: User would like to be able to see the details of program definitions.

Comment #154: User prefers for close without saving/committing to abort.

Comment #155: User doesn't recognize Hoard Walk Advice as a request for advice.

Comment #156: User confused by Hoard Walk Advice coming in under Hoard Walk.

Comment #157: User thinks it silly to have to create a task for a single directory.

Comment #158: User surprised that system saves definition automatically upon close.

Comment #159: User surprised that indicator lights are dynamic.

Comment #160: User would prefer the Urgency of Event Configurations to be in color.

Comment #161: User would prefer configuration options be checkboxes (not listboxes).

Comment #162: User suggests improved layout for the event configuration tab.

Comment #163: User thinks it strange that Control Panel is an indicator light.

Comment #164: Event configuration window should show an example notification.

Comment #165: (See Comments #254, 257 and 276.)

Comment #166: (See Comments #253-263 and 267.)

Comment #167: User confused by Phase Three of the Exercises.

Comment #168: Incident report button doesn't work while query windows visible.

Comment #169: Interface cognizant of vertical screen real estate, but not horizontal.

Comment #170: User confused about how the indicators will change colors.

Comment #171: User would like to change the background color of indicator lights.

Comment #172: User concerned about default thresholds for weak connectivity.

Comment #173: User expects program definition to contain user's configuration files.

Comment #174: (See Comment #184.)

Comment #175: User would like a "Make New Directory" button in definition window.

Comment #176: User discovered that <Backspace> key doesn't work on telos.

Comment #177: User would like hoarded tasks to appear "<priority>: <NameOfTask>".

Comment #178: User suggests that system start the request as it asks for permission.

Comment #179: Expanding/Compressing node requires three clicks.

Comment #180: Tutorial is redundant.

Comment #181: User confused by extraneous underlining.

Comment #182: (See Comment #138.)

Comment #183: User brings up a task definition to get to a data definition.

Comment #184: User confused by "already exists" dialog from "Save As" button.

Comment #185: User concerned about difficulty of determining what is necessary.

Comment #186: User likes lights to get attention and gauges to show detail.

Comment #187: User wants different sets of settings ("at school" vs. "on the road").

Comment #188: Users confused by behavior of "Save As" button.

Comment #189: Distinction between general instructions and directives unclear.

Comment #190: User wants more control over program definitions.

Comment #191: Inconsistency between name of missing object and primary task.

Comment #192: User thinks there are too many indicator lights.

Comment #193: User suggests only making abnormal indicator lights visible.

Comment #194: User needs to go backwards in Tutorial but cannot.

Comment #195: User assumes a 24 hour clock (in Tokens window).

Comment #196: User expects ^U to work correctly.

Comment #197: User misunderstands description of one of the blizzard activities.

Comment #198: User enters just a single program in each definition.

Comment #199: User hoards based upon which tasks need the most resources.

Comment #200: User confused by "questionnaire already running".

Comment #201: User observes that the weak miss query doesn't look "special".

Comment #202: (See Comments #277-280.)

Comment #203: Task interface is overly complicated.

Comment #204: + and – signs have very small active area.

Comment #205: User would like graphical interface to see what's in the cache.

Comment #206: Event configuration should contain a copy of the indicator lights.

Comment #207: User does not associate beeping with color change.

Comment #208: No feedback given when window is already visible or is iconified.

Comment #209: User confused by term "meta-information".

Comment #210: User confused by term "program", suggests "programs".

Comment #211: User confused by term "subtask", suggests "task".

Comment #212: User concerned novices will be confused by term "RVM".

Comment #213: User suggests "Tokens Information", for "Authentication Information".

Comment #214: User confused by terms "all descendants", etc, suggests "recursive".

Comment #215: User confused by term "Control Panel", suggests "Alerts".

Comment #216: User concerned novices will be confused by term "Tokens".

Comment #217: User concerned about distinguishing a state from an event.

Comment #218: User concerned that Hoard Walk Advice shows "tasks", not "objects".

Comment #219: User confused by distinction between "hoarded" and "available".

Comment #220: User uses term "cache walk", rather than "hoard walk".

Comment #221: User confused by term "flash", suggests "blink".

Comment #222: User confused by term "subtasks", suggests "recursive".

Comment #223: User confused by term "predefined".

Comment #224: User believes term "advice" is being overused.

Comment #225: User unsure whether "bandwidth" refers to current or theoretical.

Comment #226: User confused by use of "advice indicator" to refer to indicator listbox.

Comment #227: User confused by term "advice".

Comment #228: User forgets the term "cache".

Comment #229: User confused by the term "weak".

Comment #230: User confused by the term "questionnaire".

Comment #231: User confused by the term "hoard walk".

Comment #232: User confused by the term "stop asking".

Comment #233: User confused as to what is "weak" – "my link" or "server link".

Comment #234: User confused by the term "critical".

Comment #235: User confused as to whether or not system has started the hoard.

Comment #236: User doesn't understand Hoard Walk Advice offers more details.

Comment #237: User confused by similarity between "Fetch?" and "Fetch" buttons.

Comment #238: User confused because he doesn't know bovik's password.

Comment #239: User confused because he doesn't know about reintegration.

Comment #240: User doesn't understand that urgency colors affect meter colors.

Comment #241: User wonders if system would repeatedly ask about the same file.

Comment #242: User confused about whether or not an unhoarded task will be cached.

Comment #243: User wants to see Venus' on-going activities somewhere (ala mariner).

Comment #244: User believes that program hoarding not emphasized enough.

Comment #245: User wonders who defines "substantial" [re: amount of time for fetch].

Comment #246: User doesn't believe system can automatically hoard programs.

Comment #247: User wonders what would happen if you hoard more than will fit.

Comment #248: User wonders for how long tokens are valid.

Comment #249: Help on the Hoard Walk Progress window is not helpful.

Comment #250: Network Help window doesn't explain why network is degraded.

Comment #251: Control Panel Help window (Event Configuration) is not helpful.

Comment #252: Task Help window doesn't explain why task is unavailable.

Comment #253: User does not understand Task Available notification.

Comment #254: User expects Task Unavailable to be a warning rather than critical.

Comment #255: User expects a notion of yellow for Tasks.

Comment #256: User expects disconnection notification based on volumes, not servers.

Comment #257: User believes Token Expiry should be critical, not warning.

Comment #258: User surprised that Hoard Walk indicator turns yellow during walk.

Comment #259: User doesn't understand why Task indicator is red.

Comment #260: User wants indicator to warn before tokens expire.

Comment #261: User does not understand Task Unavailable notification.

Comment #262: The colors mean too many different things.

Comment #263: User misinterprets the Hoard Walk Advice event as Task Unavailable.

Comment #264: User feels that the default Normal (green) color is too dark.

Comment #265: User feels that the default Warning (yellow) color is not bright enough.

Comment #266: User feels that the default Critical (red) color is not bright enough.

Comment #267: User does not understand Hoard Walk Advice request notification.

Comment #268: User confused by task meters in Task Information window.

Comment #269: User confused by cache meter in Task Information window.

Comment #270: User confused by the cache meter in the Space Information window.

Comment #271: User confused by the disk meter in the Space Information window.

Comment #272: User confused by the meters on the Network Information window.

Comment #273: User wants system to calculate how much additional space needed.

Comment #274: User makes up answers to exercises rather than using interface.

Comment #275: User doesn't understand question asking about event configurations.

Comment #276: User would prefer Token Expiry not beep.

Comment #277: Task Help Window helps user hoard and prioritize tasks.

Comment #278: Task Help Window helps user understand meters.

Comment #279: Task Help Window helps user unhoard task.

Comment #280: Space Help Window helps user understand meters.

Comment #281: User confused by description of meta-information.

Comment #282: User confuses <Enter> (<Return>) vs. <Enter> (numeric keypad).

Comment #283: User confused by use of "Fetch?" button.

Comment #284: User confused by description of Event Configuration.

Comment #285: User confused by "same colors as the walls".

Comment #286: User confused by "pull over 5 minutes ago or risk engine damage."

Comment #287: User doesn't understand "For the record" comment.

Comment #288: User confused by Task4 screen.

Comment #289: User doesn't understand phrase "bizarre form of drag-n-drop".

Comment #290: User doesn't understand phrase "Under the X indicator".

Comment #291: User doesn't understand when to click and when to double-click.

Comment #292: User doesn't understand how to open an indicator's window.

E.4.2 Summary of Spontaneous Comments

The following pages contain a summary of the 292 comments described above. For each comment, the spreadsheet indicates which users made it and during which segments of the study. The segments are coded as follows:

- *T*: Tutorial
- *E*: Exercises
- *D*: Debriefing[3]
- *F*: Tutorial and Exercises
- *L*: Exercises and Debriefing
- *M*: Tutorial and Debriefing
- *A*: Tutorial, Exercises, and Debriefing

In addition, I provide the total number of users (*Number Reporting*) making the comment (note that a user making the comment multiple times is counted just once). Finally, if a single user made a particular comment, I identify that user under the heading *Unique User*. Comment lines that are shaded in gray indicate a comment duplicated during the organization process and document the identification number of the comment that encompasses it.

[3] An asterisk indicates that the participant made the comment to the experimenter privately, after completing the study. I code it as part of the debriefing segment for simplicity.

	Pilots				Novices			Experts			Misc.			Number	Unique
ID	2	3	4	5	1	2	4	1	2	3	C	S	T	Reporting	User
1				T	T			M		A	T		T	6	
2	(See Comments #80 and 209-243)														
3				T	D	T		A	E	F		T		7	
4						T		F		E			T	4	
5					T	T		T	T	T			T	6	
6					F	F		A			E			4	
7			E					T		E				3	
8								T		F			F	3	
9	T		E	T	T		T	T	F	T	E		A	10	
10						T		T						2	
11								T						1	E1
12								T						1	E1
13								E	E	E	E	E		5	
14	(See Comments #249-252)														
15						E		T						2	
16	(See Comments #268-272)														
17	E	E	E	E	L	E		E		F		E		9	
18				E	E	E		E		E				5	
19								E						1	E1
20				F		E		E		E	D		E	6	
21			T					E	E				T	4	
22			E		E			L					T	4	
23				F	T			A		F	E	E		6	
24			T	T				L					E	4	
25										D*				1	E3
26								E						1	E1
27	(See Comment #261)														
28								E						1	E1
29				E		E		E		E				4	
30								E		E				2	
31		T						D						2	
32				T	F	T		D	F	E				6	
33		T				T		T	T	T			T	6	
34									T				T	2	
35									T		T			2	
36			F	E	E				T	E		T		6	
37		T	T						T					3	
38			T					T	T	F	T			5	
39									L					1	E2
40	(See Comment #41)														
41			E						L					2	
42						E			D	E			A	4	
43									D				E	2	
44										T				1	E3
45						T				T			T	3	
46		T								T				2	
47										T				1	E3
48	(See Comment #1)														
49	(See Comments #253-262)														
50							T	T		T				3	

	Pilots				Novices			Experts			Misc.			Number	Unique
ID	2	3	4	5	1	2	4	1	2	3	C	S	T	Reporting	User
51										T				1	E3
52										T				1	E3
53										T				1	E3
54					T					A				2	
55										F				1	E3
56										T				1	E3
57		F	F		F	E		E	E	F	D			8	
58										T	M			2	
59		T			F	F				F			A	5	
60						T				T				2	
61		T								T				2	
62						E				F				2	
63										T				1	E3
64			T							T				2	
65	(See Comment #253-262)														
66										T				1	E3
67										T				1	E3
68										T				1	E3
69	(See Comment #253-262)														
70				E						T				2	
71			T							M				2	
72										T				1	E3
73										T				1	E3
74										T				1	E3
75										T				1	E3
76										T			T	2	
77			E							T			T	3	
78										T				1	E3
79										T				1	E3
80			E	T						T				3	
81										E				1	E3
82		T		E		T				E				4	
83										E				1	E3
84										E			E	2	
85										E				1	E3
86										E				1	E3
87					T					L				2	
88										E				1	E3
89										D			E	2	
90										D				1	E3
91					T								T	2	
92					T	E					E			3	
93					T		T							2	
94			T	T	A								A	4	
95					T									1	N1
96					E		E							2	
97					E									1	N1
98						T								1	N2
99						E								1	N2
100						E	E					E		3	

	Pilots				Novices			Experts			Misc.			Number	Unique
ID	2	3	4	5	1	2	4	1	2	3	C	S	T	Reporting	User
101						E								1	N2
102						E	E	E	E				E	5	
103						E								1	N2
104						E							E	2	
105	(See Comments #264-266)														
106							T				T			2	
107							T				E			2	
108	E						T							2	
109	(See Comments #253 and #267)														
110								L						1	E1
111											T		T	2	
112	(See Comments #281-292)														
113										T	M			2	
114											T		T	2	
115							T				M			2	
116	(See Comments #235-237)														
117											T			1	C
118											T			1	C
119											E			1	C
120							E				E			2	
121											E		T	2	
122											L			1	C
123											L			1	C
124			T			E					E	T	F	5	
125											L			1	C
126											E			1	C
127				F							D		E	3	
128					E									1	N1
129													T	1	T
130													T	1	T
131													T	1	T
132													T	1	T
133													T	1	T
134													T	1	T
135													T	1	T
136													T	1	T
137			T									T	T	3	
138			E										T	2	
139											E		T	2	
140			E										T	2	
141													T	1	T
142			T										T	2	
143			T										T	2	
144													T	1	T
145													T	1	T
146													E	1	T
147				E		E					E	E	E	5	
148	(See Comments #238-248)														
149						E					E	E	E	4	
150													E	1	T

	Pilots				Novices			Experts			Misc.			Number	Unique
ID	2	3	4	5	1	2	4	1	2	3	C	S	T	Reporting	User
151													E	1	T
152	(See Comments #219, 274 and 275)														
153													D	1	T
154													D	1	T
155	E	E		E										3	
156	E													1	P2
157	E	E												2	
158	E													1	P2
159		T												1	P3
160		T												1	P3
161		T		E										2	
162		T								D				2	
163		T												1	P3
164		T												1	P3
165	(See Comments #254, 257 and 276)														
166	(See Comments #253-263, and 267)														
167		E												1	P3
168		E												1	P3
169			T											1	P4
170			T											1	P4
171			T											1	P4
172			T											1	P4
173			T											1	P4
174	(See Comment #184)														
175			T											1	P4
176			T											1	P4
177			T											1	P4
178			T											1	P4
179			T											1	P4
180			T											1	P4
181			E											1	P4
182	(See Comments #138)														
183			E	E										2	
184			F											1	P4
185			L					E					E	3	
186			D											1	P4
187			D											1	P4
188			D											1	P4
189				D										1	P5
190				D										1	P5
191						E								1	N2
192				D										1	P5
193				D										1	P5
194												T		1	S
195										E		E		2	
196												E		1	S
197												E		1	S
198												E		1	S
199												E		1	S
200												E		1	S

	Pilots				Novices			Experts			Misc.			Number	Unique
ID	2	3	4	5	1	2	4	1	2	3	C	S	T	Reporting	User
201		T												1	P3
202	(See Comments #277-280)														
203											D			1	C
204								T						1	E1
205										T				1	E3
206										F		T		2	
207						E								1	N2
208											E		F	2	
209								T		T				2	
210								T						1	E1
211						T			T		T		E	4	
212								T						1	E1
213								T						1	E1
214									T					1	E2
215										T				1	E3
216										T				1	E3
217										T				1	E3
218										E				1	E3
219		E			E				E					3	
220						E								1	N2
221	T													1	P2
222											T			1	C
223											T			1	C
224											M			1	C
225											T			1	C
226											T	T		2	
227											D			1	C
228					E									1	N1
229			T										T	2	
230													T	1	T
231			T											1	P4
232			T											1	P4
233			E											1	P4
234												E		1	S
235				E				T			T			3	
236		E										E		2	
237			T											1	P4
238											E			1	C
239		E		D		E		D		E	E	E	E	8	
240								T						1	E1
241								T						1	E1
242		T											E	2	
243													D	1	T
244													D	1	T
245		T												1	P3
246		T	T											2	
247			T											1	P4
248			T											1	P4
249						E				E				2	
250					E	E								2	

	Pilots				Novices			Experts			Misc.			Number	Unique
ID	2	3	4	5	1	2	4	1	2	3	C	S	T	Reporting	User
251													T	1	T
252					E									1	N1
253					E			T			E			3	
254											E			1	C
255								T		T				2	
256										T				1	E3
257			E			T				A				3	
258		T								T			T	3	
259										E				1	E3
260					T	T								2	
261		E				F	T	F	T				T	6	
262				T						E				2	
263									T					1	E2
264							T						T	2	
265			T											1	P4
266			T				T							2	
267													E	1	T
268		E		T		F		T			T		T	6	
269		E						T		T				3	
270								E						1	E1
271						T								1	N2
272														0	
273						E								1	N2
274				E										1	P5
275													L	1	T
276		T												1	P3
277											E			1	C
278								E						1	E1
279								E	E					2	
280						T								1	N2
281											T			1	C
282											T			1	C
283			T								T			2	
284											T			1	C
285								T					T	2	
286													T	1	T
287													T	1	T
288													T	1	T
289													T	1	T
290													T	1	T
291													E	1	T
292		T												1	P3
Total	8	31	52	29	29	44	14	52	24	81	51	22	74		
Unq (All)	3	9	23	5	5	10	0	14	3	31	19	7	27		
Initial Repor	8	29	47	17	16	15	4	26	8	33	19	12	37	271	

E.5 Summary of Results

In this section, you will find a summary of results found in the analysis just presented. The results are organized hierarchically by category. The order of the listed results within a given category is arbitrary.

Category	Page
General	394
Bugs	394
Cosmetic Problems	395
Enhancements	396
Help	396
Meters	397
Terminology	397
Widget Library	398
Windows	398
Indicator Lights	399
Notification of Events	399
Control Panel	400
General	400
Event Configuration	401
Urgency Colors	401
Tokens	402
Space	402
Network	402
Advice	403
Hoard Walk	403
Task	404
General	404
Task Information	405
Definitions	406
Study Materials	409
General	409
Tutorial	409
Exercises	411

E.5.1 General

Qualitative Result #6: Novice user thought the "first level interface" provided too little detail to determine what was going on.

Qualitative Result #17: Expert user finds graphical user interfaces tedious.

Comment #44: User dislikes font choice.

Comment #53: User finds the scrollbars ugly.

Comment #103: User expects system to track and automatically hoard missing objects.

Comment #110: User would like a tree view of Tasks.

Comment #131: User has difficulty finding scrollbar.

Comment #133: User wants cut & paste to work.

Comment #169: Interface cognizant of vertical screen real estate, but not horizontal.

Comment #176: User discovered that <Backspace> key doesn't work on telos.

Comment #205: User would like graphical interface to see what's in the cache.

Comment #243: User wants to see Venus' on-going activities somewhere (ala mariner).

Comment #273: User wants system to calculate how much additional space needed.

E.5.1.1 Bugs

E.5.1.1.1 Interface

Comment #4: Windows do not resize.

Comment #7: Interface alphabetizes caps and uncaps.

Comment #8: Meters shown at 1% have no visible color.

Comment #12: User expects <Enter> to make the help window disappear.

Comment #19: User loses data after selecting task from listbox.

Comment #22: System double-selects under certain circumstances.

Comment #30: Password visible on Authentication Information window.

Comment #43: Newly hoarded task may not be visible in list of hoarded tasks.

Comment #77: Number unreadable on meter.

Comment #79: Urgency labels on the Event Configuration Tab are not lined up.

Comment #82: Priorities need label.

Comment #85: End of task name is hidden behind meter.

Comment #89: Certain characters, used in task names, can crash the interface.

Comment #95: Space Help window did not update to show new information.

Comment #97: User "saves" definition under the name "New..." and it is lost.

Comment #99: User confused by bogus Fetch Statistics in Hoard Walk Advice window.

Comment #107: User types pathname, hits a key near <Return>, and it disappears.

Comment #120: Advice indicator goes green too soon.

Comment #127: User closes window without saving and loses data.

Comment #135: Visible shift when switching between definitions/configurations.

Comment #139: User experienced an unexplained loss of a definition.

Comment #176: User discovered that <Backspace> key doesn't work on telos.

Comment #249: Help on the Hoard Walk Progress window is not helpful.

Comment #251: Control Panel Help window (Event Configuration) is not helpful.

E.5.1.1.2 Widget Library

Comment #179: Expanding/Compressing node requires three clicks.

Comment #181: User confused by extraneous underlining.

E.5.1.1.3 Coda

Comment #104: Submitting a password takes excessively long.

E.5.1.2 Cosmetic Problems

E.5.1.2.1 Global

Comment #44: User dislikes font choice.

Comment #53: User finds the scrollbars ugly.

Comment #55: User doesn't recognize slider lines as widgets controlling screen layout.

Comment #75: Space help button is too small.

Comment #135: Visible shift when switching between definitions/configurations.

E.5.1.2.2 Local

Comment #54: User annoyed that a list with six elements must be scrolled.

Comment #70: User dislikes checkboxes, would prefer checkmarks.

Comment #73: Peer mistaken for child in tree widget.

Comment #79: Urgency labels on the Event Configuration Tab are not lined up.

Comment #91: User dislikes the Urgency Colors tab of the Control Panel.

Comment #142: User wants the triangle icons rather than the "+" and "-" icons.

Comment #143: User would prefer the indicator lights to be alphabetized.

Comment #144: User prefers klist to Tokens indicator.

Comment #171: User would like to change the background color of indicator lights.

Comment #177: User would like hoarded tasks to appear "<priority>: <NameOfTask>".

E.5.1.3 Enhancements

Comment #15: System should predict if hoarding a task will cause space problem.

Comment #25: User wants feedback regarding "health" of Venus.

Comment #57: User wants Data Definition window to indicate the size of subtree.

Comment #103: User expects system to track and automatically hoard missing objects.

Comment #110: User would like a tree view of Tasks.

Comment #169: Interface cognizant of vertical screen real estate, but not horizontal.

Comment #185: User concerned about difficulty of determining what is necessary.

Comment #205: User would like graphical interface to see what's in the cache.

Comment #243: User wants to see Venus' on-going activities somewhere (ala mariner).

Comment #273: User wants system to calculate how much additional space needed.

E.5.1.4 Help

Comment #12: User expects <Enter> to make the help window disappear.

Comment #13: Users confused when help window grabs and maintains focus.

Comment #45: User expects "help" on the indicator lights window.

Comment #75: Space help button is too small.

Comment #76: User wants expert help for space problems.

Comment #90: User would like "balloon" help.

Comment #95: Space Help window did not update to show new information.

Comment #249: Help on the Hoard Walk Progress window is not helpful.

Comment #250: Network Help window doesn't explain why network is degraded.

Comment #251: Control Panel Help window (Event Configuration) is not helpful.

Comment #252: Task Help window doesn't explain why task is unavailable.

Comment #277: Task Help Window helps user hoard and prioritize tasks.

Comment #278: Task Help Window helps user understand meters.

Comment #279: Task Help Window helps user unhoard task.

Comment #280: Space Help Window helps user understand meters.

E.5.1.5 Meters

Comment #8: Meters shown at 1% have no visible color.

Comment #50: Meters do not cover multiple orders of magnitude well.

Comment #51: Meters do not convey relative information.

Comment #77: Number unreadable on meter.

Comment #268: User confused by task meters in Task Information window.

Comment #269: User confused by cache meter in Task Information window.

Comment #270: User confused by the cache meter in the Space Information window.

Comment #271: User confused by the disk meter in the Space Information window.

Comment #272: User confused by the meters on the Network Information window.

E.5.1.6 Terminology

Comment #80: Distinction between different modes of operation unclear.

Comment #209: User confused by term "meta-information".

Comment #210: User confused by term "program", suggests "programs".

Comment #211: User confused by term "subtask", suggests "task".

Comment #212: User concerned novices will be confused by term "RVM".

Comment #213: User suggests "Tokens Information", for "Authentication Information".

Comment #214: User confused by terms "all descendants", etc, suggests "recursive".

Comment #215: User confused by term "Control Panel", suggests "Alerts".

Comment #216: User concerned novices will be confused by term "Tokens".

Comment #217: User concerned about distinguishing a state from an event.

Comment #218: User concerned that Hoard Walk Advice shows "tasks", not "objects".

Comment #219: User confused by distinction between "hoarded" and "available".

Comment #220: User uses term "cache walk", rather than "hoard walk".

Comment #221: User confused by term "flash", suggests "blink".

Comment #222: User confused by term "subtasks", suggests "recursive".

Comment #223: User confused by term "predefined".

Comment #224: User believes term "advice" is being overused.

Comment #225: User unsure whether "bandwidth" refers to current or theoretical.

Comment #226: User confused by use of "advice indicator" to refer to indicator listbox.

Comment #227: User confused by term "advice".

Comment #228: User forgets the term "cache".

Comment #229: User confused by the term "weak".

Comment #230: User confused by the term "questionnaire".

Comment #231: User confused by the term "hoard walk".

Comment #232: User confused by the term "stop asking".

Comment #233: User confused as to what is "weak" – "my link" or "server link".

Comment #234: User confused by the term "critical".

E.5.1.7 Widget Library

Comment #53: User finds the scrollbars ugly.

Comment #55: User doesn't recognize slider lines as widgets controlling screen layout.

Comment #73: Peer mistaken for child in tree widget.

Comment #142: User wants the triangle icons rather than the "+" and "-" icons.

Comment #179: Expanding/Compressing node requires three clicks.

Comment #181: User confused by extraneous underlining.

Comment #204: + and – signs have very small active area.

E.5.1.8 Windows

Comment #1: Participants frustrated by window placement.

Comment #4: Windows do not resize.

Comment #24: User experiences window overload.

Comment #135: Visible shift when switching between definitions/configurations.

Comment #208: No feedback given when window is already visible or is iconified.

E.5.2 Indicator Lights

Qualitative Result #4: Expert user was confused by the distinction between the Hoard Walk indicator and the Task indicator.

Comment #10: User confused that two indicators open the same advice request.

Comment #11: Users confused by different windows appearing from one indicator light.

Comment #25: User wants feedback regarding "health" of Venus.

Comment #31: User expresses confusion between Hoard Walk and Task indicators.

Comment #46: User surprised that a double-click is required to open indicator window.

Comment #143: User would prefer the indicator lights to be alphabetized.

Comment #159: User surprised that indicator lights are dynamic.

Comment #163: User thinks it strange that Control Panel is an indicator light.

Comment #169: Interface cognizant of vertical screen real estate, but not horizontal.

Comment #186: User likes lights to get attention and gauges to show detail.

Comment #192: User thinks there are too many indicator lights.

Comment #193: User suggests only making abnormal indicator lights visible.

E.5.2.1 Notification of Events

E.5.2.1.1 General

Qualitative Result #16: Expert user afraid of missing an event notification.

Comment #9: User confused by two-color blinking.

Comment #33: User confused because indicator stops blinking.

Comment #41: User concerned that he might miss an indicator state change.

Comment #118: User confused as to what exactly is going to "flash".

Comment #121: User disturbed by window popping up.

Comment #170: User confused about how the indicators will change colors.

Comment #171: User would like to change the background color of indicator lights.

Comment #207: User does not associate beeping with color change.

Comment #262: The colors mean too many different things.

E.5.2.1.2 Specific

Tutorial Result #1: (Screen 16) Two users had difficulty with the "Operating Disconnected" event.

Tutorial Result #9: (Screen 36) Two users had difficulty prioritizing tasks or understanding the "Task Unavailable" event notification.

Comment #88: User would prefer more obvious notification of strong connectivity.

Comment #253: User does not understand Task Available notification.

Comment #254: User expects Task Unavailable to be a warning rather than critical.

Comment #255: User expects a notion of yellow for Tasks.

Comment #256: User expects disconnection notification based on volumes, not servers.

Comment #257: User believes Token Expiry should be critical, not warning.

Comment #258: User surprised that Hoard Walk indicator turns yellow during walk.

Comment #259: User doesn't understand why Task indicator is red.

Comment #260: User wants indicator to warn before tokens expire.

Comment #261: User does not understand Task Unavailable notification.

Comment #263: User misinterprets the Hoard Walk Advice event as Task Unavailable.

Comment #267: User does not understand Hoard Walk Advice request notification.

Comment #276: User would prefer Token Expiry not beep.

E.5.3 Control Panel

E.5.3.1 General

Comment #54: User annoyed that a list with six elements must be scrolled.

Comment #70: User dislikes checkboxes, would prefer checkmarks.

E.5.3.1.1 Commit Behavior

Comment #32: User expects "commit" to close Control Panel window (as feedback).

Comment #39: User confused about extent of "commit" button's action.

Comment #81: User surprised he must commit changes to the event configurations.

Comment #154: User prefers for close without saving/committing to abort.

E.5.3.2 Event Configuration

Tutorial Result #12: (Screen 59) Five users had difficulty examining event configurations.

Tutorial Result #13: (Screen 60) Two expert users had difficulty confirming their answer regarding event configuration.

Exercise Result #5: Two users found the events to be configured inefficiently.

Qualitative Result #5: Users had difficulty finding events in the Event Configuration tab of the Control Panel.

Comment #17: User guesses location of event in indicator list.

Comment #38: User confused about how system notifies if "everything" is turned off.

Comment #79: Urgency labels on the Event Configuration Tab are not lined up.

Comment #80: Distinction between different modes of operation unclear.

Comment #87: User wants to modify all event notifications at once.

Comment #119: User does not recognize Event Configuration window.

Comment #135: Visible shift when switching between definitions/configurations.

Comment #145: User would like to have a set of system defaults.

Comment #147: User searches for way to configure events.

Comment #160: User would prefer the Urgency of Event Configurations to be in color.

Comment #161: User would prefer configuration options be checkboxes (not listboxes).

Comment #162: User suggests improved layout for the event configuration tab.

Comment #164: Event configuration window should show an example notification.

Comment #170: User confused about how the indicators will change colors.

Comment #187: User wants different sets of settings ("at school" vs. "on the road").

Comment #206: Event configuration should contain a copy of the indicator lights.

E.5.3.3 Urgency Colors

Comment #91: User dislikes the Urgency Colors tab of the Control Panel.

Comment #146: User would like to drag colors.

Comment #171: User would like to change the background color of indicator lights.

Comment #264: User feels that the default Normal (green) color is too dark.

Comment #265: User feels that the default Warning (yellow) color is not bright enough.

Comment #266: User feels that the default Critical (red) color is not bright enough.

E.5.4 Tokens

Comment #30: Password visible on Authentication Information window.

Comment #104: Submitting a password takes excessively long.

Comment #144: User prefers klist to Tokens indicator.

Comment #195: User assumes a 24 hour clock (in Tokens window).

E.5.5 Space

Exercise Result #2: Three users could not accurately describe the current space status.

Exercise Result #10: Four users described "RVM Filling" event inefficiently.

Comment #50: Meters do not cover multiple orders of magnitude well.

Comment #76: User wants expert help for space problems.

Comment #95: Space Help window did not update to show new information.

Comment #270: User confused by the cache meter in the Space Information window.

Comment #271: User confused by the disk meter in the Space Information window.

Comment #280: Space Help Window helps user understand meters.

E.5.6 Network

Tutorial Result #1: (Screen 16) Two users had difficulty with the "Operating Disconnected" event.

Exercise Result #13: Four users described "Operating Disconnected" event inefficiently.

Comment #50: Meters do not cover multiple orders of magnitude well.

Comment #51: Meters do not convey relative information.

Comment #129: User wants interface to show what files are located on what servers.

Comment #172: User concerned about default thresholds for weak connectivity.

Comment #256: User expects disconnection notification based on volumes, not servers.

Comment #272: User confused by the meters on the Network Information window.

E.5.7 Advice

Tutorial Result #10: (Screen 40) Four users experienced difficulty orienting to the "Advice Information" window.

Exercise Result #12: Three users provided "Weak Miss" advice inefficiently.

Qualitative Result #18: Expert user would like to have the ability to stop long transfers over weak network connections.

Comment #28: User confused about what object needs to be fetched.

Comment #66: Requests needing immediate attention require too many steps.

Comment #67: User concern that 15 seconds is wrong default timeout.

Comment #68: User would like a count-down timer on advice requests.

Comment #98: User confused by options in the weak miss timeout prod.

Comment #103: User expects system to track and automatically hoard missing objects.

Comment #115: User thinks it strange that we have both kinds of advice under Advice.

Comment #120: Advice indicator goes green too soon.

Comment #150: User wants to be able to say "0" to Disconnected Miss Query.

Comment #178: User suggests that system start the request as it asks for permission.

Comment #200: User confused by "questionnaire already running".

Comment #201: User observes that the weak miss query doesn't look "special".

E.5.8 Hoard Walk

Tutorial Result #11: (Screen 49) Three users experienced difficulty completing hoard walk advice request.

Exercise Result #1: Five users provided incorrect hoard walk advice.

Exercise Result #4: Five users provided hoard walk advice inefficiently.

Exercise Result #9: Two users requested hoard walk inefficiently.

Qualitative Result #13: Novice user dislikes having fetch time expressed in seconds.

Comment #70: User dislikes checkboxes, would prefer checkmarks.

Comment #71: User dislikes "All Tasks Needing Data" pseudo-task.

Comment #73: Peer mistaken for child in tree widget.

Comment #96: Seconds too fine-grained.

Comment #99: User confused by bogus Fetch Statistics in Hoard Walk Advice window.

Comment #140: User confused by the "unselect" dialog box.

Comment #141: User confused by "Fetch" and "Stop Asking" toggling one another.

Comment #142: User wants the triangle icons rather than the "+" and "-" icons.

Comment #155: User doesn't recognize Hoard Walk Advice as a request for advice.

Comment #156: User confused by Hoard Walk Advice coming in under Hoard Walk.

Comment #179: Expanding/Compressing node requires three clicks.

Comment #199: User hoards based upon which tasks need the most resources.

Comment #204: + and – signs have very small active area.

Comment #235: User confused as to whether or not system has started the hoard.

Comment #236: User doesn't understand Hoard Walk Advice offers more details.

Comment #237: User confused by similarity between "Fetch?" and "Fetch" buttons.

E.5.9 Task

E.5.9.1 General

Comment #7: Interface alphabetizes caps and uncaps.

Comment #29: User confused by file becoming unavailable.

Comment #54: User annoyed that a list with six elements must be scrolled.

Comment #55: User doesn't recognize slider lines as widgets controlling screen layout.

Comment #92: User wants "undo".

Comment #93: User curious how the system copes with links.

Comment #126: User unsure which window allows him to prioritize tasks.

Comment #183: User brings up a task definition to get to a data definition.

Comment #185: User concerned about difficulty of determining what is necessary.

Comment #196: User expects ^U to work correctly.

Comment #203: Task interface is overly complicated.

Comment #255: User expects a notion of yellow for Tasks.

E.5.9.1.1 Lost Data

Comment #19: User loses data after selecting task from listbox.

Comment #97: User "saves" definition under the name "New..." and it is lost.

Comment #107: User types pathname, hits a key near <Return>, and it disappears.

Comment #127: User closes window without saving and loses data.

Comment #139: User experienced an unexplained loss of a definition.

E.5.9.1.2 Save Behavior

Qualitative Result #14: Expert user confused by behavior of "Save" buttons, in particular whether all tasks are saved at once.

Comment #59: User expected "save" button to also close the window.

Comment #60: User unsure as to why there is a "save as" button.

Comment #154: User prefers for close without saving/committing to abort.

Comment #158: User surprised that system saves definition automatically upon close.

Comment #184: User confused by "already exists" dialog from "Save As" button.

Comment #188: Users confused by behavior of "Save As" button.

E.5.9.2 Task Information

Tutorial Result #9: (Screen 36) Two users had difficulty prioritizing tasks or understanding the "Task Unavailable" event notification.

Exercise Result #8: Three users prioritized tasks inefficiently.

Exercise Result #11: Three users described "Task Unavailable" event inefficiently.

Exercise Result #14: Two users reorganized task to hoard subtask at a different priority.

Qualitative Result #10: Expert user unsure whether or not a program was available.

Comment #15: System should predict if hoarding a task will cause space problem.

Comment #18: Users have difficulty finding definitions in predefined lists.

Comment #26: User has difficulty finding task in Task Information window.

Comment #42: User has difficulty prioritizing tasks.

Comment #43: Newly hoarded task may not be visible in list of hoarded tasks.

Comment #63: User confused about how to select tasks to be hoarded.

Comment #64: User would like to drag tasks within the list of hoarded tasks.

Comment #82: Priorities need label.

Comment #85: End of task name is hidden behind meter.

Comment #89: Certain characters, used in task names, can crash the interface.

Comment #102: Users have difficulty unhoarding tasks.

Comment #110: User would like a tree view of Tasks.

Comment #114: User thinks it strange that we have a meter on the Tasks.

Comment #138: User has difficulty deleting a definition.

Comment #157: User thinks it silly to have to create a task for a single directory.

Comment #177: User would like hoarded tasks to appear "<priority>: <NameOfTask>".

Comment #268: User confused by task meters in Task Information window.

Comment #269: User confused by cache meter in Task Information window.

Comment #273: User wants system to calculate how much additional space needed.

Comment #277: Task Help Window helps user hoard and prioritize tasks.

Comment #278: Task Help Window helps user understand meters.

Comment #279: Task Help Window helps user unhoard task.

E.5.9.3 Definitions

Exercise Result #3: Five users failed to completely define the "fixing bugs" task.

Exercise Result #6: Three users defined "Header Files" task inefficiently.

Exercise Result #7: Four users defined "CV" task inefficiently.

Qualitative Result #8: Expert user finds he must explicitly think about how to structure his task definitions.

Qualitative Result #12: Novice user accidentally deletes entire task rather than single element.

Qualitative Result #14: Expert user confused by behavior of "Save" buttons, in particular whether all tasks are saved at once.

Qualitative Result #15: Expert user confused by creating a new task vs. changing the name of an old one.

Comment #5: Participants confused that changing name and saving creates a new defn.

Comment #19: User loses data after selecting task from listbox.

Comment #136: User confused by having to hit <Enter> after typing name of definition.

Comment #138: User has difficulty deleting a definition.

E.5.9.3.1 Task Definition

Tutorial Result #2: (Screen 20) Two novice users experienced difficulty orienting to the "Task Definition" window.

Tutorial Result #4: (Screen 22) Three users had difficulty finding the "writing thesis" task in the "Task Definition" window (under either the "predefined list" or the "contains" list).

Tutorial Result #8: (Screen 34) Three users had difficulty modifying a task definition.

Exercise Result #14: Two users reorganized task to hoard subtask at a different priority.

Comment #6: Participant expects a button to delete an element from the "contains" list.

Comment #18: Users have difficulty finding definitions in predefined lists.

Comment #22: System double-selects under certain circumstances.

Comment #23: User finds it strange that a single-click selects an item for "contains" list.

Comment #62: User has difficulty deselecting a chosen element from "contains" list.

Comment #122: User double-clicks in empty contains list and "New..." pops up.

Comment #123: User dislikes use of predefined and contains lists.

Comment #135: Visible shift when switching between definitions/configurations.

Comment #181: User confused by extraneous underlining.

E.5.9.3.2 Data Definition

Tutorial Result #5: (Screen 23) Three users experienced difficulty orienting to the "Data Definition" window.

Tutorial Result #6: (Screen 27) Five users had difficulty entering data into the "Data Definition" window.

Tutorial Result #7: (Screen 28) Three users had difficulty closing "Data Definition" window.

Qualitative Result #2: Expert users had difficulty understanding distinction between <Tab> and <Enter>.

Qualitative Result #7: Expert user confused by the "no such file" indication.

Qualitative Result #9: Expert user finds editing file/directory lists (in "Data Definition" windows) tricky

Qualitative Result #10: Expert user unsure whether or not a program was available.

Qualitative Result #11: Novice user dislikes having to enter pathnames.

Comment #3: Participants have difficulty entering data in Data Definition window.

Comment #20: User wants to click on disabled meta-information.

Comment #21: User concerned that entire pathname is not visible.

Comment #36: User confused by lack of feedback when directory does not exist.

Comment #37: User bothered by addition of empty entry widget.

Comment #56: User dissatisfied with Data Definition window.

Comment #57: User wants Data Definition window to indicate the size of subtree.

Comment #58: User dislikes default meta-information for directories.

Comment #94: User would prefer not to type in pathnames.

Comment #101: User expects arrow keys to work in the meta-information area.

Comment #113: User concerned that he would use "all descendants", "all the time."

Comment #132: User expects <Back-Tab> to work in meta-information.

Comment #175: User would like a "Make New Directory" button in definition window.

Comment #196: User expects ^U to work correctly.

E.5.9.3.3 Program Definition

Qualitative Result #10: Expert user unsure whether or not a program was available.

Qualitative Result #11: Novice user dislikes having to enter pathnames.

Comment #21: User concerned that entire pathname is not visible.

Comment #37: User bothered by addition of empty entry widget.

Comment #61: User doesn't believe system ensures program executables exist.

Comment #94: User would prefer not to type in pathnames.

Comment #107: User types pathname, hits a key near <Return>, and it disappears.

Comment #125: User expects meta-information for programs.

Comment #137: User types a pathname into the name of the definition.

Comment #153: User would like to be able to see the details of program definitions.

Comment #173: User expects program definition to contain user's configuration files.

Comment #190: User wants more control over program definitions.

Comment #196: User expects ^U to work correctly.

Comment #198: User enters just a single program in each definition.

E.5.10 Study Materials

E.5.10.1 General

Comment #1: Participants frustrated by window placement.

Comment #84: Color used to highlight in Exercises window bad with pink background.

Comment #124: User failed to read instructions completely.

E.5.10.2 Tutorial

Qualitative Result #3: Expert users thought the Tutorial went into too much detail.

Comment #47: Screen of tutorial partially covers a window of the interface.

Comment #78: User dislikes directive to not use "Destroy Tokens" and "New Tokens".

Comment #117: User believes the Tutorial to be out-of-sequence.

Comment #130: User would like to be able to highlight in tutorial (as a bookmark).

Comment #134: User disturbed by instruction to "close all windows" (only one open).

Comment #180: Tutorial is redundant.

Comment #194: User needs to go backwards in Tutorial but cannot.

E.5.10.2.1 Clarifications

Tutorial Result #3: (Screen 21) Two novice users had difficulty understanding explanation of the different ways to view task definitions.

Comment #34: User has difficulty finding "black down-arrow button".

Comment #35: Users think they should literally make up pathnames.

Comment #108: User looks under indicator light rather than indicator listbox.

Comment #111: User has difficulty identifying the indicator lights.

Comment #118: User confused as to what exactly is going to "flash".

Comment #226: User confused by use of "advice indicator" to refer to indicator listbox.

Comment #237: User confused by similarity between "Fetch?" and "Fetch" buttons.

Comment #281: User confused by description of meta-information.

Comment #282: User confuses <Enter> (<Return>) vs. <Enter> (numeric keypad).

Comment #283: User confused by use of "Fetch?" button.

Comment #284: User confused by description of Event Configuration.

Comment #285: User confused by "same colors as the walls".

Comment #286: User confused by "pull over 5 minutes ago or risk engine damage."

Comment #287: User doesn't understand "For the record" comment.

Comment #288: User confused by Task4 screen.

Comment #289: User doesn't understand phrase "bizarre form of drag-n-drop".

Comment #290: User doesn't understand phrase "Under the X indicator".

Comment #291: User doesn't understand when to click and when to double-click.

Comment #292: User doesn't understand how to open an indicator's window.

E.5.10.2.2 Bugs

Comment #52: Nothing is hoarded, yet 70% of cache is full.

Comment #86: User does not believe the Space meter (cache).

Comment #106: Tutorial assumes green/yellow/red.

Comment #189: Distinction between general instructions and directives unclear.

Comment #194: User needs to go backwards in Tutorial but cannot.

E.5.10.2.3 Omissions

Qualitative Result #1: Users did not appreciate question about reintegration after it had not been covered in the Tutorial.

Comment #38: User confused about how system notifies if "everything" is turned off.

Comment #80: Distinction between different modes of operation unclear.

Comment #238: User confused because he doesn't know bovik's password.

Comment #239: User confused because he doesn't know about reintegration.

Comment #240: User doesn't understand that urgency colors affect meter colors.

Comment #241: User wonders if system would repeatedly ask about the same file.

Comment #242: User confused about whether or not an unhoarded task will be cached.

Comment #244: User believes that program hoarding not emphasized enough.

Comment #245: User wonders who defines "substantial" [re: amount of time for fetch].

Comment #246: User doesn't believe system can automatically hoard programs.

Comment #247: User wonders what would happen if you hoard more than will fit.

Comment #248: User wonders for how long tokens are valid.

E.5.10.3 Exercises

Comment #149: User has forgotten primary "task".

Comment #151: User upset that Exercises tell him that he's done when he isn't.

Comment #274: User makes up answers to exercises rather than using interface.

E.5.10.3.1 Clarifications

Comment #100: User confused by instruction to define header files as a program.

Comment #167: User confused by Phase Three of the Exercises.

Comment #197: User misunderstands description of one of the blizzard activities.

Comment #275: User doesn't understand question asking about event configurations.

E.5.10.3.2 Bugs

Comment #74: Task indicator turns green before hoard walk completes.

Comment #86: User does not believe the Space meter (cache).

Comment #128: User notices omissions in the mock task definitions.

Comment #168: Incident report button doesn't work while query windows visible.

E.5.10.3.3 Inconsistencies

Comment #72: User notices an upper-case/lower-case inconsistency.

Comment #83: User notices inconsistent task name between exercises and interface.

Comment #191: Inconsistency between name of missing object and primary task.

F Usability Findings

This appendix presents the results of the usability study. The findings for which I present concrete recommendations are organized according to *scope* and *severity* [10], and presented in the first four sections of this appendix. I begin this appendix with a brief overview of scope and severity for those unfamiliar with the terms.

The scope of a problem indicates how widespread that problem is. A problem with *local* scope is limited to one or two screens while a problem with *global* scope applies to multiple screens. A problem with global scope is more critical than one with local scope.

The severity of a problem indicates how critical the problem is. Severity is measured on a scale of 1 to 4, with 1 being the most severe. The four levels are described by Dumas and Redish and summarized below:

Level 1: prevent completion of a task

Level 2: create significant delay and frustration

Level 3: have minor effects on usability

Level 4: represent subtle problems and may point to potential enhancements

Each finding described in this appendix is supported by some sort of evidence. I use five forms of evidence:

Heuristic Violation: the interface violates a known heuristic[4]

Tutorial Result: users experienced a problem during the Tutorial

Exercise Result: users experienced a problem during the Exercises

Qualitative Result: users commented in the Evaluation Questionnaire

Comment IDs: users commented in their verbal protocol

Heuristic violations refer to places where the interface violates one of the heuristics described in Section 3.3 of the thesis. The remaining pieces of evidence are identified by their identification numbers and described in Appendix E.

One or more forms of evidence support each finding documented in this appendix. In general, each finding is supported by two different forms of evidence or by three or more users. Any piece of evidence deemed particularly important is also documented. Those findings with only minimal objective evidence, but supported subjectively, are identified with a ◆ in the left margin.

[4] The identification of heuristic violations was made in retrospect. A heuristic analysis may not identify the same problems.

F.1 Level 1

A *level 1* usability problem prevents users from completing a task.

F.1.1 Global

Finding #1: Users confused by event notification
Evidence:
Heuristic Violation: NA
Tutorial Results: 1, 9
Exercise Results: 10, 11, 12, 13
Qualitative Results: NA
Comment IDs: 207, 253, 258, 259, 261, 263, 267
Explanation: At various times, users became confused about the cause of an event notification. Users simply did not understand what had happened or did not associate beeping with an event notification. During the Exercises, users occasionally required more than two (and sometimes more than three) times par to report certain events.
Recommendations: Although users were not frequently confused, the fact that they were confused is troublesome. User confusion was not limited to the duration of the Tutorial so this is not simply a learning curve issue. It is important to note that user inefficiency in reporting certain event notifications could be due to poor timing during that phase of the Exercises. However, these symptoms suggest the need for a mechanism through which users can be given a written explanation of the most recent event, a sort of log or history of events. The Control Panel seems to be the obvious location for such a listing.

F.1.2 Local

Finding #2: Some help windows aren't helpful.[5]
Evidence:
Heuristic Violation: Help
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 80, 95, 249, 251
Explanation: The help window (and informational text on the Event Configuration tab of the Control Panel) does not explain read disconnected mode. The help window associated with the Space Information indicator does not update if the space status changes while it is visible. The help window associated with the progress view of the Hoard Walk indicator was never written. The help window associated with the Event Configuration tab of the Control Panel was never updated to reflect the revised layout.
Recommendations: These help windows need to be written (or rewritten) to be consistent with the windows they describe. The Space help window needs to update when the space status changes.

[5] This finding is categorized as local because it represents four individual findings each on a single window rather than a finding about all help windows in the interface.

Finding #3: Users fail to include needed programs in definition.
Evidence:

Heuristic Violation: Memory Overload
Tutorial Results: NA
Exercise Results: 3
Qualitative Results: NA
Comment IDs: 185

Explanation: Users failed to realize that one of their task definitions required the use of the `uncompress` program because some of the data included in that definition was compressed. Two users failed to hoard the editing task as part of one of their task definitions or even as a top-level task.
Recommendations: The interface could track what programs access data contained in task definitions and then give users hints regarding any programs omitted from a task definition.

Finding #4: Event Configuration Tab of Control Panel layout inappropriate to task.
Evidence:

Heuristic Violation: Simple & Natural Dialogue
Tutorial Results: 12
Exercise Results: 5
Qualitative Results: 5
Comment IDs: 17, 162, 206

Explanation: In order for users to modify the configuration of a particular event, they must first know under what indicator that event is notified. For instance, to change the configuration of the "Task Unavailable" event, users must first select the "Task" indicator from the indicator listbox of the Event Configuration tab of the Control Panel. Then, they can select the "Task Unavailable" event from the event listbox. In this example, the decision to look under the "Task" indicator is relatively straightforward. However, if the event is "Weakly Connected Cache Miss Advice", users could reasonably look under "Network", "Space", or "Advice". Many did so. One user (S) completed an exercise incorrectly because she found the "Operating Weakly Connected" event under the "Network" indicator and assumed that to be the correct event. Other users were resorted to using brute-force search.
Recommendations: The Event Configuration tab was redesigned during pilot testing. The original design listed all events under a single listbox. One of the early pilot users pointed out that the Event Configuration should include the indicator on which an event will be notified. The remaining users in the study used the redesigned Event Configuration tab. Unfortunately, the redesign introduced the dependency between the indicator and the event. By the end of the pilot testing, it was clear that this was a problem, but the experimenter was reluctant to make further changes so late in pilot testing. The Event Configuration tab needs to be redesigned to address the original suggestion while eliminating the need to know event/indicator pairings. The recommendation is to consider a design that contains a single list of events and a copy of the indicator lights.

Finding #5: Certain characters, if used in task names, can crash the interface
Evidence:

Heuristic Violation: Speak the user's language
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 89

Explanation: If the user happens to use certain characters in the names of tasks, the interface will crash when that task is hoarded. The character in question is one of the "upper-case numbers". We need to explore the full extent of the problem to determine how many characters are involved.
Recommendations: This is a bug. Any (printable) character should be valid.

- **Finding #6:** Password visible on the Authentication Information window.

  Evidence:

  Heuristic Violation: NA
  Tutorial Results: NA
  Exercise Results: NA
  Qualitative Results: NA
  Comment IDs: 30

  Explanation: As the user types into entry widgets, the widget library automatically echoes the characters that are typed to the screen. Although such behavior is appropriate for most entry widgets, it is inappropriate for one whose purpose is to collect the user's password. To avoid displaying the user's password, the interface echoes the password in the same color as the background so that the user is unable to discern the characters of the password. Unfortunately, the foreground color of the text and the background color of the entry widgets were not identical and the user could, with careful examination, read the password.

  Recommendations: This problem has been corrected in the current system by making the foreground and background colors identical. Users can no longer read the password echoed to the screen. Other solutions to the problem would be to echo a '*' for each character typed into the password entry widget or to disable character echoing.

Finding #7: Users do not recognize the hoard walk advice request as a request for advice

Evidence:

Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 155

Explanation: Three pilot users failed to recognize the hoard walk advice request as a request for advice. Two of these three users were running on an earlier version of the interface in which the hoard walk advice request came in only under the Hoard Walk indicator light. The confusion was apparently caused because they found it strange that an advice request didn't come in on the Advice indicator light. The third user, however, was running on a revised version.

Recommendations: Because two of the users involved were using an earlier version of the interface and because later versions of the interface address this problem, their confusion does not apply to the current system. The fact that a user running the revised system also did not recognize the advice request is cause for concern. However, this user made it through the Tutorial in record time, clearly skipping portions of the instructions. For this reason, even though his failure to recognize the request for what it was is a cause for concern, I do not recommend any action be taken unless further symptoms arise.

Finding #8: User does not recognize that the Hoard Walk Advice offers more details

Evidence:

Heuristic Violation: Simple & Natural
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 236

Explanation: Two users apparently didn't recognize that the Hoard Walk Advice Request offers more details. One pilot user did not recognize the window as a request for advice (see Finding #7 above) and never expanded the "All Tasks Needing Data" pseudo-task. The second user expanded the "All Tasks Needing Data" pseudo-task, but did not explore further.

Recommendations: Although the pilot user's behavior can be explained as symptomatic of the fact that he didn't recognize the window as being a request for hoard walk advice, the other user's behavior is disturbing. Inadequate training may be partially to be blame however. The Tutorial encourages users to explore the details of the hoard walk advice request, but does not actually lead them through the process step-by-step nor does it give users reasons for selecting some items to be fetched but not others.

Furthermore, the Help window does not offer suggestions for why to select some items over others. Both of these deficiencies should be addressed. Finally, the interface should show the "All Tasks Needing Data" pseudo-task in an already expanded format so that the relevant information is available immediately.

◆ **Finding #9:** Indicator changes to Normal urgency level before appropriate to do so
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 120
Explanation: The Advice indicator change to the Normal urgency level at the point at which users double-clicked on a request for advice but before they had completed the request.
Recommendations: This bug must be corrected since users may incorrectly assume that the request for advice has been completed.

Finding #10: Users unsure whether hoard walk is proceeding during request for advice
Evidence:
Heuristic Violation: Feedback
Tutorial Results: NA
Exercise Results: 1
Qualitative Results: NA
Comment IDs: 235
Explanation: Three users commented, as they were completing the hoard walk advice request, they were unsure whether or not the hoard daemon was working on the hoard walk. One user failed to complete the hoard walk advice request and thus start the actual walk. Because she was not one of the users making this comment, it is unclear whether or not her failure to complete the request was related to confusion over whether or not hoard walk was already in progress. However, it is interesting to note that her behavior was consistent with the mistaken impression that the hoard walk was progressing as the user offered advice.
Recommendations: One possible way to eliminate this confusion is to have the hoard walk advice request come in under the Advice indicator only. The Hoard Walk indicator would change to red to indicate that it is stalled while it awaits advice. This change would make the situation more clear to users.

◆ **Finding #11:** User prioritized tasks inappropriately
Evidence:
Heuristic Violation: Help
Tutorial Results: NA
Exercise Results: 3 (a subset of the users)
Qualitative Results: NA
Comment IDs: 199
Explanation: One user prioritized tasks based not upon what tasks were most important to her work, but upon what tasks required the most resources. Two users failed to include an editing task as a subtask of their primary tasks and then failed to give it a high priority.
Recommendations: Neither the Tutorial nor the Help window explains that higher priorities should be assigned to those tasks most important to the user's work. Users were forced to make this assumption. The Tutorial and the Help window should both make it clear how priorities should be chosen. Once this omission has been corrected, we must wait to see if the problem resurfaces, perhaps by observing a few new users perform selected exercises informally.

F.2 Level 2

A *level 2* usability problem causes users significant delay or frustration.

F.2.1 Global

Finding #12: Users want undo facility.
Evidence:
Heuristic Violation: Clearly marked exits
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: 12
Comment IDs: 6, 92
Explanation: Users expressed desire to undo their actions. Users accidentally deleted an entire task rather than a single element of the task. Often, they had no way to recover from their misstep.
Recommendations: The interface needs to have the ability to undo actions. In the situations observed during the user study, a single-level undo would have sufficed.

Finding #13: Users find the behavior of "Save" and "Commit" confusing
Evidence:
Heuristic Violation: Feedback
Consistency (external)
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: 14
Comment IDs: 32, 39, 59, 60, 154, 158, 184, 188
Explanation: The design of the interface follows suggestions for maintaining a "unified file model" [8]. In particular, the interface

- automatically saves definitions when the user closes the definition window
- does not close a window after the user saves a definition
- does not pop-up the SAVE dialog when the user closes the definition
- allows the user to abandon all changes since opening or creating the definition
- provides an optional, manual SAVE command

The suggestions above encompass just those portions of Cooper's complete set of recommendations that seemed to apply to a simple "document" like a definition. Those recommendations not implemented include the ability to snapshot, milestone, rename, place & reposition, specify the stored format, and undo specific changes. Cooper's recommendations appeal to me because they step back and look at the functionality required of an interface from the perspective of the user rather than from the perspective of the implementation of the interface.
Recommendations: The first recommendation would be to pick a model and be consistent. If the decision is to go with Cooper's unified model, then the implementation must be modified in the following ways:

- add snapshot and milestone capabilities
- remove SAVE AS capability
- replace in-place naming with in-place renaming
- add a specialized undo facility (as discussed in Finding #12 above)

The interface does not yet need the ability to place & reposition definitions or specify the stored definition format because it manipulates such a specialized and small class of documents.

The second recommendation is to add feedback regarding the save status of definitions. Note that according to Cooper's recommendations, that feedback should not be to close the window (as five of our users expected). Perhaps the emacs * would be appropriate. A meter will probably not work well because the interface saves definitions too quickly for users to notice and understand the meter before it disappears.

Finding #14: Users are frustrated by the placement of windows on the screen.
Evidence:

Heuristic Violation: Consistency (external)
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 1

Explanation: The interface pops up window in the upper left-hand corner of the screen. Six users commented that they were frustrated with this policy.
Recommendations: The responsibility for placing new windows properly belongs to the window manager. By removing the explicit placement command, windows will be placed on the screen according to the user's default settings (generally either set to force the user to select a placement or to automatically place the window at a set offset from the last window placement). This change will make the interface consistent with the user's windowing environment.

Finding #15: Interface loses definitions under certain conditions
Evidence:

Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: 6, 7
Qualitative Results: NA
Comment IDs: 19, 97, 127, 139

Explanation: Under certain conditions, the interface loses data, program, or task definition. The known circumstances include:

- saving the definition under the name "New..."
- selecting a different definition from the listbox without saving first
- closing window without saving first

Two user also experienced an unexplained lost of data.
Recommendations: The first two known circumstances are simply bugs caused by situations overlooked during the design and implementation of the interface. The final one is suspicious because the interface was specifically tested for this. So, either the bug is subtle or the user may not have named the definition and thus the bug falls into the first category listed above.

F.2.2 Local

Finding #16: Users experienced difficulty while prioritizing tasks.
Evidence:

Heuristic Violation: Simple & Natural
Tutorial Results: 9
Exercise Results: 8
Qualitative Results: NA
Comment IDs: 26, 42, 63, 64, 102

Explanation: Users experienced difficulty while prioritizing tasks. The problem could be related to finding the task in the list of defined tasks. It could be related to the bizarre form of drag-n-drop used because as a result of the limitations of the widget library. It could be related to difficulty remembering how to select a

task to be hoarded. It could be related to difficulty unhoarding currently hoarded tasks. It could be because of a bug in the drop algorithm that decides where the task has been dropped.
Recommendations: First, we need to fix the bug in the drop algorithm that determines where the user dropped the task. It is likely that this bug is the cause of the problem. Should reports continue after the bug is fixed, we will need to reexamine the problem, perhaps by observing a few new users of the system as they learn to prioritize tasks.

Finding #17: Users have difficulty entering data into "Data Definition" window
Evidence:
Heuristic Violation: Feedback
Error Messages
Prevent Errors (modal behavior)
Tutorial Results: 6
Exercise Results: 6
Qualitative Results: 2, 7, 9
Comment IDs: 3, 20, 36, 56, 101, 132, 196
Explanation: Users experienced two problems while defining user data elements. The first of these problems occurred when users entered a non-existent directory into the pathname entry area of the window. When this happened, users expected the meta-information area to become enabled (because they had entered the name of a directory). Because the system could not verify that the pathname entered was a directory, it did not enable the directory. There was no other feedback as to the problem. Indeed, the interface didn't even recognize the problem because it assumed the pathname to be a file. The second problem occurred when users tried to specify meta-information using <Tab> and <Enter>. Users were confused about the effect of these keys and were unsure about when each was appropriate to use.
Recommendations: The first problem is easy to remedy. The interface must check that the input pathname exists before checking to see if the input pathname is a directory. If the pathname does not exist, the interface must indicate this information to the user. The indication should take the form of a pop-up dialog box initially, but the interface might allow users to customize the indication to a beep or something similar. The second problem is more perplexing. The cause of the confusion is unclear to me. The <Tab> key advances the highlighting to the next area (as it does in many other windows and in other systems). The <Enter> key selects the current area. The solution may be to abandon these accelerators as "problem causing" and find different ones that cause less confusion. Users made a number of suggestions (indirectly by saying that they "expected" a certain key to have some effect). These suggestions included:

- Using <Back-Tab> to go backwards[6]
- Using the arrow keys to move around the meta-information[7]

One user also expected ^U to delete an entire pathname. Users also wanted to be able to click in the disabled meta-information area, though this desire may have been because they expected the meta-information to have been enabled for non-existent directories.

◆ **Finding #18:** Interface loses pathnames
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 107
Explanation: The interface occasionally loses a pathname after the user types it and hits <Return> in the "Program Information" window.
Recommendations: Fix the bug.

---

[6] Note that this expectation suggests that at least one user found the <Tab> intuitive.
[7] The right arrow key, then, would work like the <Tab> key does in the current implementation; the left arrow, then, like <Back-Tab> would.

Finding #19: Users experience difficulty finding an element in predefined list
Evidence:
Heuristic Violation: NA
Tutorial Results: 4
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 18, 110, 123
Explanation: Users have difficulty finding an element in the predefined list.
Recommendations: It remains unclear how to layout this window to make it easier to find definitions. One option would be to make the window larger, allowing more of the predefined list to be visible at one time. However, unless having a larger partial listing of all the predefine elements somehow makes it easier to orient to the entire list, this "solution" only delays the fundamental problem. The only other solution is to change the way in which tasks are defined, perhaps using a tree view similar to that offered in the Hoard Walk Advice Request window. Even this solution does not imply that all elements will be visible on the screen however. Addressing this problem completely is non-trivial, perhaps impossible.

Finding #20: User has difficulty unselecting element from the contains list
Evidence:
Heuristic Violation: Simple and Natural
Tutorial Results: 8
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 62
Explanation: Two users experienced difficulty when trying to remove an element from a task definition. One user guessed correctly the first time. The other guessed correctly on the second attempt, but deleted an entire task with his first guess.
Recommendations: Redesigning the "Task Definition" to use a tree view with check boxes would probably solve this problem (see Finding #19 above). Alternatively, an undo facility would help those users who guess incorrectly (see Finding #12 above).

◆ **Finding #21:** Submitting a password takes excessively long.
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 104
Explanation: After the user enters a password to the Authentication Information window, the completion of the authentication protocol takes excessively long (on the order of a minute).
Recommendations: This problem is not local to the interface. The interface uses the "clog" program to perform authentication. It is this program that experiences the delay. Coda developers are aware of the problem.

F.3 Level 3

A *level 3* usability problem has only a minor effect on usability.

F.3.1 Global

◆ **Finding #22:** User believes colors mean too many different things
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 262
Explanation: User believes that colors in the interface mean too many different things. In particular, the warning urgency color (yellow) is used to mean both "busy" and "warning".
Recommendations: The interface should introduce a distinct color to mean "busy", perhaps a brighter shade of green.

◆ **Finding #23:** Interface offers no feedback to indicate a window is already visible
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 208
Explanation: When the user double clicks on an indicator whose window is already visible, the interface does nothing.
Recommendations: The interface should raise the window in question.

Finding #24: Windows do not resize gracefully.
Evidence:
Heuristic Violation: Consistency (internal)
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 4
Explanation: When users enlarge certain windows, those windows do not take advantage of the extra screen real estate.
Recommendations: This is a bug in the interface that should be trivial to correct.

Finding #25: Users confused when help windows grab and maintain focus.
Evidence:
Heuristic Violation: Memory Overload
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 13
Explanation: When the user clicks on a HELP button, the system displays a new window that grabs and maintains the input focus. Users who attempt to input data in a definition window or change the configuration of an event while a help window is visible find that they cannot do so until they close the button. No indication is given to them that this is the cause of their problem, though most (if not all) of them figured it out with little problem.
Recommendations: The help windows should not grab or maintain the input focus.

Finding #26: Users confused that changing name and saving creates a new definition.
Evidence:
Heuristic Violation: NA
Tutorial Results: 8
Exercise Results: NA
Qualitative Results: 15
Comment IDs: 5
Explanation: When working with any of the three definition windows (Data, Program, or Task), changing the name of the current definition and saving creates a new definition with the specified name.
Recommendations: Correcting the "Save" behavior as described in Finding #13 above will also solve this problem.

Finding #27: Users dislike pathnames entry.
Evidence:
Heuristic Violation: Memory Overload
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: 11
Comment IDs: 21, 94
Explanation: Users comment that they dislike entering pathnames and would prefer to browse the file system hierarchy. Users also comment that they are concerned that the entire pathname is not visible on the screen.
Recommendations: The interface could allow users the option of typing in the entire pathname or browsing the file system to find the object for which they are looking. The concern over having the entire pathname not visible is more problematic since pathnames can be very long and screens are just a finite width. Once browsing is added, the final component could be visible in its entirety, but that does not solve the problem. This problem applies to both the "Data Definition" and "Program Definition" windows.

◆ **Finding #28:** Hoard Walk and Advice indicators confused.
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 10
Explanation: Two users were confused by the distinction between the Hoard Walk and Advice indicators. This confusion results from the Hoard Walk Advice Request being accessible from either indicator.
Recommendations: The Hoard Walk Advice Request should be accessible from only the Advice indicator. The Hoard Walk indicator should change to a red meter to indicate that the hoard walk has stalled.

Finding #29: Hoard Walk and Task indicators confused.
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: 4
Comment IDs: 31, 126
Explanation: Two users were confused by the distinction between the Hoard Walk and Task indicators. My hypothesis is that this confusion occurs because the Task indicator allows users to select tasks for hoarding and because all other hoard-related information appears under the Hoard Walk indicator.
Recommendations: Moving the task selection process into the Hoard Walk indicator seems like a reasonable idea at first glance. However, such an approach would cause unexpected consequences in the use of the Task indicator light to notify users of "Task Available" and "Task Unavailable" events. When

one of these two events occurs, the Task indicator light changes colors (or otherwise notifies the user of the event). If the user double-clicks on the indicator light, hoarded tasks that are available or unavailable are shown in the appropriate color. Without the hoarded task list in the "Task Information" window, this second-level of information would be meaningless. Moving this second-level information into the list of tasks would make it less obvious to users that only hoarded tasks are marked "available" or "unavailable", introducing a different and potentially more problematic type of confusion. Because of the potential for introducing more confusion, extreme care must be taken when addressing this problem.

Finding #30: User experiences window overload
Evidence:
Heuristic Violation: Memory Overload
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 24
Explanation: Four users commented that they were experiencing window overload. At the time, users had multiple windows open simultaneously.
Recommendations: Although the interface certainly allows users to get into a state of window overload, it does not require them to do so. Users control how many windows are open at any given time. The interface "requires" them to have three windows open simultaneously during the process of creating a task definition. For instance, a user might have the "Task Information" window opened, a "Task Definition" window opened, and a subordinate "Data Definition" window opened. Users can "shoot themselves in the foot" by opening more definition or other informational windows, but they may also find the ability to do so useful.

Finding #31: Interface mixes upper- and lower-case letters.
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 7
Explanation: The interface mixes upper- and lower-case letters in certain lists.
Recommendations: This is a bug. The interface should not do this. The only exception is the "New..." element, which should always appear first.

Finding #32: User confused by terminology
Evidence:
Heuristic Violation: Speak User's Language
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 209-234
Explanation: Various users report confusion over the meaning of a number of different terms. Of these reports, just one was reported by more than 2 users. Four users expressed confusion over the term "subtasks" to mean "tasks". It is interesting to note that the terms were chosen based upon an analogy to directories and subdirectories. Three users expressed confusion by the distinction between "hoarded" and "available".
Recommendations: Although most of the terminology comments were made by just a single user, it seems prudent to have a technical writer examine the interface, paying particular attention to those 25 terms, and suggest improvements.

Finding #33: Meters shown at 1% have no visible color.
Evidence:
Heuristic Violation: Feedback
Prevent Errors
Consistency (internal)
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 8
Explanation: Meters shown at low, positive percentages have no visible color, though meters shown at 0% have a small sliver of color.
Recommendations: The 0% meters have been handled specially to show a small sliver of color. Other small-valued percentages should be handled similarly.

Finding #34: <Enter> does not make the help window disappear.
Evidence:
Heuristic Violation: Consistency (with Tutorial and Exercises)
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 12
Explanation: Because of the behavior of the Tutorial and Exercises, users expect <Enter> to make the help windows disappear. Because each window of the Tutorial and Exercises had a button labeled OK and because <Enter> acted as a keyboard accelerator for the OK button on these windows, users reasonably expect <Enter> to be the keyboard accelerator for help windows, which also have a button labeled OK on them.
Recommendations: Add the keyboard accelerator.

Finding #35: Numbers are sometimes unreadable on meters
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 77
Explanation: When meters are almost full, the numbers overlap with the edges of the meter, making them difficult to read.
Recommendations: This is a bug.

Finding #36: Users confused by addition of empty entry widget
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 37
Explanation: When the user enters an element into either a program or a data definition, the interface adds an empty entry widget to the end of the definition. The purpose of the empty entry widget is to allow the user to enter the next element. The addition of this widget confuses users, sometimes causing them to do extraneous saves.
Recommendations: If it were possible to redesign the program and data definition windows to work without having to automatically add the empty entry widget and without requiring the user to request

another widget, the problem would probably disappear. I do not immediately see a clean way of doing this under these constraints.

F.3.2 Local

Finding #37: User surprised indicator lights are dynamic
Evidence:
Heuristic Violation: Consistency (external)
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 159
Explanation: Because the interface draws on a static metaphor and because the tutorial didn't draw the line of departure from that metaphor, one of the pilot users was surprised that the indicator lights were dynamic.
Recommendations: No real recommendation except that this topic should be addressed explicitly in any indicator help window and possibly in balloon help. After the Tutorial was revised, no other users commented about this issue.

Finding #38: User confused by the behavior of the Hoard Walk indicator
Evidence:
Heuristic Violation: Consistency
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 10, 11
Explanation: The Hoard Walk indicator will pop-up different windows depending upon the current system state. If the hoard daemon is not active, the Hoard Walk indicator will pop-up a window displaying the time that the next hoard walk is scheduled to begin. If the hoard daemon is currently running, the indicator will pop-up a window showing the progress of the current hoard walk. If the hoard daemon is currently waiting for advice, the indicator will pop-up a window presenting the advice request. Also, both the Hoard Walk and Advice indicators can pop-up the Hoard Walk Advice Request window when such a request is pending.
Recommendations: The windows associated with the Hoard Walk indicator exhibit modal behavior. The primary problem is that the Hoard Walk Advice Request window is vastly different from the other windows associated with this indicator light. If this request were to come in only under the Advice indicator (as suggested in **Finding #10** above) and if the progress meter were to turn red until the advice request were answered, user confusion could be lessened.

Finding #39: Users confused by two-color blinking.
Evidence:
Heuristic Violation: Memory Overload
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 9
Explanation: When an event arrives that is configured to flash the indicator light, the interface blinks between the old urgency color and the new urgency color for 10-15 seconds. Users find this flashing confusing because, unless they happen to remember what the old urgency color is, the new urgency color isn't obvious until the blinking stops.
Recommendations: One suggestion is to blink between dark gray and the new urgency color. This makes the new color obvious at the expense of forgetting the previous color. A second suggestion is to show the

new urgency color for longer periods of time and the old urgency color for shorter periods of time. Unfortunately, this second suggestion still doesn't make it entirely clear that the color shown longer is the new one. The former suggestion has been implemented.

Finding #40: Users somewhat confused because the indicator stops blinking.
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 33
Explanation: Users become somewhat confused when the indicator stops blinking. One user wondered whether or not this behavior meant that the problem had been solved.
Recommendations: Overall, this problem presents only a minor concern. Even though six users commented that the flashing had stopped, only one of them seemed confused by the behavior. Two of them continued their comment saying that they guessed that the flashing was simply intended to draw their attention. The user who seemed confused happened to double-click on the indicator light just as it stopped flashing. He then continued the commented saying that "the impression [he is] left with is that, by opening this window, somehow the danger has just gone away", at which point he chuckled. Although the timing here was unfortunate, I don't think he was terribly confused. Perhaps the best way to address this potential source of confusion is with an explanation in the help window of the Control Panel. Another "solution", which would likely serve to introduce a more serious usability problem, would be to have the indicator blink until the user investigated the problem, forcing users to investigate a problem that they may not wish to investigate and possibly encouraging the misconception that double-clicking fixes the problem.

Finding #41: User disagrees with default notification for certain events
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 257, 260
Explanation: Three users commented that they thought the "Token Expiry" event should be notified at a critical urgency level.
Recommendations: Because the user can modify the way in which event notification occurs, the defaults need not be changed. However, we may want to consider adding an event that warns users prior to token expiration (perhaps an hour or so beforehand). Such an event could be notified at the warning urgency level. We could then change the "Token Expiry" event to notify at the critical urgency level with no pop-up, flash, or beep and the "Activity Pending Tokens" event to notify at the critical urgency level with beeping.

Finding #42: Users search for a way to configure events
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 119, 147
Explanation: Five users had difficulty remembering how to configure events. One of those users did not recognize the Event Configuration tab as a way to do so.
Recommendations: I suspect that this problem is a learning problem. During the Tutorial, users were lead through the process of configuring an event step-by-step. In the Exercises, they were asked how an event was configured. I'm not sure how to make finding the location of event configurations easier.

Finding #43: Users unsure how an event will be notified.
Evidence:

Heuristic Violation: Feedback
Speak User's Language
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 38, 118, 164, 170, 206, 221

Explanation: While configuring events, users were frequently confused as to how an event would be notified. For example, five users were confused as to how the system would notify an event if all the notification options were turned off. They did not realize that the minimal notification was to simply change the color of the indicator light. One user was confused as to how the indicator light would change colors. One user was uncertain what exactly was going to "flash" (another was confused as to what "flash" meant). One user suggested that the configuration tab should contain a copy of the indicator lights and another user suggested that it show an example notification.
Recommendations: The two suggestions listed above are both reasonable. If the configuration tab contained a copy of the indicator lights, it could use them to indicate which indicator would notify the user of the event (perhaps with an arrow). And, it could have a button the user could press that would show how the event is currently configured to notify users. An alternative design would be for the system to continuously notify this event on the indicator lights shown on this tab, but this would be highly annoying if the user had the system configured to beep.

Finding #44: User assumes system uses a 24-hour clock.
Evidence:

Heuristic Violation: Speak User's Language
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 195

Explanation: In the Authentication Information window, the system includes the time at which the user's tokens are expected to expire. This time is presented in military time. Two users commented that they were going to assume military time, but expressed some uncertainty about the correctness of this assumption.
Recommendations: The interface could allow users to specify whether they would prefer a.m./p.m. designations or military time. This specification belongs within the Control Panel on a new tab called "Interface Configuration" (or something similar). Within that tab, users should find another notebook that includes a "Time" tab. A simple radiobutton would work well. Alternatively, the system could always present a.m./p.m. designations because such a format leaves no room for uncertainty.

Finding #45: Users confused by the meters in the "Space Information" window
Evidence:

Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: 2, 10
Qualitative Results: NA
Comment IDs: 270, 271, 280

Explanation: A total of four users had difficulty with the meters on the "Space Information" window in one way or another. Three of these users answered the relevant exercise incorrectly. One of these users and the fourth user commented that they were confused by the meanings of the meters. After consulting the help window, the fourth user answered the relevant exercise correctly. I'm not certain if the other users consulted the help window or not.
Recommendations: Although it is clear that the meters in this window generally confused users, it isn't clear to me how to correct the problem.

Finding #46: User confused by "questionnaire already running" dialog
Evidence:
Heuristic Violation: Error Messages
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 200
Explanation: When the user attempts to open an advice request which they have already opened, the interface pops up an error message dialog that explains that the advice request is already running. This user did not understand that dialog.
Recommendations: The explanation in the dialog may need clarification.

Finding #47: The Tix tree widget requires three clicks to expanding/contracting nodes
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: 4
Qualitative Results: NA
Comment IDs: 179, 204
Explanation: To expand or contract a node in the Tix tree widget, the user must click three times on the small checkbox widget. Furthermore, the checkbox widget has an extremely small active area.
Recommendations: The triple-click behavior was a bug in the Tix tree widget that disappeared once we upgraded to a more recent version of the Tix widget library. Furthermore, the checkbox behaves more rationally and in such a way that the small active area is not as annoying.

Finding #48: User dislikes having fetch estimates expressed in seconds
Evidence:
Heuristic Violation: Speak User's Language
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: 13
Comment IDs: 96
Explanation: The hoard walk advice request lists the estimated fetch time of each object in seconds. When expected fetch time is high, the number quickly becomes meaningless to users.
Recommendations: The estimated fetch time should be listed in minutes and seconds.

Finding #49: User confused by "unselect" dialog
Evidence:
Heuristic Violation: Error Messages
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 140
Explanation: The Hoard Walk Advice Request window presents this dialog to the user if the user unselects a definition (for fetching) that is included by multiple tasks. In this dialog, the interface gives the user the choice of unselecting a task to be fetched from a single task or from all tasks. The problem arises because some elements may be listed under different tasks. The dialog confused two users who stumbled across it.
Recommendations: This situation is difficult to handle. The initial recommendation would be to clarify the explanation in the dialog and then enter a period of "watchful waiting". If further symptoms of the problem appear, the problem will need to be reconsidered.

Finding #50: Task definitions unnatural
Evidence:
Heuristic Violation: Simple & Natural
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 157, 183
Explanation: Users find it unnatural that tasks are the only definitions that can be hoarded directly. This requirement means that users must create a task definition to hoard a single file or directory. Furthermore, to examine a data or program definition, they must first examine a task definition.
Recommendations: Any redesign of task definitions should consider this issue, but there is no obvious solution within the current design.

Finding #51: Users confused by the cache meter in the "Task Information" window
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 269
Explanation: One user wondered whether this meter meant that 97 MB (the size of the Venus cache) was available or whether 70% (the percentage shown on the meter) of that 97 MB was available. Another user also thought that the meter was showing the percentage of the cache currently available. The third user was completely confused by all of the meters in the "Task Information" window.
Recommendations: In fact, not one of these users was correct. The cache meter that appears at the top of this window shows what percentage of the cache is dedicated to hoarded objects. The help dialog for this window needs to be examined to make sure that the text is clear on this point. Another recommendation would be to add a label to this meter to inform users what the meter is showing.

◆ **Finding #52:** Newly hoarded task may not be visible in list of hoarded tasks
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 43
Explanation: After a user hoards a task, that task may not be visible in the list of hoarded tasks in the "Task Information" window.
Recommendations: The interface needs to make newly hoarded tasks visible.

Finding #53: Users confused by the task meters.
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 268
Explanation: Users express confusion over the meaning of the task meters in the "Task Information" window. This comment showed up most frequently during the Tutorial phase of the study, but occasionally came up during the Exercises. Users were able to use the help window to understand the meaning of the meters.
Recommendations: Labeling the columns of the hoarded task list is likely to alleviate this confusion. For this column, the label "Percent Available" seems appropriate.

Finding #54: User finds it strange that a single click selects an item for the contains list.
Evidence:
Heuristic Violation: Consistency (internal and external)
Tutorial Results: 8
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 23
Explanation: Almost half of the participants commented that they thought it strange that single clicking on an element in the predefined list of the "Task Definition" window selected it into the contains list.
Recommendations: Because double-clicking on an element opens it, we are not left with much of an alternative. If we were to change the interface to provide a tree-based task definition procedure (see Finding #19 above), this concern would be moot.

Finding #55: Interface double-selected elements under certain circumstances
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 22
Explanation: Under certain circumstances, the interface will double-select an element from the predefined list into the contains list of the "Task Definition" window.
Recommendations: This behavior was clearly a widget bug that disappeared upon upgrading to a more recent version of tcl/tk/tix.

Finding #56: Extraneous underlining appears in lists of the "Task Definition" window
Evidence:
Heuristic Violation: Simple & Natural
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 181
Explanation: The "Task Definition" window underlines entries in the contains and predefined lists in an arbitrary way. I suspect the underlying widgets consider these underlines to be a feature, but they confused users.
Recommendations: We could remove the underlining from our versions of the widget, but that would introduce portability problems.

Finding #57: Priorities need to be labeled
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 82
Explanation: In the hoarded task list of the "Task Information" window, the columns need to be labeled. In particular, the priority column needs to be identified as such.
Recommendations: Add a line containing labels for each column.

Finding #58: Meter may hide long task names
Evidence:
Heuristic Violation: Speak the user's language

Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 85

Explanation: The meter used to show the availability of hoarded tasks may hide the end of long task names.
Recommendations: The task name needs to be a read-only entry widget that the user can scroll around in.

Finding #59: User types a pathname into the name of a program definition
Evidence:

Heuristic Violation: Speak the user's language
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 137, 198

Explanation: Three users typed pathnames into the name entry widget of a program definition.
Recommendations: Users seemed somewhat confused by program definitions. Perhaps because the name does not imply that a "program" can contain more than one program so they thought that the name of the program was the pathname to the executable?

Finding #60: User doesn't believe system verifies existence of executable programs
Evidence:

Heuristic Violation: Prevent Errors
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 61

Explanation: User does not believe that the system verifies the existence of programs.
Recommendations: User is correct. The system does no such verification. In the interests of preventing misunderstandings, the system should verify their existance.

F.4 Level 4

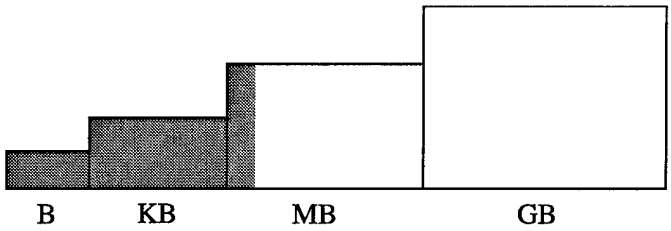
A *level 4* usability problem is more subtle and often suggests a future enhancement.

F.4.1 Global

Finding #61: Meters do not cover orders of magnitude or relative information very well
Evidence:

Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 50, 51

Explanation: Two sets of meters in the interface are required to cover multiple orders of magnitude. One set appears in the "Space Information" window and must cover a few megabytes of space to a few gigabytes of space. Another set appears in the "Network Information" window where the system must show differences between kilobytes (or no) network bandwidth and hundreds (or more) of megabits of bandwidth. As currently designed, the meters shown in these windows do a poor job of presenting differences that span orders of magnitude. Furthermore, they are unable to indicate how close the system is to exceeding a threshold or how far beyond the threshold the system has wandered.
Recommendations: One possible redesign the meters would be to use a pseudo-log scale applied to meters. Another possibility is to use meaningful breakpoints rather than log-scale breakpoints. For example, to use just one cut-off that corresponds to the threshold between strong and weak connectivity. Such a scheme would give users a visual indication regarding how close current network conditions are to the threshold.



Finding #62: User requests balloon help
Evidence:
Heuristic Violation: Help
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 90
Explanation: User requested balloon help be added to the interface.
Recommendations: Excellent area for improvement.

Finding #63: User believes task interface to be overly complicated
Evidence:
Heuristic Violation: Memory Overload
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 203
Explanation: User believes the task interface to be overly complicated. We have no further details as to what user meant.
Recommendations: Any redesign of the task definitions needs to consider simplifying the definition process. A tree view of tasks (see Finding #19 above) might help.

F.4.2 Local

- **Finding #64:** Interface is cognizant of vertical screen real estate, but not horizontal
Evidence:
Heuristic Violation: NA

Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 169

Explanation: The interface takes pains to reduce the amount of screen real estate required by the indicator lights. The indicator lights require just 1"x2" of the screen. However the interface does not provide a means for displaying the indicator lights horizontally across the top of the screen. Such a layout could require 9"x¼" of the screen. Users may be more willing to give up ¼" across the entire width of the screen than a block along one side.

Recommendations: The interface should allow a horizontal layout of the indicator lights.

Finding #65: Users would like help on the indicator lights window.
Evidence:

Heuristic Violation: Help
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 45

Explanation: Three users comment that they were surprised no help was available for the indicator lights themselves.

Recommendations: Adding a Help indicator might be appropriate.

Finding #66: User concerned about missing an event notification
Evidence:

Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: 16
Comment IDs: 41

Explanation: Two users were afraid of missing an event notification.

Recommendations: One suggested that the interface make it clear what is new since the last time the user looked at the indicators. Although this would be nice, it is a tad difficult for the interface to determine when the user last looked at the indicators—short of installing specialized hardware to monitor the user's eye movements. At best, we could make it clear, perhaps with an asterisk, what is new since the last time the user opened an indicator window.

Finding #67: User would like feedback regarding the "health" of Venus
Evidence:

Heuristic Violation: Feedback
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 25

Explanation: The user would like the interface to provide feedback regarding the "health" of Venus. By this, the user could mean many things. One possibility is whether or not Venus is "alive".

Recommendations: The interface could indicate to users that its connection to Venus is alive by using all of the indicator lights. If the interface detects that its connection to Venus has disappeared, it could indicate this by turning all indicator lights dark gray. Once the connection is reestablished, the indicator lights could go back to their actual urgency state.

◆ **Finding #68:** User would like different configuration settings.
Evidence:

Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 187

Explanation: User suggested that the system allow different configuration settings. For example, he'd like to have one set of configurations for use "at school", a second one for use "at home", and a third set for use "on the road". His reasoning is that when he is at school, the fact that he is disconnected is interesting and important news. When he's at home, it's the norm.

Recommendations: This suggestion seems to be a reasonable request. The Control Panel will need an additional tab that allows the user to switch between different configurations easily.

Finding #69: User would like a set of system defaults
Evidence:

Heuristic Violation: Clearly Marked Exits
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 145

Explanation: User suggests having a set of system defaults that the user could revert back to if necessary.

Recommendations: Such a set of system defaults would give new users confidence to explore different configurations and seems reasonable.

◆ **Finding #70:** User wants to modify all event notifications at once
Evidence:

Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 87

Explanation: Two users wished for the ability to modify all event notifications at once. Such a facility would be particularly useful for users who would like to turn beeps or pop-ups off all at once.

Recommendations: Extending the interface in this way would provide a modest improvement in usability, but other changes should take precedence.

Finding #71: User would like to be able to stop long transfers over weak networks
Evidence:

Heuristic Violation: Clearly Marked Exits
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: 18
Comment IDs: 178

Explanation: When the system fetches an object over a weak network connection, users would like to have the ability to abort the fetch.

Recommendations: This suggestion, though a very reasonable request, is difficult to implement with the current RPC2 package. To abort a fetch, Venus would have to completely tear down the RPC2 connection and re-establish it. In the long-term however, users need this feature.

Finding #72: User would like a count-down timer on advice requests
Evidence:

Heuristic Violation: Feedback
Tutorial Results: NA

Exercise Results: NA
Qualitative Results: NA
Comment IDs: 68

Explanation: User would like to see a count-down timer on any advice requests that timeout after a given amount of time.
Recommendations: This is a reasonable request.

Finding #73: User wants expert help for Space problems
Evidence:

Heuristic Violation: Help
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 76

Explanation: User wants the system to explain how to solve space problems. In particular, the user wants the system to explain how to reinitialize Venus, how to increase RVM size, and how to increase cache size.
Recommendations: If the user were to reinitialize Venus while operating disconnected, they would lose any modifications made after the point of disconnection. Because of the potential for data loss, the help window simply instructs users to contact their system administrator. Although it would not be difficult to provide instructions for any of these procedures, I question the wisdom of doing so.

Finding #74: The Network help screen should explain why the network is degraded
Evidence:

Heuristic Violation: Help
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 250

Explanation: User wants the Network indicator's help window to explain why the network is down or weak.
Recommendations: Although this suggestion sounds spectacular, the request is unreasonable because it is unlikely that the system will know why the network connection is down or weak. To the extent that the system knows this information, it should make the information available.

Finding #75: User wants the Task help screen to explain why a task is unavailable
Evidence:

Heuristic Violation: Help
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 252

Explanation: User wants the Task indicator's help window to explain why a task is unavailable.
Recommendations: Again, to the extent that the system knows why a task is unavailable, it should make that information available to the user.

Finding #76: System should predict if hoarding a task would cause space problems
Evidence:

Heuristic Violation: Minimize User's Memory Load
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA

Comment IDs: 15, 273

Explanation: When the user selects a task to be hoarded, the system should predict whether or not doing so will cause a space problem. If it would, the system should warn the user. Furthermore, the system should calculate how much additional space would be required to hoard everything the user has requested.

Recommendations: These suggestions are both good ideas for future extensions. However, there are limitations to the system's ability to predict space problems. The space estimates for tasks are only estimates, based upon the last hoard walk. The next hoard walk may discover that the size of a task has either increased or decreased. Within the limitations of the system's knowledge, the interface should certainly be able to offer this information to the user.

Finding #77: User would like a tree view of tasks

Evidence:

Heuristic Violation: Consistency

Tutorial Results: NA

Exercise Results: NA

Qualitative Results: NA

Comment IDs: 110

Explanation: User suggests using a tree view in the "Task Definition" window similar to that used in the Hoard Walk Advice Request window.

Recommendations: Although this suggestion is appealing, its implementation involves some rethinking of the task definition. One stumbling block is how users would specify meta-information for data directories (e.g. immediate children vs. all descendants). Overall, this suggestion sounds like a good idea, but will require careful thought.

Finding #78: The "Data Definition" window should indicate the size of subtrees.

Evidence:

Heuristic Violation: Memory Load

Tutorial Results: NA

Exercise Results: NA

Qualitative Results: NA

Comment IDs: 57

Explanation: In the "Data Definition" window, the size of each subtree should be indicated to help users determine the appropriate meta-information.

Recommendations: Adding this information appears at first glance to be a very simple way to improve the usability of this interface. Unfortunately, obtaining the information could be somewhat tricky. The information is not maintained dynamically so it would have to be calculated, requiring a recursive walk down the tree. Such a walk would require us to fetch all of the data contained in the tree and could have far-ranging effects by triggering resolutions. Getting this information to the interface is an area to explore as it would certainly help users make these decisions, but it will require care to implement efficiently.

Finding #79: User wants more from program definitions

Evidence:

Heuristic Violation: Simple and Natural

Tutorial Results: NA

Exercise Results: NA

Qualitative Results: NA

Comment IDs: 153, 173, 190

Explanation: User wants to be able to see the details of program definitions and expects program configuration files (*e.g.* `.cshrc` or `.twmrc`) to appear in program definitions automatically.

Recommendations: Both requests seem to be reasonable extensions to the current system, though showing the details of the program definition should not be the default.

Finding #80: User expects interface to track and automatically hoard missing objects
Evidence:
Heuristic Violation: Minimize User's Memory Load
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: 23
Comment IDs: 103
Explanation: Because the system asks the user about the severity of disconnected cache misses, the user expects the system to use this information to track these misses and automatically hoard the more important ones.
Recommendations: This request seems very reasonable.

◆ **Finding #81:** Bogus fetch statistics confuse user
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 99
Explanation: One user was confused by the bogus fetch statistics in the Hoard Walk Advice Request window. The Tutorial specifically told the user to ignore this area of the screen, but he evidently missed the instruction or didn't understand it.
Recommendations: Implement this feature of the window.

Finding #82: Users confused when a file becomes unavailable
Evidence:
Heuristic Violation: Feedback
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 29
Explanation: At one point during the Exercises, all tasks were available. Shortly thereafter, a task became unavailable. Three users were uncertain as to why it was no longer available. Later, a different user became confused because the interface asked whether or not a file within a previously available task should be fetched over a weak connection.
Recommendations: Although the second of these two situations points to what is likely a bug in the Exercises, the first situation points to a usability problem. Users are clearly having difficulty understanding under what conditions an object (and therefore a task) can become unavailable. The problem is that the interface does not explain why something has become unavailable. The user has no idea whether it was updated at the servers or whether we simply do not know whether or not the object is valid (e.g. the client has lost its callback). The interface offers no hint as to the problem. One possible solution to the problem is to track this information by paying attention to whether we've lost contact with a server or a volume validation failed or a file validation failed. This information could then be presented to the user, perhaps if the user right-clicks on the task entry in the hoarded list. This problem and its proposed solution needs careful attention. I question the wisdom of exposing the details of cache consistency to novice users.

◆ **Finding #83:** User uncertain whether or not a program is available
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: 10
Comment IDs: NA

Explanation: User was not certain whether or not a program was available in the cache. His uncertainty occurred in the process of providing hoard walk advice. The advice request did not mention the program in question.
Recommendations: At least part of the user's confusion can be attributed to the artificial nature of the Exercises. He had not created the task definitions that he was using in the advice request so he didn't have the information needed to realize whether or not the task definition contained the program. Because the hoard walk advice request only lists those objects not in the cache, he was unable to use his background knowledge to deduce that the program was available. The question that remains is whether or not this is a good model. Even users who created the original definitions are likely to forget them over time. The design should be reevaluated with this information in mind to determine if there is a clean way to add this information to the request without overloading the user's ability to comprehend the information available.

Finding #84: Users express concern over difficulty knowing what data is required
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 185
Explanation: Users express concern over the difficulty in knowing precisely which files and programs are necessary to work on a given task.
Recommendations: Determining what information to hoard is, indeed, a difficult task. Any additional help the system can provide is well worth the effort. For instance, the interface could track what programs access data the user has hoarded and offer hints if the user has not also hoarded those programs.

Finding #85: User would like a graphical representation of what is in the cache
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: 10
Comment IDs: 205
Explanation: One expert user expressed uncertainty over whether a particular program was available in the cache. Another expert user suggested that the system offer a graphical representation of what is currently cached.
Recommendations: I have mixed opinions regarding the value of such a tool. On the one hand, I see it as exposing too much information regarding caching—in essence, it makes caching opaque. On the other, I see that it offers a secondary way for users to determine what is and is not available for use during disconnected operation. As a tool for developers and experts (the user making the suggestion fell into this category), it would certainly be valuable.

◆ **Finding #86:** User would like a visual representation of Venus' on-going activities
Evidence:
Heuristic Violation: NA
Tutorial Results: NA
Exercise Results: NA
Qualitative Results: NA
Comment IDs: 243
Explanation: One user suggested a visual representation of Venus' on-going activities, akin to the current mariner shown in the codacon. In essence, the user has requested that the codacon be incorporated into the new interface.
Recommendations: Again, I have mixed opinions regarding the value of such an extension. The codacon is a useful tool for developers and even expert users. Unifying the two tools makes the developers job easier at the expense of adding additional information into a user interface intended to help novices. The

information provided in the codacon is not something that novice users should be exposed to. Yet, should they run into a problem, the information in such a tool would be useful to system administrators and developers. If such an extension is added to the interface, it must be clearly marked as being intended for developers and system administrators.

Finding #87: Users may feel limited by constraint that only tasks can be hoarded
Evidence:

Heuristic Violation: Simple & Natural Dialogue
Tutorial Results: NA
Exercise Results: 14
Qualitative Results: 8
Comment IDs: NA

Explanation: Users had to reorganize their tasks so that they could give a priority to something they had originally included as a subtask.
Recommendations: This behavior suggests that users may feel constrained by the limit that only tasks may be prioritized. Future versions of the interface may need to reconsider this design decision.

Bibliography

[1] *Dashboard: A Knowledge-Based Real-Time Control Panel.* Santa Cruz, CA: UCO/Lick Observatory,www.ucolick.org/~de/Docs/tcl97ht/TCL97.html.

[2] *Net.Medic.* www.vitalsigns.com.

[3] Alonso, Rafael; Barbará, Daniel; Cova, Luis L. Using Stashing to Increase Node Autonomy in Distributed File Systems. In *Proceedings of the Ninth Symposium on Reliable Distributed Systems* (Huntsville, AL; October 1990). Los Alamitos, CA: IEEE Comput. Soc. Press.

[4] Barrett, R. C.; Selker, E. J.; Rutledge, J. D.; Olyha, R. S. Negative Inertia: A Dynamic Pointing Function. In *Human Factors in Computing Systems: CHI '95 Conference Companion* (Denver, Colorado; May 1995).

[5] Birrell, Andrew D. and Nelson, Bruce J. Implementing Remote Procedure Calls. *ACM Transactions on Computer Systems* 2(1):39-59. (February1984).

[6] Boff, Kenneth R.; Kaufman, Lloyd; Thomas, James P. (Editors). *Handbook of Perception and Human Performance.* New York: Wiley, 1986.

[7] Card, Stuart K.; Moran, Thomas P.; Newell, Allen. *The Psychology of Human-Computer Interaction.* Hillsdale, NJ: L. Erlbaum Associates, 1983.

[8] Cooper, Alan. *About Face: The Essentials of User Interface Design.* Foster City, CA: IDG Books Wordwide, 1995.

[9] Cornsweet, Tom N. *Visual Perception.* New York: Academic Press, 1970.

[10] Dumas, Joseph S. and Redish, Janice C. *A Practical Guide to Usability Testing.* Norwood, NJ: Ablex Publishing Corporation, 1993.

[11] Ericsson, K. A. and Simon, Herbert A. Verbal Reports As Data. *Psychological Review* 87(3):215-51. (May1980).

[12] Hippocrates. *Of the Epidemics.* Translator Adams, Francis.

[13] Howard, John H.;Kazar, Michael L.;Menees, Sherri G.;Nichols, David A.;Satyanarayanan, Mahadev;Sidebotham, Robert N.; West, Michael J. Scale and Performance in a Distributed File System. *ACM Transactions on Computer Systems* 6(1):51-81. (February1988).

[14] Huston, Larry B. and Honeyman, Peter. Disconnected Operation for AFS. In *Proceedings of the USENIX Mobile and Location-Independent Computing Symposium* (Cambridge, Massachusetts; August 1993). Berkeley, CA: USENIX Association.

[15] Huston, Larry B. and Honeyman, Peter. Partially Connected Operation. In *Proceedings of the Second USENIX Symposium on Mobile and Location-Independent Computing* (Ann Arbor, Michigan; April 1995). Berkeley, CA: USENIX Association.

[16] John, Bonnie E. TYPIST: A Theory of Performance in Skilled Typing. *Human-Computer Interactions* 11(4):321-55. (1996).

[17] Kaashoek, M. F.; Engler, Dawson R.; Ganger, Gregory R.; Briceño, Héctor M.; Hunt, Russell; Mazières, David; Pinckney, Thomas; Grimm, Robert; Jannotti, John; Mackenzie, Kenneth. Application Performance and Flexibility on Exokernel Systems. In *Proceedings of the Sixteenth ACM Symposium on Operating Systems Principles* (Saint-Malo, France; October 1997).

[18] Kaiser, Gail E.;Feiler, Peter H.; Popovich, Steven S. Intelligent Assistance for Software Development and Maintenance. *IEEE Software* 5(3):40-49. (May1988).

[19] Kernighan, Brian W. and Pike, Rob. *The UNIX Programming Environment.* Englewood Cliffs, NJ: Prentice-Hall, 1984.

[20] Kiczales, Gregor. Beyond the Black Box: Open Implementation. *IEEE Software.* (January1996).

[21] Kieras, David. *A Guide to GOMS Task Analysis.* University of Michigan, 1994.

[22] Kistler, James J. *Disconnected Operation in a Distributed File System.* Doctoral dissertation, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, May1993.

[23] Kistler, James J. and Satyanarayanan, Mahadev. Disconnected Operation in the Coda File System. *ACM Transactions on Computer Systems* 10(1):3-25. (February1992).

[24] Kuenning, Geoffrey. Personal communication, February 1998.

[25] Kuenning, Geoffrey H. The Design of the SEER Predictive Caching System. In *Proceedings of the Workshop on Mobile Computing Systems and Applications* (Santa Cruz, CA; December 1994). Los Alamitos, CA: IEEE Comput. Soc. Press.

[26] Kuenning, Geoffrey H. *Seer: Predictive File Hoarding for Disconnected Mobile Operation*. Doctoral dissertation, University of California, Los Angeles, Los Angeles, CA, May1997.

[27] Kuenning, Geoffrey H. and Popek, Gerald J. Automated Hoarding for Mobile Computers. In *Proceedings of the Sixteenth ACM Symposium on Operating Systems Principles* (Saint-Malo, France; October 1997).

[28] Kumar, Puneet. *Mitigating the Effects of Optimistic Replication in a Distributed File System*. Doctoral dissertation, Carnegie Mellon University, Pittsburgh, PA, December1994.

[29] Lam, Ioi K. Designing Mega Widgets in the Tix Library. In *Proceedings of the Tcl/Tk Workshop* (Toronto, Ont., Canada; July 1995). Berkeley, CA: USENIX Associ.

[30] Lundell, Jay and Anderson, Steve. Designing a "Front Panel" for Unix: The Evolution of a Metaphor. In *Human Factors in Computing Systems: CHI '95 Conference Proceedings* (Denver, CO; May 1995). Denver.CO.

[31] Molich, Rolf and Nielsen, Jakob. Improving a Human-Computer Dialogue. *Communications of the ACM* 33(3):338-48. (March1990).

[32] Monk, Andrew. Mode Errors: a User-Centred Analysis and Some Preventative Measures Using Keying-Contingent Sound. *International Journal of Man-Machine Studies* 24(4):313-27. (April1986).

[33] Mummert, Lily B. *Exploiting Weak Connectivity in a Distributed File System*. Doctoral dissertation, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, December1996.

[34] Mummert, Lily B.; Ebling, Maria R.; Satyanarayanan, Mahadev. Exploiting Weak Connectivity for Mobile File Access. In *Proceedings of the Fifteenth ACM Symposium on Operating Systems Principles* (Copper Mountian Resort, Colorado; December 1995).

[35] Newell, Allen. *Unified Theories of Cognition*. Cambridge, Massachusetts: Harvard University Press, 1990.

[36] Nielsen, Jakob. Heuristic Evaluation. In *Usability Inspection Methods*, Jakob Nielsen and Robert L. Mack (Editors). New York: Wiley, 1994.

[37] Nielsen, Jakob. *Usability Engineering*. Boston: AP Professional, 1993.

[38] Nielsen, Jakob and Landauer, Thomas K. A Mathematical Model of the Finding of Usability Problems. In *Human Factors in Computing Systems. Proceedings of INTERCHI '93* (Amsterdam, Netherlands; April 1993). Amsterdam, Netherlands: IOS Press.

[39] Ousterhout, John K. *Tcl and the Tk Toolkit*. Reading, Massachusetts: Addison-Wesley, 1994.

[40] Pausch, Randy. Personal communication, July 1997.

[41] Person, Ron. *Special Edition Using Windows 95*. Indianapolis, IN: Que, 1996.

[42] Pike, Rob and Kernighan, Brian W. Program Design in the UNIX Environment. *AT&T Bell Laboratories Technical Journal* 63(8, part 2):1595-605. (October1984).

[43] Rutledge, Joseph D. and Selker, Ted. Force-to-Motion Functions for Pointing. In *Human-Computer Interaction – INTERACT '90* (Cambridge, U.K.; August 1990), Editors D. Diaper, G. Cockton, D. Gilmore, and B. Shackel. North-Holland: Elsevier Science Publishers.

[44] Satyanarayanan, Mahadev. Integrating Security in a Large Distributed System. *ACM Transactions on Computer Systems* 7(3):247-80. (August1989).

[45] Satyanarayanan, Mahadev.*RPC2 User Guide and Reference Manual*. Carnegie Mellon University, School of Computer Science, 1991.

[46] Satyanarayanan, Mahadev;Kistler, James J.;Kumar, Puneet;Okasaki, Maria E.;Siegel, Ellen H.; Steere, David C. Coda: A Highly Available File System for a Distributed Workstation Environment. *IEEE Transactions on Computers* 39(4):447-59. (April1990).

[47] Satyanarayanan, Mahadev;Mashburn, Hank H.;Kumar, Puneet;Steere, David C.; Kistler, James J. Lightweight Recoverable Virtual Memory. *ACM Transactions on Computer Systems* 12(1):33-57. (February1994).

[48] Shneiderman, Ben. *Designing the User Interface: Strategies for Effective Human-Computer Interaction* (Third edition). Reading, Massachusetts: Addison Wesley Longman, 1998.

[49] Skopp, Peter D. and Kaiser, Gail E. Disconnected Operation in a Multi-User Software Development Environment. In *Proceedings of the IEEE Workshop on Advances in Parallel and Distributed Systems* (Princeton, NJ; October 1993).

[50] Stinson, Craig. *Running Microsoft Windows95: in-Depth Reference and Inside Tips From the Software Experts*. Redmond, Washington: Microsoft Press, 1995.

[51] Tait, Carl D. and Duchamp, Dan. Detection and Exploitation of File Working Sets. In *Proceedings of the Eleventh International Conference on Distributed Computing Systems* (Arlington, TX; May 1991). Los Alamitos, CA: IEEE Comput. Soc. Press.

[52] Tait, Carl; Lei, Hui; Acharya, Swarup; Chang, Henry. Intelligent File Hoarding for Mobile Computers. In *MobiCom '95: Proceedings of the First Annual International Conference on Mobile Computing and Networking* (Berkeley, CA; November 1995).

[53] Tesler, Larry. The Smalltalk Environment. *Byte* 6(8):90-147. (August1981).

[54] Virzi, Robert A. Refinding the Test Phase of Usability Evaluation: How Many Subjects Is Enough? *Human Factors* 34(4):457-68. (1992).

[55] Virzi, Robert A. Streamlining the Design Process: Running Fewer Subjects. In *Proceedings of the Human Factors Society 34th Annual Meeting* (1990).

[56] Welch, Brent B. *Practical Programming in Tcl and Tk.* Upper Saddle River, New Jersey: Prentice Hall PTR, 1995.

[57] Wharton, Cathleen; John Rieman; Clayton Lewis; Peter Polson. The Cognitive Walkthrough Method: A Practitioner's Guide. In *Usability Inspection Methods*, Jakob Nielsen and Robert L. Mack (Editors). New York: Wiley, 1994.

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Carnegie Mellon University does not discriminate and Carnegie Mellon University is required not to discriminate in admission, employment, or administration of its programs or activities on the basis of race, color, national origin, sex or handicap in violation of Title VI of the Civil Rights Act of 1964, Title IX of the Educational Amendments of 1972 and Section 504 of the Rehabilitation Act of 1973 or other federal, state, or local laws or executive orders.

In addition, Carnegie Mellon University does not discriminate in admission, employment or administration of its programs on the basis of religion, creed, ancestry, belief, age, veteran status, sexual orientation or in violation of federal, state, or local laws or executive orders. However, in the judgment of the Carnegie Mellon Human Relations Commission, the Department of Defense policy of, "Don't ask, don't tell, don't pursue," excludes openly gay, lesbian and bisexual students from receiving ROTC scholarships or serving in the military. Nevertheless, all ROTC classes at Carnegie Mellon University are available to all students.

Inquiries concerning application of these statements should be directed to the Provost, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, telephone (412) 268-6684 or the Vice President for Enrollment, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, telephone (412) 268-2056.

Obtain general information about Carnegie Mellon University by calling (412) 268-2000.