4

RADC-TR-90-128
Final Technical Report
June 1990

INTELLIGENT MULTI-MEDIA INTEGRATED INTERFACE PROJECT

Calspan-University of Buffalo Research Center (CUBRC)

Sponsored by
Defense Advanced Research Projects Agency
ARPA Order No. 6085

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

Rome Air Development Center
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700

This report has been reviewed by the RADC Public Affairs Division (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-90-128 has been reviewed and is approved for publication.

APPROVED: Sharon M. Walter

SHARON M. WALTER
Project Engineer

APPROVED: Raymond P. Urtz Jr

RAYMOND P. URTZ, JR.
Technical Director
Directorate of Command & Control

FOR THE COMMANDER: Igor G. Plonisch

IGOR G. PLONISCH
Directorate of Plans & Programs

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (COES) Griffiss AFB NY 13441-5700. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

INTELLIGENT MULTI-MEDIA INTEGRATED INTERFACE PROJECT

J. G. Neal
S. C. Shapiro
C. Y. Thielman
J. R. Gucwa
J. M. Lammens
D. J. Funke
J. S. Byoun
R. Paul
Z. Dobes
S. Glanowski
M. S. Summers

Contractor: Calspan-University of Buffalo Research Center (CUBRC)
Contract Number: F30602-87-C-0136
Effective Date of Contract: 28 Oct 87
Contract Expiration Date: 27 Oct 89
Short Title of Work: Intelligent Multi-Media Integrated Interface Project
Period of Work Covered: Oct 87 - Oct 89

Principal Investigator: Dr. Jeannette G. Neal
Phone: (716) 631-6900

Project Engineer: Sharon M. Walter
Phone: (315) 330-3577

Approved for public release; distribution unlimited.

This research was supported by the Defense Advanced Research Projects Agency of the Department of Defense and was monitored by Sharon M. Walter RADC (COES), Griffiss AFB NY 13441-5700, under Contract F30602-87-C-0136.

REPORT DOCUMENTATION PAGE

Form Approved
OPM No. 0704-0188

1. AGENCY USE ONLY *(Leave Blank)*

2. REPORT DATE
June 1990

3. REPORT TYPE AND DATES COVERED
Final Oct 87 to Oct 89

4. TITLE AND SUBTITLE
INTELLIGENT MULTI-MEDIA INTEGRATED INTERFACE PROJECT

5. FUNDING NUMBERS
C - F30602-87-C-0136
PE - 61101E
PR - F085
TA - 01
WU - 00

6. AUTHOR(S)
J. G. Neal
S. C. Shapiro
C. Y. Thielman
J. R. Gucwa
J. M. Lammens
D. J. Funke
J. S. Byoun
R. Paul
Z. Dobes
S. Glanowski
M. S. Summers

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Calspan-University of Buffalo Research Center (CUBRC)
P.O. Box 400
4455 Genesee Street
Buffalo NY 14225

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Advanced Research
Projects Agency (DARPA)
1400 Wilson Blvd
Arlington VA 22209-2389

Rome Air Development Center (COES)
Griffiss AFB NY 13441-5700

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
RADC-TR-90-128

11. SUPPLEMENTARY NOTES
RADC Project Engineer: Sharon M. Walter /COES/(315) 330-3577

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report discusses the results of the Intelligent Multi-Media Interfaces (IMMI) project. The IMMI project has been devoted to the application of aritificial intelligence technology to the development of human-computer interface technology that integrates speech input, speech output, natural language text, graphics, and deictic gestures for interactive dialogues between human and computer. These dialogues are modelled on the manner in which two people naturally communicate in coordinated multiple modalities when working at a graphics device. The goal has been to develop multi-modal human-computer interface technology that enhances human-computer communication and is more capable of meeting the demands of modern information intensive systems.

As part of this project, a knowledge-based multi-media interface system, called CUBRICON (the CUBRIC Intelligent CONversationalist), has been developed as a proof-of-concept prototype. The CUBRICON system is described in this report.

14. SUBJECT TERMS
Artificial Intelligence
Human-Computer Interfaces
Intelligent Interfaces
Multi-Media Interfaces
Interfaces

15. NUMBER OF PAGES
286

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Contents

APPENDICES

1 Introduction

1.1 Motivation

The introduction of improved and advanced processing capabilities into Air Force Command and Control (C2) systems is proceeding at an ever-increasing rate. As the number and sophistication of military information processing systems rapidly increase, the impact on human operational users must be considered very carefully. Typically, large amounts of information must be communicated for use by the human operator in performing time-critical decision-making tasks for command and control functions. The problem is to make such sophisticated systems easy for military operators to use quickly and efficiently. These modern information processing and decision-aiding systems require a well-integrated selection of communication media to facilitate interaction and provide the increased bandwidth for information transfer with the human user.

It is essential that the human-computer interfaces to information intensive systems not become limiting factors which degrade the C2 functions. Too often in the past, the human-computer interface was either overlooked or handled much like a retrofit after the fact. In today's information-explosive environment it is critical that human-computer interface technology be developed and applied to meet the demands of modern sophisticated computer-based systems. Martin [Martin73] expressed it well:

> "For man, this is a hostile environment. His mind could no more cope with this deluge of data, than his body could cope with outer space. He needs protection. The computer - in part the cause of the problem - is also the solution to the problem. The computer will insulate man from the raging torrents of information that are descending upon him."

This report discusses the results of the Intelligent Multi-Media Interfaces (IMMI) project. This research effort was motivated by the need for more effective human-computer interface technology. The IMMI project has been devoted to the application of artificial intelligence technology to the development of human-computer interface technology that integrates speech input, speech output, natural language text, graphics, and deictic gestures for interactive dialogues between human and computer. These dialogues are modelled on the manner in which two people naturally communicate in coordinated multiple modalities when working at a graphics device. The goal has been to develop multi-modal human-computer interface technology that enhances human-computer communication and is more capable of meeting the demands of modern information intensive systems. By accepting user inputs via combinations of multiple media selected by the user, the resulting human-computer interface was expected to be both natural to use and highly effective. By dynamically selecting output media based on the features and capabilities of that media vis-a-vis human sensing and understanding mechanisms, while also considering the context of the communication and combining multiple output media to achieve increased bandwidth and expressive redundancy, it was expected that the human-computer interface efficiency could be enhanced.

The development of new user interface technology is driven by interface requirements but is constrained and/or enabled by computer hardware and software capabilities. With the increased functionality and reliability provided by new developments in human-computer interface devices such as speech recognition and production systems, high-resolution color and monochrome graphic displays, and pointing devices as well as the availability of increasingly powerful workstation environments, it is a natural and timely step in the evolution of human-computer interfaces that the media be integrated to meet the information processing needs of the user community. Before parallel processing and high resolution integrated displays technology, human-computer interfaces employing integrated windowing environments were not possible. Today these are commonplace. A plethora of human factors research projects have explored, or been undertaken to explore, how to best apply these technologies. This project has applied the emerging technologies of artificial intelligence, natural language understanding, graphics, and voice recognition and synthesis to the human-computer interface.

Human-computer interface technology is developed in an empirical fashion. It is necessary to build upon our existing understanding of interface design techniques and our knowledge of past, current, and emerging human-computer interface requirements, to design, evaluate, and build better human-computer interfaces. As new technology provides opportunity for better human-computer interfaces, testbeds for applying these technologies and exploring alternative implementation approaches are needed. CUBRICON was developed as one such testbed, and this project has started the process of determining how best to apply these technologies to achieve the goals of an enhanced human-computer interface.

1.2 Functionality Overview

The CUBRICON system design is based upon a *unified view of language*. Language is a means of communication, whether verbal, visual, tactile, or gestural. Human beings communicate with each other via written and spoken natural language, frequently supplemented by pictures, diagrams, pointing to objects, and other gestures. It is a unified language, in that these various modalities are integrated and combined to represent and describe a single underlying reality.

The CUBRICON system design provides for the use of a unified multi-media language, defined by an integrated grammar, consisting of textual, graphic, and combined text/graphic symbols. Inputs to, and outputs from, *CUBRICON* are treated as compound symbol streams with components corresponding to different media. This approach is intended to imitate to a certain extent, the ability of humans to simultaneously accept input from different sensory channels (such as eyes and ears), and to simultaneously generate communications in different media (such as voice, pointing motions, and drawings).

The CUBRICON system includes: (a) language parsing and generation capabilities to support the understanding and creation of multi-media I/O streams, (b) knowledge representation and inferencing capabilities to provide for reasoning about the meanings of all communications vis-a-vis the underlying application, (c) knowledge bases and models to provide a basis for intelligent decision-making, and (d) automated knowledge-based media selection and the response formulation that takes advantage of human sensing and understanding capabilities.

Functionally, CUBRICON is distinct from other human-computer interface systems because it provides intelligent integration and use of multi-media input and output. The following unique features are important parts of the CUBRICON capability:

- CUBRICON integrates multiple input and output modalities. Input modalities include voice, pointing via mouse, form-based input, and typed text. Output modalities include voice, maps, pointing/highlighting, forms, tables, and typed text. CUBRICON's unified view of language will allow efficient addition of other input and output modalities if desired.

- CUBRICON accepts inputs from human users in a manner that is natural and desirable to the user. Specifically CUBRICON:

  - Coordinates input from different devices.
  - Allows a variety of object types to be the targets of point gestures.
  - Accepts varying numbers of point gestures within a phrase.
  - Accepts varying numbers of multi-modal phrases within sentences.
  - Uses natural language inputs to disambiguate corresponding point gestures.
  - Handles certain types of ill-formed multi-modal inputs.

- CUBRICON provides for intelligent and automatic management of windows. This includes:

  - A method for determining window importance (used for deciding which windows to remove when display space is needed for other windows).
  - A procedure for automatically managing windows in a dual monitor environment. This procedure considers window importance and type.

- CUBRICON generates multi-media outputs in a manner that enhances understandability. Specific features are:

- Relevant information is selected by CUBRICON for presentation to the user. Relevant information is that which is (1) specifically requested by the user, (2) relevant to the dialogue, (3) relevant to the user's task, and (4) helps maintain consistency between related displays.
- Modality selection is based on the characteristics of the information to be expressed vis-a-vis human sensing and understanding capabilities, as well as task and dialog context.
- Multiple modalities are combined to: 1) take best advantage of the relative strengths of each; 2) add emphasis or orientation to accompanying modalities; and 3) provide redundancy to ensure understanding and notice of important information.
- All multi-media outputs are temporally synchronized (e.g., highlighting of graphics is temporally coordinated with related speech).
- Spoken and written natural language outputs are designed for short-term and long-term reference, respectively. For example, written outputs include specific object referencing while a spoken reference can consist of one or more point gestures and a simple demonstative pronoun.
- System outputs maintain format consistency within and across displays, and also provide for contextual orientation across all displays throughout the user-computer dialogue.

- CUBRICON is a knowledge-based system. Input understanding and output composition considers dialog context (i.e., what is currently being displayed and has recently been expressed), task context (i.e., the importance of information relative to the ongoing task), and information context (i.e., the nature of the information vis-a-vis human sensing and understanding capabilities).

1.3 Implementation and Status

The CUBRICON system is implemented on a Symbolics Lisp Machine with a mouse pointing device, a color-graphics monitor, and a monochrome monitor. Speech recognition is handled by a Dragon Systems VoiceScribe 1000. Speech output is produced by a DECtalk speech production system. CUBRICON software is implemented using the SNePS semantic network processing system [Shapiro79a; Shapiro86; Shapiro89], an ATN parser/generator [Shapiro82a], and *Common Lisp*. SNePS is a fully intentional propositional semantic network and has been used for a variety of purposes and applications [Maida85; Shapiro86; Neal86, Neal87]. SNePS provides: (a) a flexible knowledge representation facility in the semantic network formalism; (b) representation of rules in the network in a declarative form so they can be reasoned about like any other data; (c) a bi-directional inference subsystem

[Shapiro82b] which focuses attention towards the active processes and cuts down the fan-out of pure forward or backward chaining; (d) a simulated multi-processing control structure [McKay80]; (e) special non-standard connectives [Shapiro79b] to model human reasoning processes; and (f) existential, universal, and numerical quantifiers [Shapiro79c].

CUBRICON is a proof-of-concept system. It integrates multiple-media input and output, and provides knowledge-based understanding and generation of human-computer communication including natural language, pointing/highlighting, and form-based interface technologies. As a proof-of-concept system, it performs the functions of multi-media human-computer communication (see Section 1.2), but not at the level of sophistication that would be expected of a final system for operational users. For example, speech input uses a discrete voice recognition system rather than a more expensive continuous voice recognition system. The CUBRICON grammar and lexicon have been developed to support the present proof-of-concept application. It does not provide for the processing of all possible English language structures and terms. Finally, CUBRICON has been developed in the lab. It does not use a hardware and software design that would permit extremely fast and efficient processing of inputs and outputs (i.e., CUBRICON dialogue is slower than human to human dialogue). Rather, it was designed for efficient development and evaluation of the technology itself. Improvements in speed and naturalness could be made in future CUBRICON implementations.

Finally, the focus of the CUBRICON implementation has been to develop a proof-of-concept knowledge-based integrated multi-modal human-computer interface system, which can be implemented efficiently as a *front-end to a variety of application systems*. State-of-the-art technologies have been applied and developed to achieve this end. The emphasis has been on the application of artificial intelligence based technologies to the human-computer interface problem. Only peripheral efforts have been expended in the application of more routine or standard human-computer interface techniques. For example, significant effort has been applied to achieve understanding of simultaneous spoken natural language and pointing inputs, while little effort has been made to provide menu-based alternatives. While a final version of CUBRICON would include such standard communication techniques in its suite of modalities for human-computer communication, the emphasis in this effort has been to design and develop a multi-modal system that can use a variety of state-of-the-art technologies in an intelligent, highly integrated manner for human-computer communication.

1.4 Organization Of The Report

This report describes the research efforts conducted during this project and presents the results of the evaluation which attempted to measure how well the above goals (see Section 1.1) were achieved. A brief overview of the organization of this report is given below:

Report Section: Summary of Contents:

1 Introduction. Provides an introduction to and functional overview of the Intelligent Integrated Multi-Media Interfaces Project and of the CUBRICON system.

2 Review of background and related research.

3 Overview of System Design. Contains an overview description of the CUBRICON design including a brief description of each major system component.

4 Knowledge sources. Contains a discussion of the major CUBRICON knowledge sources that support intelligent decision-making.

5 Multi-modal Input. This section discusses the major CUBRICON multi-modal input processing components.

6 Executor. Discusses the CUBRICON executor component which executes the actions intended by the user.

7 Multi-modal output. This section discusses the top-level CUBRICON multi-modal output planning component.

8-12 Multi-modal output. These sections describe major CUBRICON output technologies and modalities.

13 This section discusses the CUBRICON intelligent window manager.

14 KB Builder Tool. Describes a CUBRICON Tool which can be used to develop knowledge bases from relational databases.

15 Evaluation. Describes the CUBRICON evaluation that was conducted during this effort and summarizes the results.

16 Future Directions. Recommends future directions for the CUBRICON system and related research.

17 Summary. Provides a summary of CUBRICON and the research that was accomplished during this effort.

18 References. Contains a complete list of all references made within this document and its appendices (excepting Appendix H).

Appendixes

A Example CUBRICON Dialogue. Contains examples of user-CUBRICON dialogue which illustrate important CUBRICON features. These sample dialogues are illustrated with pictures of actual CUBRICON displays.

B Grammar and Lexicon. Contains a description of the grammar and lexicon used by CUBRICON.

C Graphic presentation of representative knowledge base structures from the CUBRICON KB.

D References to Published Technical Papers Describing CUBRICON and the Research Conducted Under the Intelligent Multi-Media Interfaces Program.

E Evaluation Training Material and Results. Contains: a complete set of material used to train subjects for the CUBRICON evaluation; all work aids used during the evaluation; and data generated during the evaluation.

F Working Paper on Computer Speech Generation. Contains a working which presents the results of a literature review on human factors issues relating to the use of computer generated speech. This paper was delivered to DARPA and RADC earlier in the program and is included here for completeness.

2 Related Research

With the advent and increased availability of high-quality reliable interface hardware such as fast high-resolution monochrome and color graphics display systems, speech recognition systems, speech production systems, and pointing devices, it is a natural and timely step in the evolution of human-computer interfaces that the media be integrated to meet the information processing needs of the user community. Research and development of artificial intelligence systems for man/machine interfaces have previously focused on natural language text, speech recognition, speech generation, and graphics primarily in isolation, rather than in integrated interfaces. The results of these various efforts are now beginning to converge in research and development of multi-modal human-computer interface technology.

Computer-based multi-media communication between people has received support via the development of multi-media electronic document systems and mail systems. This research and development includes the multimedia electronic document systems at Brown University [Feiner82], an experimental multimedia mail system at ISI [Katz84], a multimedia message system at SRI [Aceves84], and the Diamond message system at BBN [Thomas85]. Hypertext [Conklin87; CACM88] provides for multi-modal multi-dimensional document representation and access.

Intelligent interactive human-computer dialogue via multi-media language (e.g., simultaneous natural language and graphics) has more recently begun to develop. Work has begun on intelligence in interfaces [Neches86; Sullivan88] and, in particular, on the issue of the intelligent use of multiple media and/or modalities for human-computer communication [Hollan88; Neal88a; Neal88b; Neal88c; Neal89a; Neal89b; Arens88; Roth88; Kobsa86; Reithinger87].

Key features of the CUBRICON design, discussed in this paper, include the integration of NL and graphics in a unified language that is defined by a multi-modal grammar and the generation of synchronized speech and graphics in real time. The integration of NL and graphics in a unified language distinguishes this research from other approaches to multi-modal interface technology [Sullivan88, Arens89]. The Integrated Interface system [Arens88] and the XTRA system [Kobsa86, Allgayer89] are two of the most relevant. The Integrated Interface system is a multi-modal system in that it uses both maps and NL for the presentation of information to the user. The system provides information about the status and movements of naval platforms and groups in the Pacific Ocean. The system displays NL in text boxes positioned on a map display near the relevant objects. The system does not use a multi-modal language, however. The language generated is purely NL with no integrated graphics. The XTRA system is a multi-modal interface system which accepts and generates NL with accompanying point gestures for input and output, respectively. In contrast to the XTRA system, however, CUBRICON supports a greater number of different types of pointing gestures and does not restrict the user to pointing at form slots alone, but enables the user to point at a variety of objects such as windows, table entries, icons on maps, and geometric points. In added contrast to XTRA, CUBRICON provides for multiple point gestures per NL phrase and multiple point-accompanied phrases per sentence during both user input and system-generated output. CUBRICON also includes graphic gestures (i.e., certain types of simple drawing) as part of its multi-modal language, in addition to pointing gestures. Furthermore, CUBRICON addresses the problem of temporally coordinating NL (speech) and graphic gestures during both input and output.

The CUBRICON project has also addressed the problem of having the system select the media/modalities for expressing information to the user as well as composing the output in the selected media/modalities. Related work includes that of Reithinger [Reithinger87] in generating referring expressions and pointing gestures. The Integrated Interfaces project [Arens88] uses a variety of output modalities, but does not include speech production or deictic pointing gestures during output. The CUBRICON project and the work of Roth et al. [Roth88] both are concerned with the problems of selecting relevant information to present to the user, composing text and selecting and designing pictures to convey the information, and the problems of coordinating the two different modalities. CUBRICON, however, is concerned with more output modalities (e.g., speech).

The CUBRICON system includes several knowledge sources (e.g., application-specific knowledge base, discourse model, user model) in order to generate relevant helpful responses, maintain the discourse context when appropriate, manage its display resources, provide the user and system with the ability to reference the display objects, and use the modalities *in coordinated combinations* for output generation. Cheikes and Webber [Cheikes88] and Kaplan [Kaplan82] address the problem of generating relevant cooperative responses. The issue of models to support intelligent behavior of interface systems is also addressed by Wahlster [Wahlster88], Mason and Edwards [Mason88], Kass and Finin [Kass88], and Young [Young88]. *Computational Linguistics* Vol. 14 No. 3 [Kobsa88] focuses on user modeling.

3 Overview of System Design

The CUBRICON team has designed and implemented an integrated user interface system with the functionality described briefly in Section 1.2. Figure 3-1 provides an overview of the software system and hardware I/O devices currently supported by CUBRICON.

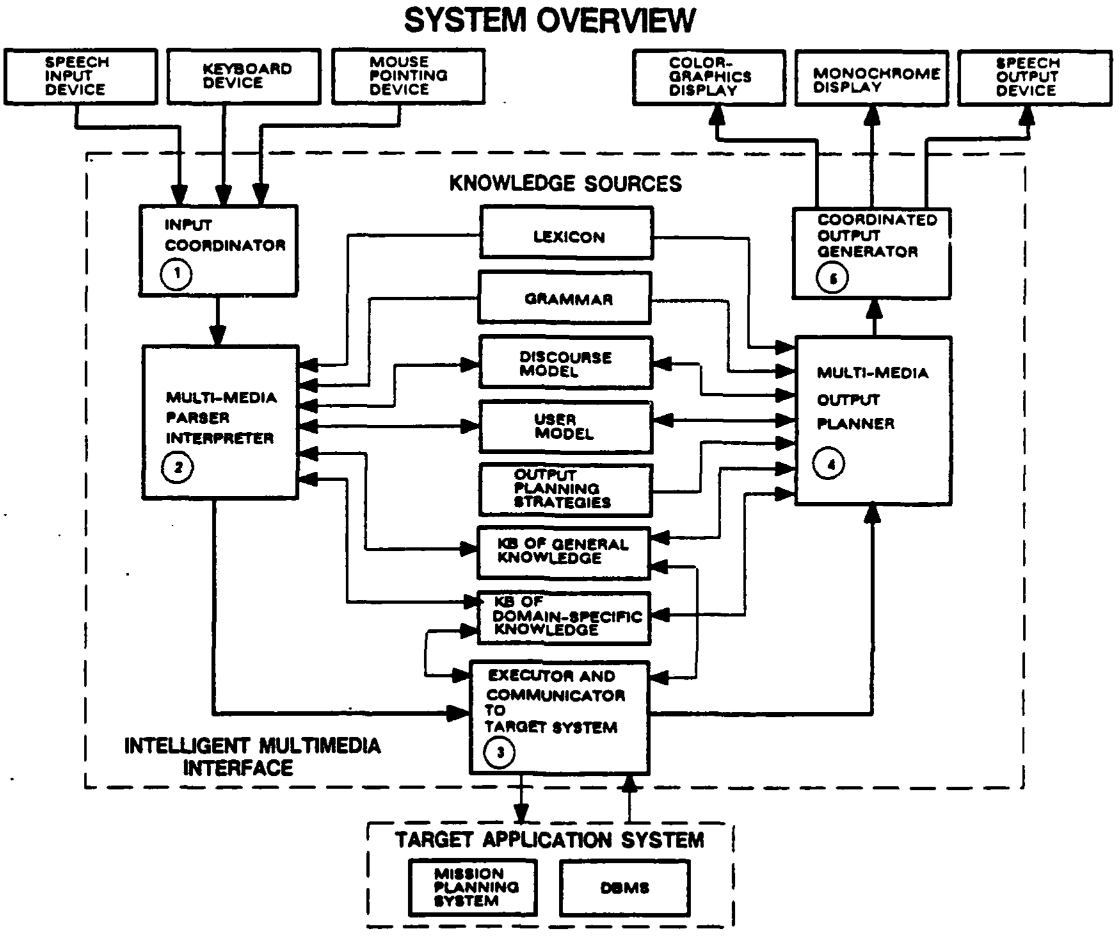


Figure 3-1: System Overview

CUBRICON accepts input from three input devices: a speech recognition system, a keyboard, and a mouse. CUBRICON produces output via three output devices: a high-resolution color-graphics display, a monochrome display, and a speech output device.

The primary data processing flow through CUBRICON is indicated by the numbered modules in Figure 3-1. These are: (1) Input Coordinator, (2) Multi-med' Parser Interpreter, (3) Executor/Communicator to Target System, (4) Multi-media Output Planner, and (5) the Coordinated Output Generator. Each of these are briefly described in the following paragraphs.

Inputs to CUBRICON are handled by the Input Coordinator and the Multi-Media Parser Interpreter. The Input Coordinator module accepts input from the three input devices and fuses the input streams into a single compound stream, maintaining the temporal order of tokens in the original input stream. The Multi-media Parser/Interpreter is an augmented transition network (ATN) that has been extended to: 1) accept the compound stream produced by the Input Coordinator and 2) produce an interpretation of this compound stream.

Once inputs are received and understood by CUBRICON appropriate action is then taken by the Executor module. This action may be a command or database query to the underlying application (e.g., a mission planning system, a database), or an action that entails participation of the interface system only.

An expression of the results of CUBRICON action (completed by the Executer) are planned by the Multi-Media Output Planner for communication to the user. The Multi-Media Output Planner is a generalized ATN that produces a multi-media output stream representation, with components targeted for different devices (e.g., color-graphics display, speech output device, monochrome display). This output stream representation is translated into visual/auditory output by the Coordinated Output Generator module. This module is responsible for producing the multi-media output in a coordinated manner in real time. For example, the Multi-Media Output Planner module may specify that a certain icon on the color-graphics display must be highlighted when the entity represented by the icon is mentioned in the simultaneous natural language voice output. The Coordinated Output Generator implements this coordinated output.

The CUBRICON system incorporates several knowledge sources that are used during processing. The knowledge sources currently include: (1) a lexicon, (2) a grammar defining the language used by the system for multi-media input and output, (3) a discourse model, (4) a user model, (5) a knowledge base of output planning strategies to govern the composition of multi-media responses to the user, (6) a knowledge base of information about generally shared world knowledge, and (7) a knowledge base of information about the specific task domain of tactical air control. These knowledge sources are used for both understanding input to the system and planning/generating output from the system. They are discussed in more detail in the next section.

In its entirety, the CUBRICON system provides an integrated multi-media human-computer interface system which can be implemented as a front-end to a target application system. Inputs are accepted via a combination of input modalities. Outputs are accepted via a combination of output modalities. CUBRICON is designed in a way that allows it to be applied to a variety of application systems with only minimal programming efforts. It also is configured to accept the incorporation of additional input and output modalities to support future interface needs.

4 Knowledge Sources

The CUBRICON system includes several knowledge sources for use in multi-media language understanding and production. These knowledge sources are: a lexicon, grammar, discourse model, user model, and a knowledge base of information about the task domain of tactical air control and related interface information.

This section contains: a description of the knowledge base relating to the task domain of tactical air control and related display information; the discourse model; and the user model. Descriptions of the other knowledge sources are contained within other sections of this report. Specifically, the lexicon and grammar are described in Sections 5 and 8, and in Appendix A.

4.1 Knowledge Base

The CUBRICON Knowledge Base (KB) is central to the CUBRICON system. One of the key features of CUBRICON is that it is a unified system in which various displays and presentations reflect a single integrated underlying reality. The KB provides the representation for this "unified underlying reality" and the SNePS semantic network processing system [Shapiro79a, Shapiro89] provides the representational formalism for this KB. A key feature of the SNePS knowledge base representation is that each conceptual object has a unique representation in the SNePS semantic network. Thus, whenever different CUBRICON components (e.g., parser, interpreter, natural language generator) use the SNePS KB representation of a given conceptual object (e.g., the Dresden Airbase), they are all using the *same KB representation*. In other words, all components use the same unique KB node representing an object such as the Dresden Airbase and there are no inconsistencies with regard to knowledge available and used by the different CUBRICON components. That is, all the CUBRICON components "speak the same language".

The CUBRICON knowledge base contains domain-specific information concerning the particular task domain of the application system to which CUBRICON is serving as the human interface as well as interface information concerning the presentation or expression of the task domain entities or concepts. For example, this knowledge base includes information about domain specific entities such as airbases, surface-to-air missile systems (SAMs), and fuel storage facilities as well as information about how these objects should be expressed via verbal/graphic output. This includes words and symbols that can be used to express the entities or concepts.

As indicated in Section 1, the current CUBRICON application domain is that of tactical air control and mission planning. Therefore, the CUBRICON knowledge base includes information about concepts that relate to this domain. This information can be categorized into information about:

- tangible objects such as different types of air bases, SAM systems, plants, factories, radars, runways, fuel storage facilities, cities, heliports, tank battalions, infantry divisions, and aircraft-pools. and their properties, parts, and components.
- intangible objects or concepts such as different types of AF tactical mission plans (e.g., Offensive Counter Air, Refueling, Service, Target Strike) and instances of these mission plan types as well as their components and characteristics.

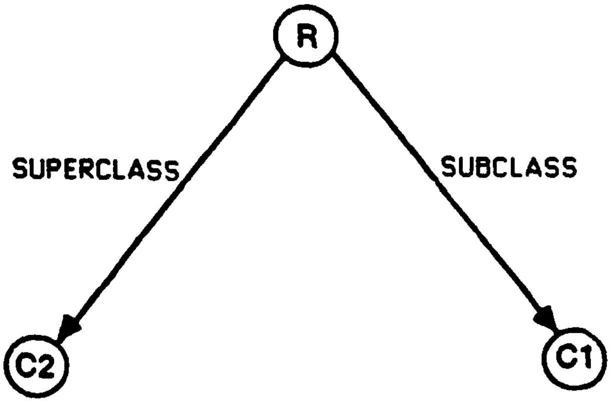
Much of the CUBRICON application domain information has been drawn from the AMPS DataBase developed by The MITRE Corporation for RADC. The AMPS data base was designed to support the planning of Air Force Air Tasking Orders. The CUBRICON KB Builder Tool that interfaces between the CUBRICON knowledge base and a relational database such as the AMPS database is discussed in Section 14.

The CUBRICON knowledge base is implemented using the SNePS semantic network processing system [Shapiro79a, Shapiro89]. A SNePS semantic network is a directed graph with labeled arcs in which nodes represent concepts and the arcs represent nonconceptual binary relations between concepts. A concept is something in our domain of interest about which we want to store information and which may be the subject of "thought" and inference. The arcs of the network are not conceptual, but structural [Shapiro79a, Shapiro86].

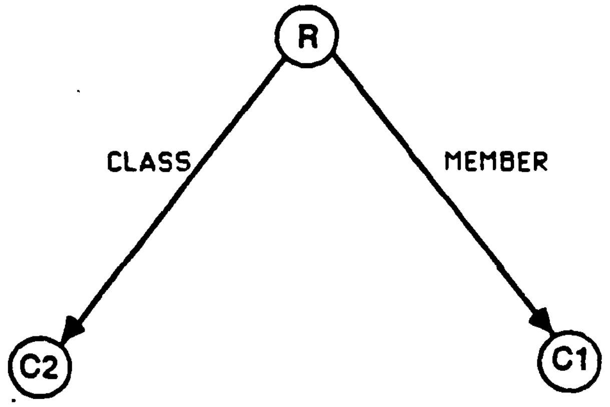
The primary type of arc in a SNePS network is the descending arc and if there is a path of descending arcs from node N to node M, then N is said to dominate M. Two important types of nodes are molecular and atomic. Molecular nodes are nodes that dominate other nodes and atomic nodes are simply not molecular. Molecular nodes represent propositions and atomic nodes represent objects.

The more important types of propositions or relations that we have used in the CUBRICON knowledge base are listed below. In each case, we have displayed the corresponding SNePS structure. For each SNePS structure, it is the top molecular node that represents the proposition or relation. represents

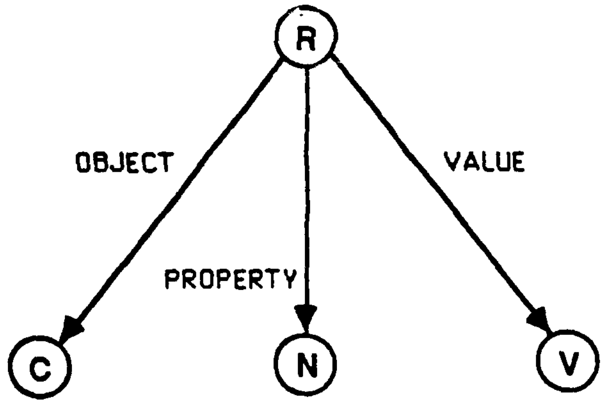
- Cl is a subclass or subtype of C2. For example, fighter bases are a subclass or subtype of airbases.



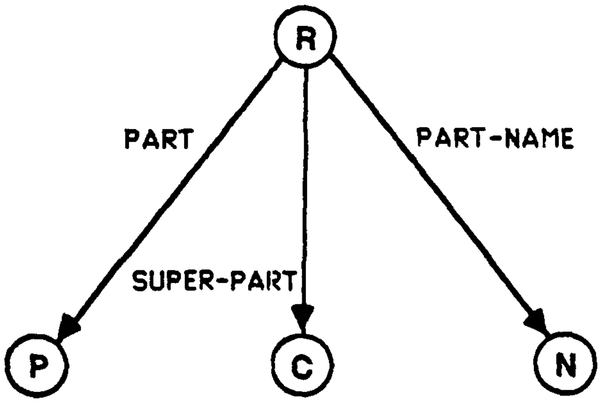
- C1 is a member of class C2. For example, Erfurt Airbase is a member of the class of fighter bases.



- Object C has a property called N with value V. For example, the object Erfurt Airbase has a property, called nationality, which has value GDR.

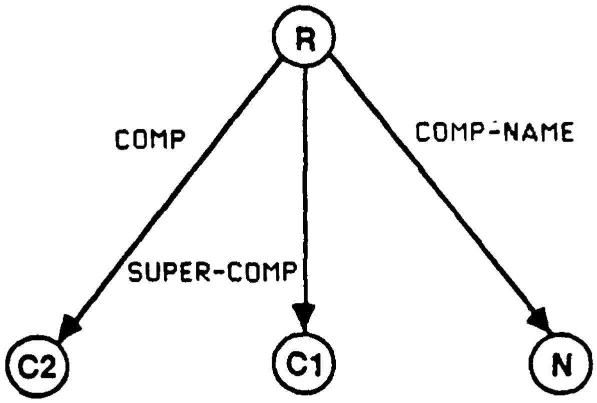


- Object C has a part P and the part-of relation is called N. This can also be stated as: P is an N kind of part of C. Object C is called the superpart of P. This relation specifically refers to a part-superpart relation in which the part P is inseparable from the superpart C. For example, a wing of an aircraft is an inseparable part of the aircraft. More specifically, the 17-35-Erfurt runway is an inseparable part of the Erfurt airbase. In this latter case, the value of N would be the name "runway" to specify the kind of part.

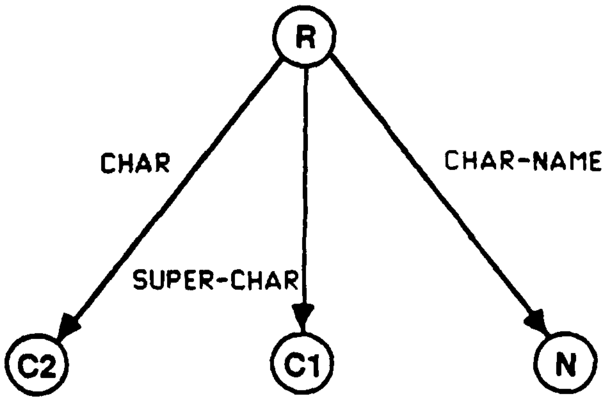


- Object C1 has a component C2 and the component relation is called N. Object C1 is called the supercomponent.

By a component, we mean an object that is physically separable from the supercomponent. For example, a specific unit, called the 435th Tactical Fighter Wing (435TFW), is a component of the Nuernberg airbase since it is physically separable from the airbase. The values of C1, C2, and N would be the concept of the Nuernberg airbase, the concept of the 435TFW, and the name "Unit" (as the name of the component-supercomponent relation), respectively.



- Intangible object C1 has a characteristic C2 and this characteristic-relation is called N. C1 is called the super-char of C2. This relation is only used for intangible objects such as mission plan types and instances. For example, a refueling service mission is represented as a child-task (sub-mission) of an offensive counter air (OCA) mission using the characteristic-relation. That is, the structure would have the concept of the OCA mission as the value of C1, the concept of the refueling service mission as the value of C2, and the name "child task" as the value of N using the Char-Superchar-Charname structure.



For any object or concept, the knowledge base will typically contain a considerable amount of information that is known about it and that is attached to its representation via the semantic network structures. For example, Figure 4-1 shows the node representing the Merseberg Airbase in the center. This figure also shows the knowledge base representation of some of the information that is known about this airbase. Each node of the figure is, in actuality, assigned a SNePS-generated label, not shown in the figure for purposes of reducing clutter in the figure. Each of the molecular nodes in the figure are instantiations of the case frame structures discussed above. For example, the node labeled M represents the proposition that the airbase is of enemy disposition. The disposition is represented as a property of

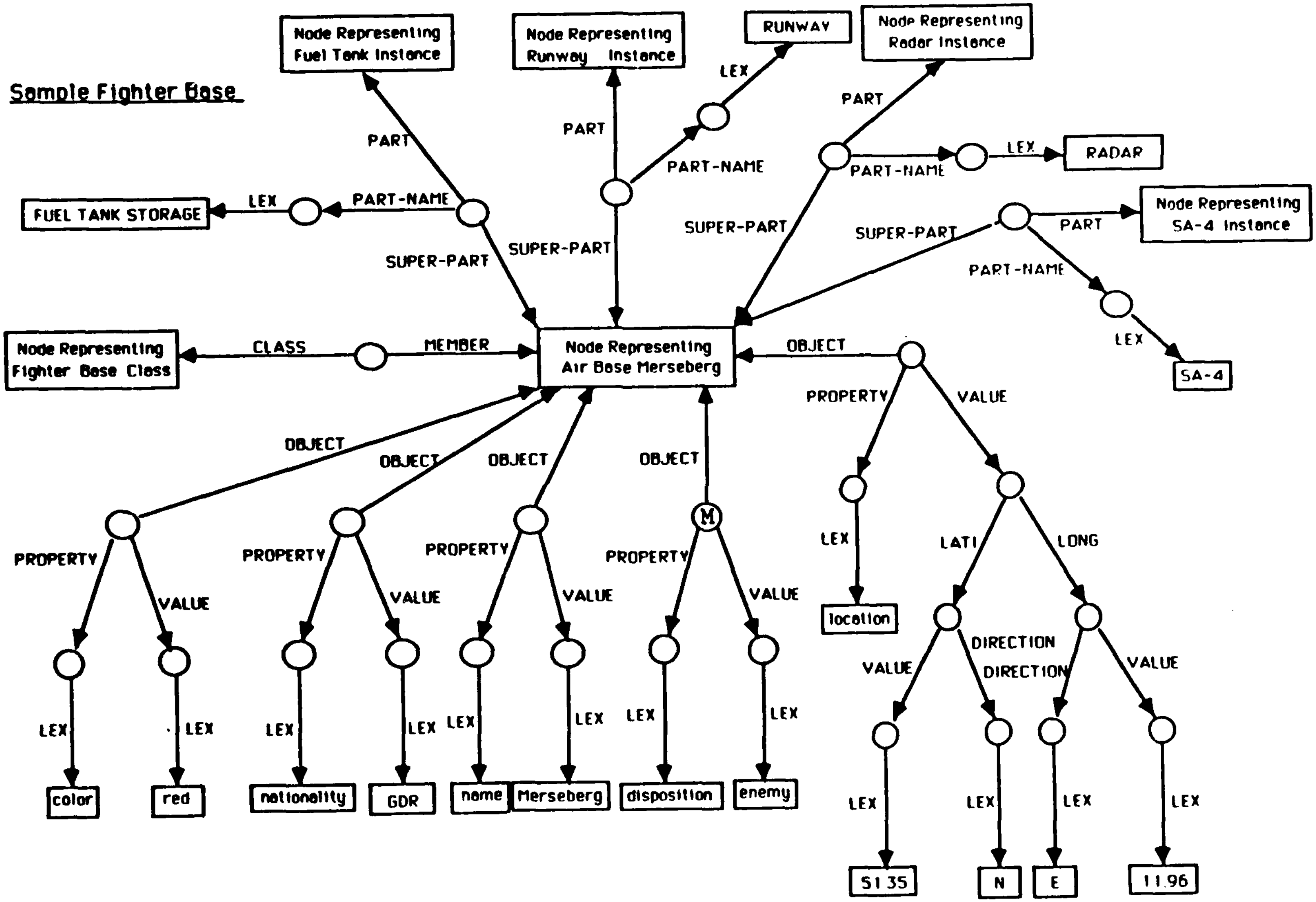

Sample Fighter Base
Node Representing Fuel Tank Instance
Node Representing Runway Instance
RUNWAY
Node Representing Radar Instance
RADAR
Node Representing SA-4 Instance
SA-4
FUEL TANK STORAGE
Node Representing Fighter Base Class
Node Representing Air Base Merseberg
PART
PART-NAME
SUPER-PART
LEX
CLASS
MEMBER
OBJECT
PROPERTY
VALUE
M
LATI
LONG
DIRECTION
location
color
red
nationality
GDR
name
Merseberg
disposition
enemy
51.35
N
E
11.96

Figure 4-1: Examples of Semantic Network KB Representations

the airbase. Each atomic or base node is represented by a rectangular shape in the figure. For each of the concepts represented by one of these atomic nodes, again, much information is represented in the knowledge base and there are many network propositional structures that involve these atomic nodes that are not shown in the figure. For the radar instance, for example, network structures are present in the knowledge base representing its location, disposition, type, etc.

Appendix C contains additional figures that illustrate some of the primary types of knowledge represented in the CUBRICON knowledge base.

4.2 Discourse Model

The CUBRICON discourse model is a dynamic model of the attentional dialogue focus space [Grosz78, Grosz86; Sidner83; Grosz85]. This model serves two primary purposes in the CUBRICON system: to maintain continuity and judge relevance. Without the key factors of continuity and relevance, people find discourse disconcerting and unnatural. The discourse model is used to determine the interpretation of dual-media references, determine the interpretation of anaphoric references [Sidner83] and definite references [Grosz81] that are input by the user in natural language, decide when and how to use pronouns when generating natural language output, and decide when presentation objects should be removed from the displays depending on the function that they serve.

The CUBRICON discourse model consists of four structures: (1) the Main Focus List; (2) the Display Model; (3) the Presentation Object Data Structure; and (4) the Form Model. Each of these Discourse Model components is discussed in this section.

4.2.1 Main Focus List

The Main Focus List is CUBRICON's primary means of tracking the attentional discourse focus space. It consists of a continually updated list of those entities and propositions that have been explicitly expressed (by the user or by CUBRICON) via natural language, pointing, highlighting, or blinking. The Main Focus List maintains a temporal record of when and how the various objects or concepts were referenced and is used by CUBRICON (1) in determining the referents of pronouns and definite noun phrases spoken by the user and (2) in generating pronouns in natural language output.

For each multi-modal sentence uttered by the user or by CUBRICON, a representation of each of the referenced propositions and concepts is added to the Main Focus List. Each entry to the Focus List takes the form of a quintuplet. The components of the quintuplet are:

- the knowledge base representation of the object or proposition referenced,
- the part of speech used to express the object or proposition,
- the gender of the object, if any,
- the linguistic number of the object, if any,
- a weight assigned as a representation of the object's relative importance within the utterance.

Each time a new sentence is processed and new objects are added to the Focus List, the weights of the objects already on the Focus List are reduced by a certain factor, calied the *fade factor*. Objects are kept on the Focus List as long as their weight is above a certain *threshold.* Therefore, objects fade off the Focus List over time, being kept on the Focus List for approximately ten dialogue interactions. The fade factor and threshold can be set/changed by the system developer. Additional details concerning the management of the Focus List are available in a technical report by N. Li [Li87].

4.2.2 Display Model

The display model represents all the objects that are "in focus" because they are visible on one of the monitors. Graphics are an integral part of CUBRICON's language along with natural language and other forms of language and pointing. The CUBRICON system treats objects presented on the graphics displays as having been intentionally "expressed" or "mentioned". All objects on the graphics display are therefore "in focus" and CUBRICON maintains a representation of all these objects in the form of a display model. The display model is defined at two levels: (1) a continually updated list of all the displayed windows on each monitor and, (2) for each window, a continually updated list of all the objects that are visible within it.

The window list for each monitor includes the following information for each window:

- The Lisp window object identifier including its status (e.g., exposed, active).
- The SNePS knowledge base representation of the window object.
- The ordinal number corresponding to the dialogue cycle at which the window was created.
- The Lisp window identifier of any associated windows (e.g., table window associated with a map window, or vice versa).

For each window, the content list includes the list of objects visible in the window. For a map type window, the following information is available via the content list for each object:

- The SNePS knowledge base representation of the conceptual object that the presentation object (icon) represents.
- The Lisp identifier of the presentation object (icon).
- Information about the icon such as the coordinates of its extent and its color, form, and size.
- Information concerning any labels or text boxes associated with the icon.
- Information concerning any type of highlighting of the icon.

Similar content lists are associated with table windows and the form window.

The Display Model is used by CUBRICON during both input interpretation and output composition. During input processing, it plays a critical role in the interpretation of user inpu'- that include point gestures. During the interpretation process, CUBRICON determines which icons are "touched" by a point gesture or mouse click on a given window. In so doing, CUBRICON selects those icons for which the coordinates of the mouse click fall within the extent of the icon. The Display Model contains, for each window, the list of icons visible in the window and, for each icon, a list of corresponding information including the conceptual object that the icon portrays. This conceptual object is represented in its SNePS semantic network knowledge base form. (Remember that the SNePS knowledge base representations constitute the unifying language in which the domain knowledge is represented for all of the CUBRICON components. Each conceptual object has a unique representation in the SNePS semantic network.) Thus the Display Model serves as CUBRICON's mechanism for mapping from display icons to the conceptual objects represented by the icons. If the mouse point gesture was used in conjunction with natural language input, then the results obtained by mapping icons into the semantic network representation of the conceptual objects that they represent would undergo further processing by the multi-modal parser interpreter as the rest of the input sentence is processed. The processing of multi-modal inputs is discussed in more detail in Section 5.

The Display Model is used by CUBRICON in the determination of how to express new outputs to the user. All display updates are generated based on the pre-existing display context, represented by the Display Model. Display updates are designed to build upon the pre-existing display context in a way that minimizes display (and dialogue) disruption and maximizes display (and dialogue) continuity. For example, when composing a new map display to show new entities (e.g., airbases) that were not previously displayed, if the pre-existing display content is still relevant based on CUBRICON's representation of the user's task, CUBRICON will compose a new map display that includes both the pre-exiting map contents as well as the new entities to be displayed. As indicated previously, CUBRICON does this to maintain context by preserving the display of task-relevant objects and entities. This map composition process is discussed in more detail in Section 10.

Another example of CUBRICON's use of the display model involves its composition of object references when composing natural language output. If CUBRICON is composing a natural language response to the user, CUBRICON consults the display model to determine whether the object is visible in one or more of the display windows. If so, CUBRICON uses a deictic dual-media expression to refer to the object in the output sentence. A deictic dual-media expression might consist of a phrase such as "this airbase" with simultaneous blinking/highlighting of the airbase icon as CUBRICON's means of pointing to it. If the entity is the most salient of its gender according to the main focus list, CUBRICON may use a pronoun as the verbal part of the expression. The Display Model plays a central role in this process, since it is the source of current up-to-date knowledge as to what is on the display windows at any moment in time.

Although not yet implemented, the CUBRICON design calls for the use of the Display Model in interpreting definite references along with the use of the Focus List and knowledge base. That is, when a person expresses a definite reference such as "the airbase" with just one such object in view (as on a graphics display), then CUBRICON should first consult the Focus List for the most salient object of the required type. If none has been previously discussed and therefore none is found on the Focus List, then CUBRICON should conclude that the one in visual focus (represented in the display model) is the one being referred to, even though several others may be contained in the knowledge base. If many airbases are currently displayed in this situation, CUBRICON might select the airbase most relevant to the user's task (e.g., only friendly airbases would be selected as an origin for a strike mission). If no disambiguating information at all were available, it might respond with the question, "Which airbase do you mean? "

4.2.3 Presentation Object Data Structure

The Presentation Object Data Structure (PODS) supports continuity and consistency in the human-computer dialogue and supports the de-cluttering of map displays. More specifically, the PODS is used to determine which presentation objects to regenerate when a map is redisplayed by zooming in or zooming out to maintain continuity and consistency in map displays. Secondly, it is used to determine when and how to remove presentation objects from any display window.

A presentation object is an output mode of expression, such as a highlighted icon or a dynamic text window. The PODS organizes the presentation objects by the function they serve within the CUBRICON system. This is necessary since the same presentation object is treated differently within CUBRICON depending on why it was created. The organization of the PODS is that of a tree structure. The PODS is continually updated so that it always contains representations of all the presentation objects on the displays at any given time.

4.2.3.1 Presentation Objects.

CUBRICON does not record every output mode of expression in the PODS. The PODS records presentation objects which are dynamic in nature, appearing and disappearing based on recency of creation. Table entries, for example, are static in nature. They are never removed from a table once they are included in a table. Therefore, these presentation objects are not included in the PODS. The following is a list of the presentation objects included in the PODS: icons which mark the location of mouse points, string labels, the highlighting of windows, the highlighting of table entries, the highlighting of icons, pointing text windows, dynamic windows, flight paths, icons that are part of a flight path presentation, and context boxes which show the relationship between the previous and currently displayed map area.

4.2.3.2 Functionality Types.

The PODS is organized primarily according to the functions that the presentation objects serve within the system. Presentation objects serving different functions are handled differently by CUBRICON. For example, the same type of label, placed on a map, is used to identify the order in which point gestures occurred as well as to identify the property and property-value of an entity. These presentations are treated differently: labels marking the locations at which pointing gestures "touch" the window are removed prior to output generation for the current dialogue cycle, whereas a label identifying a property-value pair for any entity is removed after fifteen dialogue cycles.

As indicated previously, one of the purposes that the PODS serves is to enable CUBRICON to remove certain ancillary presentation objects from its displays in an appropriate manner so that the displays do not become too cluttered. Presentation objects are removed from a display based on the functionality and recency of creation of the presentation object.

The following list contains the various functions that a presentation object can serve. The list also includes the time at which the particular functional class type of presentation object would be removed from the display.

- Location Mark for User's Pointing Gesture

  The presentation object which marks the location at which a user's pointing gesture "touches" a window. This includes a icon indicating the location of the mouse point, a pointing arrow indicating the icon referred to if the mouse point was "off" by some distance (the system would infer which object was the intended referent of the point gesture), and a label indicating the numerical order in which the mouse points occurred. This type of presentation object is displayed when the user points via the mouse device.

It is removed during the same dialogue cycle, after CUBRICON generates a response to the user.

- Non-window Object Pointing

  Deictic gestures generated by CUBRICON which point to objects visible within a CUBRICON window. This includes icon highlighting, table entry highlighting, mission planning form highlighting, and a pointing text box which points to a map icon. This type of presentation object is removed during the next dialogue cycle following its creation, at the beginning of output generation.

- Window Pointing

  The deictic gesture generated by CUBRICON in order to point at a window. This consists of highlighting the window frame of the particular window. This type of presentation object is removed during the next dialogue cycle following its creation, at the beginning of output generation.

- Map Context Box

  This presentation object consists of a rectangular box outlining a particular geographical region within a current map window. It is frequently used to outline the previously displayed geographical region when a map window undergoes a "zoom out" transformation. Thus it is used to show the new geographical region in the context of the old. This type of presentatior object is removed during the second dialogue cycle following its creation, at the beginning of output generation.

- Property Label

  The label which expresses a property and its value for an entity which is currently displayed as a map icon. The label is located near the map icon. This type of presentation object is removed during the fifteenth dialogue cycle following its creation, at the beginning of output generation.

- Mission Presentation

  The presentation objects which were created during a mission presentation. This includes map icons, the highlighting of map icons, the highlighting of table entries, the highlighting of form entries, dynamic text windows, and flight paths. This type of presentation object is removed when requested by the user or by CUBRICON when new missions are to be presented.

4.2.3.3 Data Structure Format. The PODS is a tree structure including the following information:

```
((<functionality> <recency> <key>
  ((<window identifier>
    ((<PO type> <node list> <optional argument>)* )
   )* )
 )* )
```

Figure 4-2: PODS Structure Diagram

- Functionality

  The functionality of the presentation object as described in Section 4.2.3.2.

- Recency

  Time the presentation object was created. This item is used along with the functionality of the presentation object to determine when to remove the presentation.

- Key

  Key used to differentiate presentation objects with the same functionality. For example. presentation objects created during different mission presentations.

- Window Identifier

  The identifier of the window instance containing the presentation object.

- PO Type

  The type of presentation object, used to determine the function call and arguments needed to remove or regenerate the presentation object. Section4.2.3.1 identifies these presentation objects.

- Node List

  List of SNePS node(s) for which the presentation was performed. This field is needed to remove and regenerate presentation objects which refer to a map icon (e.g., map icon highlighting).

- Optional Arguments

  Any additional arguments needed to regenerate the presentation object. For example, the contents of a string label is stored as an optional argument.

The structure of the PODS is shown in Figure 4-2. The PODS includes three association lists with keys of functionality, window identifier, and PO type respectively. A sample PODS is shown in Figure 4-3. The first list, which contains the functionality keyword

```
((:PROPERTY-LABEL 3 NIL
  ((#<GUIDE-WINDOW Guide Window 5 11010157 exposed>
    ((:STRING-LABEL (B25) ("mobility: high"))
     (:STRING-LABEL (B15 B18) ("mobility: low"))))
   (#<GUIDE-WINDOW Guide Window 6 11010665 exposed>
    ((:STRING-LABEL (B25) ("mobility: high"))
     (:STRING-LABEL (B15 B18) ("mobility: low"))))))
 (:MISSION 5 "OCA345"
  ((#<GUIDE-WINDOW Guide Window 5 11010157 exposed>
    ((:HIGHLIGHTED-ICON (B40) :CIRCLE)
     (:STRING-LABEL (B40 B171) ("Origin Air Base."))
     (:FLIGHT-PATH (B40 B172) NIL)
     (:STRING-LABEL (B172) ("5:55")) (:FLIGHT-PATH (B172 B173) NIL)
     (:STRING-LABEL (B173) ("6:15")) (:FLIGHT-PATH (B173 B174) NIL)
     (:STRING-LABEL (B174) ("6:30")) (:FLIGHT-PATH (B174 B175) NIL)
     (:STRING-LABEL (B175) ("6:45")) (:FLIGHT-PATH (B175 B176) NIL)
     (:STRING-LABEL (B176) ("6:50")) (:HIGHLIGHTED-ICON (B50) :EXPLODE)
     (:FLIGHT-PATH (B176 B177) NIL)  (:STRING-LABEL (B177)("7:00"))
     (:FLIGHT-PATH (B177 B178) NIL)  (:STRING-LABEL (B178) ("7:05"))
     (:FLIGHT-PATH (B178 B179) NIL)  (:STRING-LABEL (B179) ("7:15"))
     (:FLIGHT-PATH (B179 B180) NIL)  (:STRING-LABEL (B180) ("7:25"))
     (:ICON (B227) NIL)              (:FLIGHT-PATH (B180 B181) NIL)
     (:STRING-LABEL (B181) ("7:40")) (:FLIGHT-PATH (B181 B182) NIL)
     (:STRING-LABEL (B182) ("8:00")) (:FLIGHT-PATH (B182 B183) NIL)
     (:STRING-LABEL (B183) ("8:10")) (:FLIGHT-PATH (B183 B40) NIL)
     (:STRING-LABEL (B40 B184) ("Mission completed."))))
   (#<MISSION-WINDOW Form Window 11000744 deexposed>
    ((:HIGHLIGHTED-FORM (M1571!) NIL) (:HIGHLIGHTED-FORM (M1542!) NIL)
     (:HIGHLIGHTED-FORM (M1573!) NIL) (:HIGHLIGHTED-FORM (M1721!) NIL)
     (:HIGHLIGHTED-FORM (M1660! M1661!) NIL) (:HIGHLIGHTED-FORM (M1723!) NIL)
     (:HIGHLIGHTED-FORM (M1660! M1661!) NIL)
     (:HIGHLIGHTED-FORM (M1660! M1661!) NIL)))
   (#<TEXT TEXT WINDOW 11011373 deactivated> ((:DYNAMIC-WINDOW B221 NIL)))
   (#<TEXT-PRESENTATION-WINDOW Text Presentation Window 8 11010431 deactivated>
    ((:HIGHLIGHTED-TABLE-ENTRY (B40) NIL)))))
 (:POINT-AT 6 NIL
  ((#<TEXT-PRESENTATION-WINDOW Text Presentation Window 8 11010431 deactivated>
    ((:HIGHLIGHTED-TABLE-ENTRY (B47) NIL)))
   (#<GUIDE-WINDOW Guide Window 5 11010157 exposed>
    ((:TEXT-WINDOW (B47) ("dresden air base"))
     (:HIGHLIGHTED-ICON (B47) :CIRCLE)))))
```

Figure 4-3: Sample PODS

:PROPERTY-LABEL, was added to the PODS as a result of a request for the mobility of three sam systems. One of the presentation objects generated is a label associated with an icon represented by the SNePS node identifier B25. This label contains the string "Mobility High". Two additional labels were generated during this request. These labels are associated with the icons represented by the SNePS node identifiers B15 and B18, containing the string "Mobility Low".

The second list in the PODS, which contains the functionality keyword :MISSION, was added to the PODS as a result of a request to generate the OCA345 mission plan. There are numerous presentation objects generated when presenting a mission plan. First, the origin airbase is pointed to by highlighting and labeling, resulting in the addition of the first two PODS lists with keywords :HIGHLIGHTED-ICON and :STRING-LABEL. The origin airbase is represented in the knowledge base by the the SNePS node identifier B40. The flight path generated during a mission plan consists of waypoints connected by arrows, indicating the direction of the aircraft, and labels, indicating the time of arrival at each waypoint. The presentation objects comprising a flight path are represented in the lists containing the keywords :FLIGHT-PATH and :STRING-LABEL. The explosion of the target airbase added the list containing the keyword :HIGHLIGHTED-ICON to the PODS. An orbit occurring during the mission presentation added an icon to the map window and consequently added a list containing the keyword :ICON to the PODS. In addition, a flight path presentation generates a dynamic window containing text describing the mission plan (:DYNAMIC-WINDOW), the highlighting of relevant information on a mission planning form (:HIGHLIGHTED-FORM), the highlighting of relevant information on tables (:HIGHLIGHTED-TABLE-ENTRY), and a label indicating the mission presentation is complete (:STRING-LABEL).

The last list in the PODS was added as a result of the request for the location of the Dresden airbase. The result was the the highlighting of the table entry identifying the properties of the Dresden airbase (:HIGHLIGHTED-TABLE-ENTRY), a text-box containing the string "dresden air base" pointing to the icon representing the Dresden airbase (:TEXT-WINDOW) and the highlighting of the icon representing the Dresden airbase (:HIGHLIGHTED-ICON). The Dresden airbase is represented in the knowledge base by the SNePS node identifier B47.

4.2.4 Form Model

The Form is a display, titled the "Package Worksheet", to aid mission planners in constructing packages of OCA and other related mission plans. The Form display and the Form modality are discussed in Section 12. The Form Model consists of three data structures that are used by the Form component to maintain a visual display representation and conceptual knowledge base representation of the user's mission plans and packages as they are being developed. The access functions support user construction and modification of mission plans via the Form presentation window. Representations of many types of forms could be included within the Form Model, but in the current CUBRICON implementation, only one type of form is represented. The form can have many instantiations, however, in the current CUBRICON implementation.

The Form Model consists of three data structures:

- The Form Window Object. The Lisp window object whose visualization is the Package Worksheet display or Form. The window object contains information about how to display the form on the CRT (e.g., the size and location of each blank or slot of the display). The Form's visual slots or blanks are implemented as Lisp pane objects which collectively make up the window. The information in the Form Window Object enables CUBRICON to accept inputs via deictic gestures on the Form, and allows outputs to be displayed in their proper location on the Form. Slots in the form are related to corresponding slots in the larger Mission Template.

- The Informational Data Structure (IDS) keeps track of the information entered or displayed via the form. When information is entered on a form, for example, SNePS KB structures are created representing the information and they are stored within the IDS for the particular mission. These KB structures are created in accordance with the data relationships defined in the KB Mission Template (discussed below).

  The IDS consists of a list of Common Lisp structures, each corresponding to a mission package currently being planned. The one package displayed in the Form window is at the top of the list. The IDS is dynamic in that when a new package is started or a package is retrieved from the computer file system, a structure representing the package is added to the IDS. When a certain package is made the "current package" (the current package is the one displayed in the Form window), it is put at the top of the IDS list. For each mission package structure in the IDS, there is a one-to-one mapping between the slots in structure and the visual panes or blanks of the Form. Each slot in a package structure contains a list of six items. This list is called the form blank information list (FBIL). The elements of this list are:

  1. The Slot Filler Representation: The SNePS knowledge base node which best represents the value in the particular Form slot or blank. For example, if the entry in a slot representing the origin airbase for an OCA mission is the Nuernberg airbase, then this element of the FBIL would be the SNePS base (atomic) node representing the Nuernberg airbase.
  2. The Relation Between the Slot Filler and Mission Plan: The SNePS knowledge base propositional structure which represents the relation between the value (filler) of the particular Form slot (e.g., the node representing the Nuernberg airbase in the example above) and the particular mission plan (e.g., the particular OCA mission plan under development). For example, this knowledge base structure would represent the proposition that Nuernberg airbase (continuing with the example

from the previous paragraph) is the origin airbase for the OCA123 mission plan, if the OCA123 were the one being developed.

3. The Slot Representation: The SNePS node from the mission template (discussed below) which corresponds to the particular blank or pane on the mission Form window.
4. The Relation Generator: A Lisp form which, when evaluated, will build the knowledge base structure that is the second FBIL element listed above. The reason for this Lisp form is to be able to regenerate a mission package from a computer disk file.
5. The Slot Filler Generator: A Lisp form which, when evaluated, will return the knowledge base object listed as FBIL element number 1 above. The reason for this Lisp form is the same as for the Lisp form in number 4 above. The above Lisp form uses the Lisp form discussed in this item.
6. The Textual String: The string which is displayed in the Form window pane to express the concept which fills the particular slot or pane of the Form.

- The Mission Template is a SNePS knowledge base representation of the structure of a generic package of mission plans and subsidiary mission types. There is a one-to-one correspondence between structures within the Mission Template and the slots or panes of the Form window and the slots of a package structure of the IDS. A diagram of the mission template is provided in Appendix B.

  The mission template is used when parsing and interpreting natural language. When the concept of a mission component (corresponding to a form blank or slot) is mentioned, either via speech or a pointing gesture, the predicate node representing the mission component from the template is the resulting representation of the interpretation of the natural language phrase. The mission template provides information about the relationships between the particular referenced mission component and other components of a mission plan.

4.3 User Model

Many aspects of a user are highly relevant to interface technology. These aspects include level of expertise in the current task, perspective based on his role, his value system, degree and nature of impairedness due to fatigue or illness, and preferences concerning mode of communication. Carberry [Carberry87] provides a brief summary of recent research on user modeling. *Computational Linguistics* Vol. 14 No. 3 [Kobsa88] provides in-depth papers on several current approaches to user modeling. To address all of these aspects of user modeling is, of course, beyond the scope of this project. The aspects of the user that are most relevant in the CUBRICON system are (1) the importance rating that the user attaches to the different entity types that are relevant to each given task, which we call the user's *entity rating system*; and (2) the task on which the user is currently engaged.

CUBRICON includes a representation of the user's entity rating system as a function of the task being addressed by the user. For a given task in the process being carried out by the user, the entity rating system representation includes a numerical importance rating (on a scale from zero to one) assigned to each of the entity types used in the application task domain. The numerical rating assigned to a given entity type represents the degree of importance of the entity to the user. Associated with the entity rating system is a *critical threshold* value: Those entities with a rating above the critical threshold are critical to the current task and those with ratings below the threshold are not.

The entity rating system also includes the most important properties, characteristics, and components associated with the various entities. For example, the important properties associated with an airbase in the entity rating system are name, disposition, location. Other properties not listed among the most important are its nationality and the presentation color of the icon representing the airbase. The associated attributes listed in the entity rating system are primarily used by the table composition component to determine which ancillary attributes are relevant to the user and should be displayed in addition to the attributes explicitly requested by the user. A partial listing of the CUBRICON entity rating system list is shown in Figure 4-4.

The CUBRICON design provides for the entity rating system representation to change automatically under program control in the following manner: (1) when the user's task changes the system replaces the current entity rating list with the standard initial rating list for the new task; and (2) when the user mentions an entity whose rating is lower than the critical threshold, then its rating is reset to be equal to the critical threshold to reflect the user's interest in the entity and its seeming relevance to the current task from the perspective of the user. In the current implementation, CUBRICON performs the only second function listed above. The implementation of the first function is not complete.

The entity rating system is used by CUBRICON for determining which entities to display in response to user requests. This is accomplished as follows: (1) It is used in determining what information is relevant in answering questions or responding to commands from the user. (2) It is used in selecting ancillary information to enhance or embellish the main concept being expressed and to prevent the user from making false inferences that he might otherwise make. (3) It is used in organizing the form in which information is presented, particularly in the composition of tables.

As an example of (1) above, if the user instructs the system to "Display the Fulda Gap Region", CUBRICON uses the entity rating system representation to determine what

```
(("air base"            99   ((object (|disposition| |location| |name|))))
 ("SA-2"                98   ((object (|disposition| |location| |mobility|))))
 ("SA-3"                97   ((object (|disposition| |location| |mobility|))))
 ("factory"             85   ((object (|disposition| |location| |name|))))
 ("control radar"       75   ((object (|disposition| |location| |name|))))
 ("search radar"        75   ((object (|disposition| |location| |name|))))
 ("runway"              72   ((object (|disposition| |location| |name|))))
 ("fuel storage tank"   71   ((object (|disposition| |location| |name|))))
 ("tank battalion"      27   ((object (|disposition| |location| |name|))))
 ("infantry division"   26   ((object (|disposition| |location| |name|))))
 ("ORB mission plan"    10   ((object (|location| |name|))))
 ("STN mission plan"    10   ((object (|location| |name|))))
 ("package"              0   ((object (|name| |preparer| |date|))))
 ("ac-pool"              0   ((object (|name| |max-availability|))
    (comp   (|ac-pool|))
    (value  (|ac-pool-used|))))))
```

Figure 4-4: Partial Entity Rating System List

objects within the Region should be displayed. If the user is a military mission planner, then displaying all the country cottages in the region, for example, is irrelevant. The objects to display are those that are relevant to the job of the mission planner. Thus the objects that the system selects from its data base for display are airbases, missile sites, targets, etc. Section 10 discusses examples of the use of this entity rating system representation in interactive dialogue between a user and the CUBRICON system.

CUBRICON includes a simple representation of the current task in which the user is engaged. CUBRICON's mode of response to the user is affected by whether or not the user's task has just changed. The CUBRICON team is developing a task hierarchy: a decomposition of the user's main tasks into subtasks. This a priori task knowledge can be used by CUBRICON to help track the discourse focus, manage the displays, and anticipate the needs of the user.

5 Multi-Modal Language Understanding

Multi-modal communication is common among humans. People frequently supplement natural language communication with simultaneous coordinated pointing gestures and drawing. Similar multi-modal communication can facilitate human interaction with modern sophisticated information processing and decision-aiding computer systems. Multi-modal communication is not only natural for humans, but has more expressive power and is frequently a more efficient means of communication than single-modal communication. A human-computer interface should not only support multi-modal input, but should make use of the synergistic properties of the modalities to maximize expressive power and efficiency for the user.

The CUBRICON system has been developed to enable users to express themselves using multiple media. More specifically, CUBRICON accepts spoken and/or typed natural language accompanied by simultaneous coordinated deictic pointing gestures for input to the system. Multi-modal language understanding refers to the system's ability to accept input from the different input devices and interpret it in a consistent and coordinated way. The underlying viewpoint is that the input streams from the different devices should not be seen as separate symbol streams, but as components of a single multi-modal input stream. Users are therefore free to combine different modalities as well as substitute expressions in one modality for equivalent expressions in another.

The use of multi-media language provides the user with greater expressive power, but it entails certain problems also. For example, a point gesture by the user can be ambiguous if his point gesture touches the area where two or more graphical figures or icons overlap. The user can also inadvertently miss the object at which he intended to point, thereby providing the system with an expression that has an apparent null referent. CUBRICON includes methodology to handle these problems. CUBRICON provides synergistic mutual disambiguation of simultaneous natural language and pointing gestures as well as the ability to handle certain types of ill-formed multi-modal input such as inconsistent NL/pointing expressions and expressions that have an apparent null referent.

This section discusses the CUBRICON methodologies for (1) parsing and interpreting multi-modal inputs, (2) the mutual disambiguation of natural language and simultaneous coordinated point gestures, and (3) handling certain types of ill-formed multi-modal inputs.

5.1 Multi-Media Input Coordination

The system accepts input of three types:

- spoken natural language as provided by a discrete speech recognition system,
- written natural language as provided by keyboard input, and

- pointing and graphical gestures as provided by mouse input.

Input from the three devices is integrated into a single multi-modal input stream before any parsing or interpretation takes place. This process of integrating the input streams into a single stream is referred to as input coordination. We first discuss the three types of input separately and then take a look at the integration process.

5.1.1 Speech

Speech input is processed by a Dragon Systems discrete speech recognition system running on a micro-computer. It uses a context free grammar that describes acceptable input sentences. Since this grammar is separate from the one used for CUBRICON's multi-modal language parsing and interpretation (Section 5.2), there can be differences between the type of spoken and typed sentences that are accepted. For example, the speech system could be programmed to accept shorter forms of long words and translate them into their full-length version before communicating them to the multi-modal language parser-interpreter which resides on the Symbolics Lisp machine. The main limitations of the speech recognition system are its discrete and speaker dependent nature.

Speech input that has been recognized by the speech recognition system is communicated to the CUBRICON system running on the Symbolics Lisp machine in the form of ASCII text. The ASCII text is input to the Symbolics *via* a *serial port connection* and is channeled into the input buffer of the Natural Language Interaction Window (see Section 5.1.4).

5.1.2 Written Language

The user can also input sentences in natural language using the keyboard. The standard Symbolics input editing facilities are available to the user. The end of a sentence is detected when a period, question mark or exclamation mark is read, making the use of the return key unnecessary. Typed input is also channeled into the input buffer of the Natural Language Interaction Window.

5.1.3 Deictic Gestures

Deictic gestures are made using a standard three-button mouse. Since one of the basic premises of our system design is that the language provided to the user for input to the system should be natural, the system does not distinguish between the use of the different mouse buttons. We do not feel that single or double clicking the mouse, left, right, or middle, is a natural type of language for the user. Therefore, CUBRICON ignores the fact that a click was entered using a particular button (left, right, or middle). Thus, the user does not need to worry about the difference between the three buttons on the mouse: he can use them interchangeably.

As part of a multi-modal language input, the user can use a deictic gesture to point to a variety of object types that may be visible on either the monochrome or the color-graphics screen. There are frequently many different window types (e.g., tables, maps, form) on the screens at any given time, each displaying different object types. The objects that can be referenced via a point gesture are of six types: geometric points, entities represented by a graphic icon, table entries, form slots, the content of any form slot, windows on either the monochrome or color graphics screens.

In addition to being used as a simple pointing device, the mouse can also be used to input graphic drawing gestures. Currently, the only type of graphic drawing gesture that CUBRICON accepts from the user is the drawing of a closed polygonal path representing a flight path. The user sequentially indicates the vertices of the polygonal path on a map window on the color graphics screen (Section 10). As the user points out each of the vertices on the map window, the system connects the vertices in sequence with directed line segments so the user can view the path as it is being drawn.

Just as in the case of speech and typed language input, all mouse input is redirected to the input buffer of the Natural Language Interaction Window. Mouse clicks are represented in a symbolic form in the input buffer. When the user clicks a mouse button, the Symbolics operating system (Genera 7.2) returns the button clicked (left, middle or right), the window the mouse cursor was over, and the X and Y window coordinates of the mouse cursor. CUBRICON converts this information into a string (which we call a "blip string") which is put into the input buffer. A blip string has the following format:

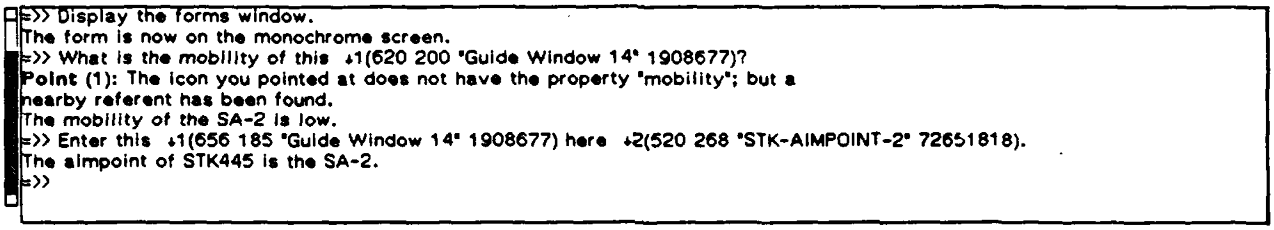
↓*n(x y window-name window-pointer)*

where n is the ordinal number of the mouse click in the current sentence, x and y are the window coordinates of the mouse cursor location, *window-name* is the name of the window the mouse cursor was over, and *window-pointer* is a Genera pointer to the window. This string is echoed in the Natural Language Interaction Window, providing feedback for the user. Since the mouse click information is now an ordinary string in the input buffer, it can be edited using the Genera editing facilities.

5.1.4 Coordination

Input from the three input devices is combined in the single input buffer associated with the Natural Language Interaction Window. The input tokens in the buffer are in the form of a list which preserves the order in which they were entered into the system by the user.

All input to this buffer is echoed to the Natural Language Interaction Window in a visual representation (see Figure 5-1). This window is also used to display feedback messages from the various CUBRICON components.



```
=>> Display the forms window.
The form is now on the monochrome screen.
=>> What is the mobility of this ↓1(620 200 "Guide Window 14" 1908677)?
Point (1): The icon you pointed at does not have the property "mobility"; but a
nearby referent has been found.
The mobility of the SA-2 is low.
=>> Enter this ↓1(656 185 "Guide Window 14" 1908677) here ↓2(520 268 "STK-AIMPOINT-2" 72651818).
The aimpoint of STK445 is the SA-2.
=>>
```

Figure 5-1: *The Natural Language Interaction Window echoes all input in its printed representation. It also displays natural language output produced by the system.*

Conceptually, the input streams may be seen as parallel data streams before they are integrated. The integration of these streams proceeds in a linear way, inserting whatever is available from any device at the current point in time into the input buffer. The input buffer therefore reflects the order in which multi-modal tokens (spoken or written words or mouse clicks) were entered.

5.2 Multi-Modal Language Parsing and Interpretation

After a multi-modal token list (string) has been assembled from the different input media, the list is passed on to the parser-interpreter component. This component's task is to perform syntactic analysis (parsing) of the token list and assign a meaning to it (interpretation). Although we discuss parsing and interpretation separately they are really interleaved in time. The guiding principle is that syntactic representations serve only as intermediate structures until an interpretation (semantic structure) can be determined. For any syntacticly recognized substring of an input sentence or utterance, semantic interpretation is performed as soon as possible. That is, a semantic interpretation is derived for any given substring or phrase as soon as the antecedents or conditions are satisfied for the applicable semantic procedures.

5.2.1 Parsing

The integrated multi-modal input stream is processed on a per sentence basis. A search of the lexicon is first performed for each word in the input string. CUBRICON informs the user about any words not found in the lexicon, and presents the user with a list of completions for any incomplete multi-word expressions. It then asks the user to re-enter the sentence if necessary.

The parser performs syntactic analysis for each multi-modal input sentence. This includes checking that the mouse clicks occur in appropriate places in the sentence. Mouse clicks may occur anywhere within a noun phrase or an adverbial phrase. They can also replace a noun phrase completely. The parser builds a parse tree for each recognized phrase or sentence and invokes the interpretation process to determine a meaning representation for each parsed phrase or sentence.

5.2.1.1 The Grammar. The grammar used by CUBRICON is implemented in the form of a Generalized Augmented Transition Network (GATN) [Shapiro82a]. For a general introduction to ATN grammars, see [Bates78]. The same type of GATN grammar is also used in the CUBRICON natural language generation component (see Section 8). A description of the input grammar appears in Appendix B.

5.2.1.2 The Lexicon. The lexicon is a dictionary of words that the system understands. Associated with each word is one or more lists of syntactic, morphological, and semantic features. Multiple feature lists are associated with lexically ambiguous words (e.g., "start", which can be a noun or a verb). The syntactic and morphological features are used by the parser to determine sentence structure. These features include syntactic category, multi-word lexeme status, linguistic number, gender and case for pronouns, and *root form*. *Semantic* features specify meaning-related attributes of words such as the key or function to be used in accessing the knowledge base for information about the concept a word expresses, case frames associated with verbs, or special media-related attributes of the corresponding concept. Examples of the latter are words like "monitor" or "map" that are classified in the lexicon as display objects or form slot names (see Section 12) that are listed with a feature indicating how to retrieve information from the corresponding slot.

5.2.2 Interpretation

A distinguishing characteristic of the CUBRICON system is its ability to interpret multimodal sentences consisting of simultaneous coordinated natural language and deictic gestures. In some cases the multi-modality of an input phrase is used to determine a referent when it would not be possible to do so for an equivalent phrase of single modality. This cross-modal dereferencing capability is a unique feature of the CUBRICON architecture. This capability demonstrates one aspect of the synergism between different modalities as well as the fact that the multi-modal total is greater than the sum of its parts.

Point gestures that are used in combination with natural language can be categorized as follows:

- the point gesture touches the intended object(s) only,
- the point gesture touches more objects than were intended,
- the point gesture misses the intended object altogether.

The first case is generally trouble-free. The latter two cases provide a challenge for a multi-modal interface system. CUBRICON handles these latter two cases if the accompanying natural language mentions either a property or an object type that the system can use to infer the intended referent.

If a point gesture touches the area where two or more (presentation) objects overlap (e.g., the intersection of the extents of a SAM icon and an airbase icon), then the interpretation of the point gesture alone is ambiguous. Without further information the system will not be able to infer which is meant and will return both as intended referents. However, if the accompanying natural language mentions an (conceptual) object type or class (e.g., SAM or airbase), then only the objects that satisfy the class membership criteria would be accepted as referents of the multi-modal phrase. Similarly, if the accompanying natural language mentions a property (e.g., mobility), then only the objects that satisfy the property criteria would be accepted as referents of the multi-modal phrase.

Another problem with multi-modal input is that a point gesture can be inconsistent with the accompanying natural language phrase. For example, a person might enter *"this airbase <point>"* while his point gesture touches a SAM icon. The immediate interpretation of the gesture is the SAM object corresponding to the touched icon, but since the natural language part of the phrase indicates an airbase, this interpretation is rejected. CUBRICON tries to infer the intended referent by performing an incremental bounded search to find an object that satisfies the criteria imposed by the accompanying natural language phrase. The incremental search starts at the point at which the user's point gesture touched the window. This CUBRICON procedure for handling inconsistent natural language and pointing gestures can utilize either object types or properties that may be mentioned in the natural language part of the multi-modal input. CUBRICON stops the search when the first object (one or more) is found that satisfies the criteria or when the distance bound on the search is reached.

Another problem with multi-modal input is that the user's point gesture may touch no objects at all (other than the window itself). For example, the user may enter a sentence such as *"What is the mobility of this <map-point>?"* and his point gesture may touch no icons on the map. The point gesture alone has an apparent null referent and is inconsistent with the accompanying natural language phrase. In order to infer the intended referent, CUBRICON performs an incremental bounded search starting at the location of the user's point gesture. The system searches for objects that satisfy any type or property criteria that was mentioned in the accompanying natural language phrase. In this example, the system searches for objects that have a property called "mobility." CUBRICON stops the search when the first object (one or more) is found that satisfies the criteria or when the distance bound on the search is reached.

As an example of disambiguation in the other direction (i.e., graphical gestures disambiguating natural language), consider the sentence *"Enter this <point>$_1$ here <point>$_2$"*. Without the graphical gestures this sentence is obviously not interpretable, since both "this" and "here" can refer to many different objects and locations, respectively. But given an interpretation for each of the graphical gestures (e.g., an object represented as a map icon and a form slot, respectively), the sentence is perfectly interpretable and unambiguous. In fact the disambiguation process is bidirectional in this case. The adverb "here" signifies a locative phrase, thereby directing the interpretation process to search for something corresponding to a location in the given context (the kind of window that the point is entered on), viz. a form slot. In contrast with the second point, the first one is interpreted to refer to a domain object and not to a location on the form (an interface object) because of the syntactic context and the deictic pronoun "this". The syntactic context alone will suffice in this case, so even if the pronoun is omitted the sentence will still be interpreted correctly. In fact if both mouse points touch the same form slot, and if that slot already contains a (description of) a domain object, the net result is that the content of the slot will be overwritten with what was already there, since the first point is taken to refer to the content and the second to the slot itself. Although this is not very useful, it is a good indication of the dereferencing capabilities of the system.

The interpretation process takes (as input) parse trees produced by the syntactic parser and determines interpretations for them. Since point gestures can occur in noun phrases and locative adverbial phrases, we discuss these types of multi-modal phrases in the following subsections.

5.2.2.1 Multi-Modal Noun Phrases Multi-modal noun phrases can consist of zero or more words of text along with zero or more pointing references to objects on one of the displays. There must be at least one word or one point. Mouse points can occur anywhere within the phrase, including before the first word or after the last word (if any). Multi-modal noun phrases can have subordinate prepositional phrases. Common uses of mouse points include the following:

- substituting for an entire noun phrase: *"What is the mobility of <point>?"*
- substituting for the head noun: *"What is the type of these <point>$_1$ <point>$_2$ <point>$_3$?"*
- in conjunction with a complete natural language noun phrase: *"Display the status of this <point> airbase."*

The objects that can be referenced via pointing with the mouse are of six types:

1. a geometric point represented by a pair of coordinates on a map
2. an entity represented graphically (as an icon)
3. a table entry
4. the content of a form slot
5. the form slot itself
6. a window on the monochrome or color display.

The interpretation of any noun phrase that includes one or more deictic gestures depends on the semantic type of the head noun of the noun phrase and/or the type of the main verb of the sentence:

1. if the natural language portion of the noun phrase contributes no constraining information to be used by the interpretation procedure and the point gesture touches a window other than a map window, then the interpretation is deferred to a later stage of the parsing/interpretation process when additional information will be available. The mouse blip is retained as the interim interpretation. An example of such a noun phrase is *"this <point>"* where the point gesture touches on the mission planning form window and, without additional constraining information, the phrase could be interpreted to mean either the form slot itself or the contents of the slot. The interpretation of point gestures on a map window is discussed below.

2. if the head noun of the NP expresses the concept of a geographical point (e.g. a word such as "point" or "location"), the mouse point is not interpreted any further at this stage. The mouse blip is retained as the interim interpretation of the phrase. In the case of a sentence such as *"Zoom in on this <point>"*, the mouse blip is translated into lat-long coordinates on the map when the sentence level processing is performed and the constraints of the case frame of the verb are applied.

3. if the head noun expresses the concept of a window, the window name is retrieved from the blip string, and the KB object (node) representing the window is retrieved and returned as the interpretation.

4. if the point gesture touches a map window and the head noun does not express a window concept, then the mouse points are taken to refer to objects represented visually by map icons. This case uses some special disambiguation routines. Pointing gestures can be categorized as follows: (1) the mouse point touches the intended icon only,

(2) the mouse point touches the area where the extents of two or more icons overlap and not all were intended to be selected, (3) the mouse point misses the intended icon. Determining the intended referent(s) for a multi-modal noun phrase proceeds as follows:

(a) for each point gesture, collect the KB domain objects (nodes) represented by the icons that were toucned by the point gesture (these KB objects or nodes are available from the window's content list which is part of the Display Model, discussed in Section 4.2.2). There may be more than one object or none at all.

(b) if at least one KB domain object (node) was found for a point gesture, filter the nodes as follows (return the first non-empty set of nodes):

   i. if the NP contains no head, return all nodes

   ii. if the head refers to the superclass of any nodes, return them

   iii. if the head refers to the class of any nodes, return them

   iv. if the head is the name of any nodes, return them

   v. if the head refers to a property of any nodes, return them

(c) if the previous step yields an empty node list, perform an incremental bounded search in the area around the point touched by the user's mouse click, up to a maximum predefined distance in pixels. The search procedure stops when one or more objects are found that satisfy the filtering criteria listed above or when the maximum predefined search distance is reached.

(d) present feedback messages to the user about near misses (mouse points that did not touch any icon or the wrong kind of icon, but for which a referent has been found by the incremental bounded search) and complete misses (mouse points for which no referent could be determined).

If the interpretation process for the noun phrase has been unsuccessful, then interpretation of the noun phrase will be attempted again at a later stage in the analysis of the input sentence. Such a later stage is at the sentence level when the main verb and its case frame are available. This is discussed below.

5.2.2.2 Multi-Modal Adverbial Phrases In the current CUBRICON implementation, the only type of multi-modal adverbial phrases are locative phrases. In general, the mouse blip is returned as the interpretation of the adverbial phrase, to be used at a higher level of interpretation (e.g., sentence level). An example of a sentence using a locative multimodal adverbial is *"Enter the Nuernberg airbase here <point>"*, where the point gesture touches a slot on the mission planning form window. For this example, the final interpretation of the adverbial phrase *"here <point>"* is determined by using the case frame for the main verb "enter". This case frame requires a form slot as the recipient (destination), so the final interpretation of the adverbial phrase is the slot on the mission planning form.

5.2.2.3 Final Interpretation The interpretation of a sentence is represented in the form of a case frame structure. The case frame used for a particular sentence is determined by the main verb and the syntactic structure of the sentence (imperative, declarative, truth question, content question). A case frame is associated with each verb in the lexicon. It has slots that are associated with specific semantic roles. Possible slots are *agent, object, action, value, recipient, location,* and *context.* The representation of the case frames in the lexicon includes either a value for the slots (e.g. the value *display* for the *action* slot associated with the verb "display") or a directive for the final interpretation procedure to use in determining a value.

The *agent* slot refers to the agent of an action, and is frequently interpreted as the *system* (i.e., CUBRICON is to perform the specified action). The *object* slot refers to the object on which the action is being performed (i.e., some KB domain object or an interface-related object like a window). The *action* slot refers to the kind of action to be performed (e.g., *display, enter, present*). The *value* slot may be used to refer to a value being assigned to some object. The *recipient* slot refers to the recipient of an object (e.g., a mission plan for which a flight path is to be planned). The *location* slot refers to a location in a broad sense (e.g., a table on which an object is to be highlighted).

As indicated in the previous subsections on interpreting multi-modal noun phases and adverbial phrases, the final interpretation of some phrases must be delayed to a later stage of processing when additional information is available. During the sentence level interpretation process, if the case frame of the main verb of the sentence imposes any constraints on the referents of constituent noun phrases, then the constraints are applied. This process enable CUBRICON to determine the final interpretation of any unresolved constituent phrases. For example, in the current CUBRICON implementation, the verb "enter" can be used as a command to the system and requires two constituents in the sentence: the object upon which the action is performed and a recipient for the object. For the sentence *"Enter this <point>$_1$ here <point>$_2$"* where the both *<point>*$_1$ and *<point>*$_2$ are gestures that point to slots on the form window, the first point gesture cannot be interpreted at the noun phrase processing level. At that level of processing, the point gesture could be interpreted to mean the form slot itself or the contents of the form slot. When sentence level processing is performed, however, the constraints associated with the case frame slot fillers for the verb "enter" are applied and the ambiguity of the first noun phrase is resolved. It is interpreted as refering to the content of the touched form slot. This input sentence would have the effect of copying the contents of one form slot into the second.

In this section we discussed the CUBRICON multi-modal language understanding capabilities. The CUBRICON parser-interpreter produces a representation of the interpretation of the user's multi-modal input sentence. This interpretation structure is handed over to the Executor component which executes the action that was intended by the user. The Executor component is discussed in the next section.

6 Executor

After an interpretation has been determined for a multi-modal input sentence, this interpretation representation is handed off to the Executor component from the Multi-Modal Parser-Interpreter. From the representation of the sentence interpretation, it is the responsibility of the Executor to perform the *action intended by the user*. Other components of the system perform actions, but they would not be actions performed in carrying out the intention of the user. Instead, these other types of actions are actions that CUBRICON performs to maintain its knowledge sources/models and displays in a form that it believes is accurate and relevant for the user. For example, the Parser-Interpreter modifies knowledge sources such as the Focus List and Entity Rating List to reflect the importance that the system believes or infers that the user attaches to these concepts due to the manner in which the concepts are referenced by the user. Such modifications of the Focus List or Entity Rating List are not performed because it is the intent of the user to modify them.

In general, the types of actions that an executor component performs, is partly dependent on the application system(s) to which the interface system communicates. The types of actions currently handled by CUBRICON are: knowledge base addition or modification, knowledge base retrieval, interface manipulation, and file system update. The Executor extracts appropriate information from the representation of the interpretation of the user's input that was provided by the Parser-Interpreter, formulates a command or request (e.g., knowledge base update or retrieval) to carry out the action intended by the user, and executes the action. Information to be communicated back to the user as a result of the action is handed off to the Multi-Modal

Output Planner (see Section 7) which handles the expression/presentation of information to the user.

The following subsections discuss the main types of actions available to the CUBRICON user.

6.1 Knowledge Source Modification

The user can modify two of the knowledge sources discussed in Section 4: the knowledge base and the representation of the user's current task.

6.1.1 Task Representation Modification

As discussed in Section 4.3, the system maintains a representation of the task on which the user is currently working. This task representation is used by the system in composing displays to judge the relevance of various conceptual objects. The user can change the current task representation by an input statement such as *"Make PKG0026 the current package"*, where *PKG0026* refers to a mission planning package with a knowledge base node representation (it could be referred in some alternative way such as with to a point gesture on a table of packages instead of by the proper noun "PKG0026"). Such an input statement then replaces the representation of the user's current task with the knowledge base node representation of the PKG0026.

6.1.2 Knowledge Base Modification

The user can add or modify the knowledge that is represented in the knowledge base. This type of action is currently concerned with application domain concepts, particularly the composition of a package and its component mission plans. The user can enter new information into the knowledge base via input statements that manipulate the form such as: *"Enter this <point-on-map> here <point-to-form-slot>"* or *"Enter the value 0730 here <point-to-form-slot>"*. The user can also enter new information into the knowledge base via declarative input statements such as: *"Enter the Nuernberg Airbase as the origin of OCA111"*. Such mission planning statements update the visible form as well as the knowledge base structures.

Another knowledge base update action that the system can perform is that of assisting the user in specifying a flight path for an OCA mission. The flight path planning action is taken for input sentences like *"Plan a flight path for OCA123"*. The executor invokes the flight path definition function (see Section 5.3) which accepts user input designating waypoints and builds the knowledge base structures corresponding to the path, an arrival time for each waypoint, and the relationship with the particular OCA mission.

6.2 Knowledge Base Retrieval

The user can request that information from the knowledge base be retrieved and presented. The user can query the system with regard to the various types of concepts, properties, components, parts, and characteristics discussed in Section 4.1 on the Knowledge Base. The following list summarizes the types of knowledge structures or relations that can be retrieved from the knowledge base.

- **Member of a class.** One or more members of any given class can be retrieved and presented to the user. This type of information would be retrieved in response to user inputs such as: *"What is this <map-point>?"*, *"Is this <point> a SAM?*, *"List the SA-2s."*, *"List the packages."*.
- **Property of an object.** Any property of any object can be queried by the user. The properties available depend on the type of object, of course. The properties

that are more common among all objects are: name, disposition, location, nationality. These properties can be requested by such user inputs as: *"Where is the Nuernberg Airbase?"*, *"What is the mobility of this <map-point>?"*, *"What is the origin of the OCA123 mission plan?"*

- **Part of an object.** As discussed in Section 4.1, a "part" refers to an inseparable part of an object, such as the runway of an airbase. Example requests that retrieve part-of relationships from the knowledge base include: *"What radars are at the Dresden Airbase?"*, *"What are the aimpoints within the Dresden Airbase?"*.

- **Component of an object.** As discussed in Section 4.1, a "component" refers to a separable part of an object, such as in the relationship between the aircraft pools at an airbase. An example request that retrieves a component-of relation from the knowledge base is: *"What ac-pools are at the Nuernberg Airbase?"*

- **Characteristic of an object.** In the CUBRICON system, the characteristic-of relation is used only for properties of intangible objects such as mission plans and packages (see Section 4.1). It is primarily used to represent the subtask-supertask relationship between mission types. Such characteristics can be retrieved by user inputs such as *"Whct is the STN mission plan for SVC1002?"*.

- **Region information.** The relevant contents of any given region can be retrieved by the user by an input statement such as *"Display the Fulda Gap region."*. The manner in which CUBRICON composes map displays is discussed in Section 10.

6.3 Interface Manipulation

Certain actions could be classified as interface manipulation, but may result in knowledge base retrieval also. Although a basic tenet of this project has been to minimize the effort on the part of the user to manage and control the interface so that the user can concentrate on his application domain task (e.g., mission planning), the user may request the system to perform interface actions that assist him in finding or focusing on certain information on the displays. Such actions include: *display, blink, highlight, zoom in, bury (a window), expose (a window)*. A command such as *"Highlight the heliports"* will cause the system to highlight the heliports if they are already on the display, but if they are not, the system will retrieve relevant information from the knowledge base so that it can display them and then highlight them. Similarly with a "blink" request. Requests such as "display" and "zoom in" will modify the interface displays, but will always entail knowledge base retrieval as well.

6.4 File System Update

There is currently only one form of file system update action that the system can perform. This action consists of saving the current contents of a mission planning package (equivalent to thr contents of the Form) that the user has been constructing to a disk file with the same namr as the name of the package. An input sentence that will cause the current package to be saved to disk is *"Save the package as PKG2222"*.

7 Multi-Modal Output Planner

The Multi-Media Output Planner composes the response that is to be produced to the user by the Output Generator in coordinated multiple modalities. The Output Planner determines the media and modalities for expressing the response information to the user, but then must determine whether the resources are available in order to do so. If they are not, then the Planner must take appropriate action to modify the state of the resources, modify the information to be expressed, and/or select different modalities for expressing the information before the composition of the output can be accomplished.

The top level output planning process is summarized below. This planning process presupposes that the primary relevant information has been obtained to respond to the user.

1. Assess the availability of the monochrome and color graphics devices. If none of the window positions on the monochrome device are available and there are window positions available on the color graphics device, then the color graphics device is the preferred device. This would supersede the monochrome device as the preferred media for the table modality.

2. For each information item or cluster, determine the modality in which it should ideally be expressed. Graphic/pictorial presentation is always desirable. Natural language can always be used, as a last resort if no other modality is available.

3. Determine whether the resources are available to express the information as desired. Resources: (1) Color graphics display: Are the items to be expressed graphically already on the color display (e.g., objects of interest in a geographical domain may already be displayed on a map)? If so, no additions are necessary. If not, is there room to add them in their "natural" position? (e.g., can the desired objects be inserted in the area already on the graphics display without changing the area shown or does the displayed area need to be extended or changed totally?) (2) Monochrome display: Similar to the color graphics display. (3) Speech output device: Always available.

4. If the desired resources are not available, modify the state of the resources. The desired resources would be "not available" if the device (e.g., a display) already contains critical information that cannot be disrupted nor covered by a window. For the graphics displays, if not all the items to be expressed graphically are on the graphics display, then the system must compose a new display. Borrowing terminology from the geographical situation, the possible cases are:

    - *Zoom out* with intelligent addition of relevant ancillary objects to fill in the new area to maintain consistency throughout the display.

- *Zoom in* with intelligent addition of relevant objects to create an intelligible display.
- *Pan* to a different area maintaining consistency in the types of objects displayed.
- Combination of the above.
- Display a different disjoint area. (i) Completely replace display with new "area" or (ii) Open a window on the monitor to show new information.

A detailed explanation of the methodology used to dynamically compose geographic maps is in Section 10.

5. If the desired resources are still not available to accommodate the information to be expressed, try modifying the information to be expressed: trim the amount of information by filtering on the basis of relevance with regard to user model and/or discourse model.

6. If the information can still not be expressed in the given modality due to insufficient resources for the selected modality, then select another modality and go back to step 3.

7. Compose the output, having resolved resource constraints.

8. Repeat the modality selection and generation process until all modalities have been evaluated.

7.1 Modality Selection

Selection of the most appropriate modalities for expressing information in the CUBRICON system is based on the nature and characteristics of the information. Our system design is based on the premise that graphic/pictorial presentation is always desirable. The following is a brief summary of the selection criteria.

1. *Map*: Selected whenever CUBRICON knows how to represent the information pictorially.

2. *Table*: Selected when the values of common attribute(s) of several entities must be expressed.

3. *Form*: A predefined form is selected when the task engaged in by the user requires the form.

4. *Pointing Gesture*: Pointing gestures are selected whenever an object or the property of an object is requested, so that the attention of the user will be drawn to the object. Four types of pointing gesture modalities exist; map, table, form, and window.

5. *Text Box*: Text boxes are selected whenever textual information is to be written on the color graphics screen.

6. *Natural Language Prose*: Selected for the expression of a proposition, relation, event, or combination thereof, when the knowledge structures being expressed are heterogeneous. Natural language can be presented in either spoken or written form.

The selection of the media and modalities in which to express the response information to the user is based primarily on SNePS nodes and/or a command which results from the parsing and interpretatic of the user's request. In addition, the selection of some modalities depends on the modalities previously generated. The selection of modalities is done sequentially, evaluating modalities in order of importance. The following sections describe the types of modalities, described in order of preference.

7.1.1 Interface Manipulation by the User

Although the basic tenet underlying the CUBRICON design is to minimize the need for the user to manipulate or control the interface, CUBRICON does provide the user with a limited capability to directly manipulate the interface. The performance of this type of action on the part of the Output Planner is based solely on the command that is input to the Output Planner from the Executor component.

The current CUBRICON implementation provides the user with the ability to expose and remove (iconize) windows, request removal of flight path presentation objects, and blink or highlight presentation objects. Other types of user inputs that seem like interface manipulation actually entail decision-making on the part of CUBRICON with respect to what is output to the user and how it is expressed. An example of this is a request from the user to zoom in on a certain part of a geographical display. Such user inputs are discussed in other parts of this section.

7.1.2 Map Modality Selection Criteria

The map modality is selected whenever the information is geographic and CUBRICON can express the information visually. Two types of map modalities exist: (1) geographic area map and (2) a part-whole decomposition map. The criteria for selecting the map modality is based on the objects, represented as SNePS nodes, and the command that are input to the modality selection function as follows:

- Part-whole decomposition map

The input nodes represent the assertion that the objects are a part of an airbase, and there does not exist a map which contains all of the objects represented in the input nodelist. A user request which would generate a call the modality selection function with these nodes is "What are the aimpoints within the Merseberg airbase?". The output generated as a result of selecting this modality is a part-whole decomposition map containing the icons representing the objects which are parts of the Merseberg airbase.

- Geographic area map
  - Zoom In

    The zoom in modality is selected whenever the command :zoom-in is input to the modality selector. This command represents a user request to zoom in on a geographic area by stating "Zoom in on this point <point>.".
  - Map Icon

    The input nodes represent an object or class instance whose superclass can be represented as a map icon and at least one of the objects being requested is not on an active map. One user request which generates this case is "Blink the heliports." which inputs a node representing the class of heliports to the Modality Selector. The result of this request is to add all map icons which represent heliports and are located within the boundary of an active geographic map to the map.
  - Locative

    The input nodes represent the assertion that an object is located at a particular latitude and longitude and at least one of the objects whose location was requested are not on an active map. One user request generating this case is "What is the location of the Merseberg airbase?". The resulting output generated by the map icon modality is a map containing an icon representing the Merseberg airbase.
  - Region

    The region modality is selected whenever the input node represents the instance of a region and there does not exist a map which contains the area defined by the region. A region node has a latitudinal and longitudinal boundary defined for it. A user request which results in the selection of the region modality is "Display the Fulda Gap region." which generates a map containing the area included in Fulda Gap region boundary.

7.1.3 Pointing Modality Selection Criteria

The pointing gesture modalities are selected whenever an object or the property of an object is requested, so that the attention of the user will be drawn to the object. Several types of objects can be referenced by the Output Planner: windows, icons, table rows, and form panes. The modalities representing gestures which point to geographic objects, windows and icons, are selected and generated following the Map Modality. The remaining pointing gesture modalities are selected following other modalities. A detailed explanation of the pointing modality is presented in Section 9, Deictic Gestures.

- Window Pointing

  Window pointing occurs whenever a geographic map is requested and it is contained in an active map. One case which generates the window pointing modality is whenever the input node represents the instance of a region and a map exists which contains the regional boundary. In this case the input node is identical to the node input in the region modality described above. The second case which generates the window pointing modality is whenever the input nodes represent the assertion that the objects are a part of an airbase, and a map exists containing all of these objects. Once again, these input nodes are identical to the nodes input in one of the map icon modality cases described above.

- Map Icon Pointing

  Map icon pointing occurs whenever the CUBRICON system can express information graphically, without modifying the display, and a subset of the icons on a map are being pointed to. The input nodes which meet the map icon pointing criteria are identical to the nodes input in two of the map icon modality cases described above. The input node represents either an object whose superclass can be represented as a map icon, or the assertion that an object is located at a particular latitude and longitude.

- Highlight

  The highlight modality highlights map icon(s) and/or table entries based on user request. The highlight modality is selected whenever a highlight command is input to the output planner. The object to be highlighted is represented by the input node(s). The highlighting occurs on every window containing the object, unless a specific window has been requested by the user. In this case the window to be highlighted is passed to the output-planner in the destination-window parameter. A user request which generates the highlight modality is "Highlight this <point at the Nuernberg airbase> on the table.". The output planning system is passed a command highlight, a node representing the Nuernberg airbase, and a destination window which represents the table which is related to the map on which the input point gesture occurred. Based on this information the highlight modality is chosen and the table entry containing the Nuernberg airbase is highlighted on the destination window.

7.1.4 Table Modality Selection Criteria

The table modality is selected when the values of common attribute(s) of several entities must be expressed. In the current CUBRICON implementation, a table is generated when the number of entities is greater than four. This threshold is, of course, subject to change. Futhermore, the number of common attributes needed for the table modality to be selected is one. When determining if common attributes exist, not all properties are considered. The system only considers the properties listed in the User Model as being important to the user for the given task.

Two types of table modalities exist: monochrome table and color-graphics table. The monochrome table modality generates tables which are placed on the monochrome device, whereas the color-graphics table modality generates tables which are placed on the color-graphics device. Within the hierarchy of modalities, monochrome tables are the preferred, unless the Modality Selection system has determined that the color-graphics display has window positions available and the monochrome display does not. It is possible for the Window Management system to reject a request to create a monochrome table, due to the monochrome device being unavailable. If the Modality Selection system attempts to create a monochrome table and the Window Management system rejects this request, then the Modality Selector will choose the color-graphics table as an alternate modality. The same table, however, would not be presented in multiple modalities. A detailed explanation of the table window placement algorithm is in Section 13.3.

7.1.5 Table Entry Pointing Modality Selection Criteria

As previously mentioned one of the types of pointing gesture modalities is table entry pointing. Pointing to a table entry occurs whenever the information to be expressed is on an visible table. The objects represented in the input node list are compared with the Content list of each active table. If the object is contained in the table, then the corresponding entry is pointed to. The type of entities which meet the table entry pointing criteria are identical to the entities which generate the map icon pointing modality, which is described above.

7.1.6 Form Modality Selection Criteria

The predefined mission planning form is displayed upon request by the user. Section 12 provides additional information on the form modality.

7.1.7 Form Pane Pointing Modality Selection Criteria

An additional type of point gesture modality is form pane pointing. Form pointing occurs whenever the information to be expressed is on an active (visible) mission planning form. If the object or property being expressed is contained in the form, then the corresponding entry is pointed to. The information to be expressed would be in the form of a relation between a particular mission plan, a property of the mission, and the value of the property. An example user input that would cause CUBRICON to generate a the form pane point gesture is "Enter the Nuernberg airbase as the origin for OCA099."

7.1.8 Selection Criteria for the Text Box Modality

The text box modality is selected whenever textual information is to be displayed on the color graphics screen. There are three forms of text boxes: (1) dynamic text windows that are sized to fit the text within the window and to which additional text can be added with a corresponding increase in the size of the window, (2) a static text box with an associated directed line segment that extends from the text box to the entity (e.g., a map icon) about which the text speaks, and (3) a static text box without the directed line segment that is usually place near the visual object about which the text speaks.

The current CUBRICON system uses text boxes as part of integrated multi-modal output in three situations: when composing point gestures, when generating a mission presentation, and when a natural language response is being generated to express a property-value pair for a given entity that is visible on one of the displays. An example user request which selects the text box modality is "What is the mobility of this <point2>?" where the point gesture touches on one of the map displays. As part of the response, the property name and value associated with the object are placed on the map next to the icon representing the object. In this case, the text would be: ‘ obility: low".

7.1.9 Natural Language Prose Modality Selection Criteria

Natural language prose is selected for the expression of a proposition, relation, event, or combination thereof, when the knowledge structures being expressed are heterogeneous. Natural language can be presented in either spoken or written form. The following summarizes the selection criteria for spoken versus written language

- Spoken Natural Language
  - Dialogue descriptions to assist the user in comprehending the presented information. These include explanations of graphic displays or display changes and verbal

highlighting of objects on the displays (e.g., "The enemy airbases are highlighted in red").

- Informing the user about the system's activity (e.g., "I'm still working" when the user must wait for output from the system).
- Short expressions of relatively non-technical information that can be remembered when presented serially (e.g., a "yes"/"no" answer to a user's question).

- Written Natural Language Selected for longer technical responses that would strain the user's short term memory if speech were used (see [Miller56]).

7.2 Output Composition

Most frequently, multiple modalities are desirable to express a body of information to the user. For example, to inform the user about the details of a certain OCA mission plan, a desirable presentation would be an explanation delivered in combined spoken speech and coordinated drawing on a graphic map display showing movements of the aircraft, as well as a printed textual summary with ancillary information. The multiple modalities should be selected to complement and enhance one another. Andriole [Andriole86] has used "graphic equivalence" effectively using dual displays or split screens to present the same material in different forms to aid user comprehension and problem solving performance. We are not restricting the system to presenting the *same* material in different forms, but, instead, our system presents related material or different aspects of a given event or concept in different forms/modalities (as appropriate based on the nature and characteristics of the information).

The CUBRICON system rarely restricts output to one modality: typically multiple media and modalities are selected. Written and Spoken Natural Language, for example, are utilized in nearly every output presentation. In general, if CUBRICON represents and object graphically (eg. the location of an airbase is requested) output generation combines Map Icon and Icon Pointing modalities on the color graphics display, the Table and Table Pointing modalities on either the monochrome or color graphics display, Written Natural Language on the monochrome display, and Spoken Natural Language via the speech output system. In this example, the tabular presentation was selected because there are important attributes associated with the entities displayed on the map display. Specific examples illustrating the composition of multiple media and modalities are presented in the next section, Multi-Media and Multi-Modal Output Examples.

7.3 Multi-Modal Output Generation Examples

In order to further illustrate CUBRICON's modality selection and output composition process, consider the next user input. The user queries the system about the location of the

Nuernberg airbase in a manner that provides no instruction to the system as to how to present the information (e.g., map, natural language only, etc).

USER: "Where is the Nuernberg airbase?"

DEVICE CONFIGURATION:

The color graphics display contains a map displaying the Fulda Gap region and a table showing the important attributes of the objects is displayed on the map. The monochrome display contains a mission planning form.

CUBRICON: (Refer to Figure 7-1.)

Speech output:

- Statements to direct the user's attention to the appropriate monitor when a major window is presented. As the map is expanded on the color monitor: "The map on the color graphics screen is being expanded to include the Nuernberg airbase."

Color Graphics Display:

- Map of the Fulda Gap Region with added area that includes the Nuernberg airbase.
- Main roads, major cities, waterwsys, and national boundaries (as before but across the whole map, old and new areas).
- Icons representing entities within the new map area displayed that are above the critical threshold on the entity rating system for the user's task.
- An airbase icon representing the Nuernberg Airbase.

Speech Output with coordinated Color Graphics:

- After the map is expanded, statement to direct the user's attention to the Nuernberg airbase on the map: "Its location is here <point> 50 miles southeast of the East-West Germany border." The word "here" is accompanied by a visual point gesture in the form of blinking the airbase icon and the addition of a pointing text box.

Written Natural Language:

- A written an more detailed version of the previously spoken response is "Its location is 50 miles southeast of the East-West Germany border.".

Speech Output:

- As the table is presented on the monochrome monitor: "The corresponding table is being generated" and "The corresponding table is now on color graphics screen."

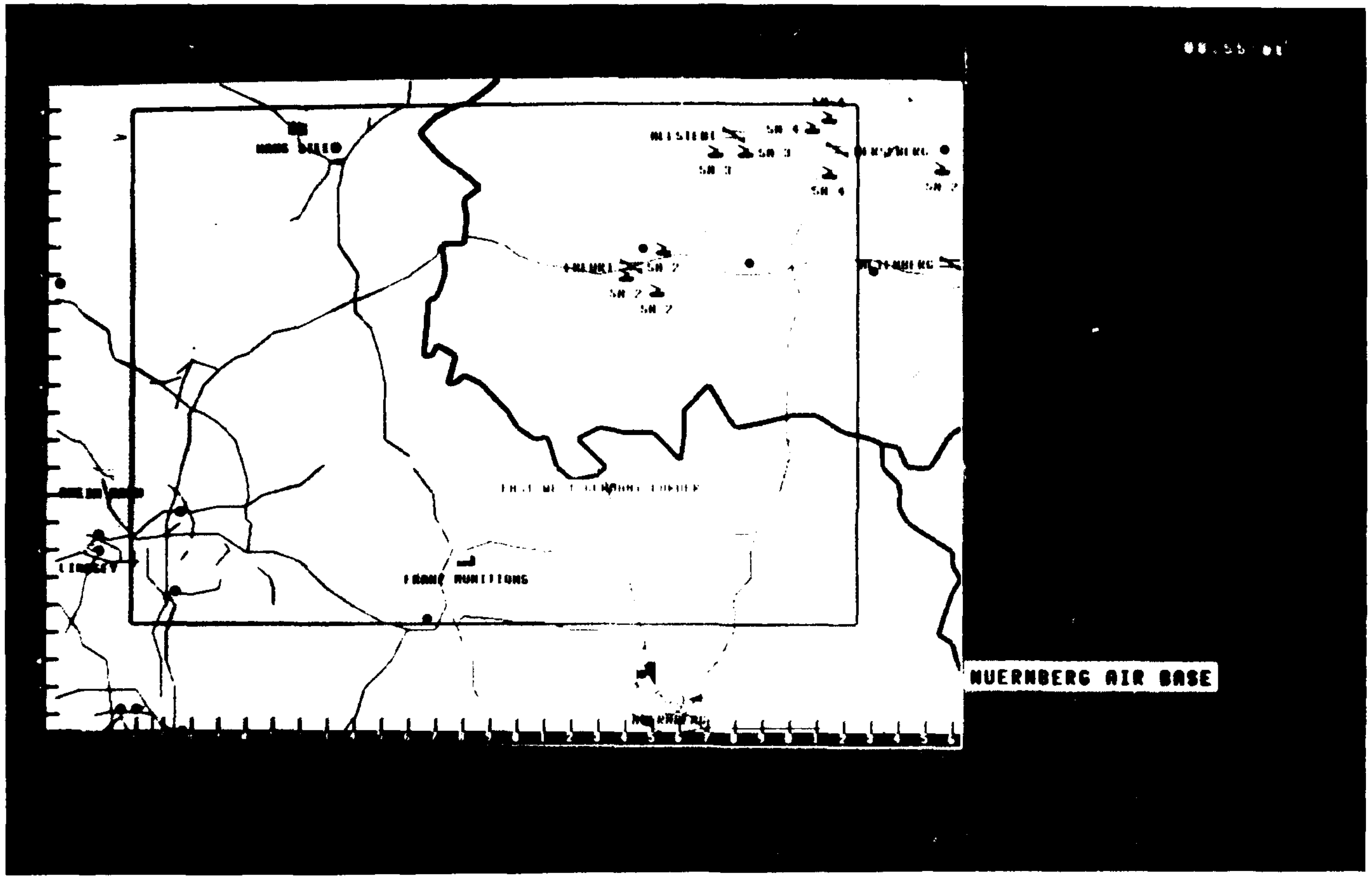
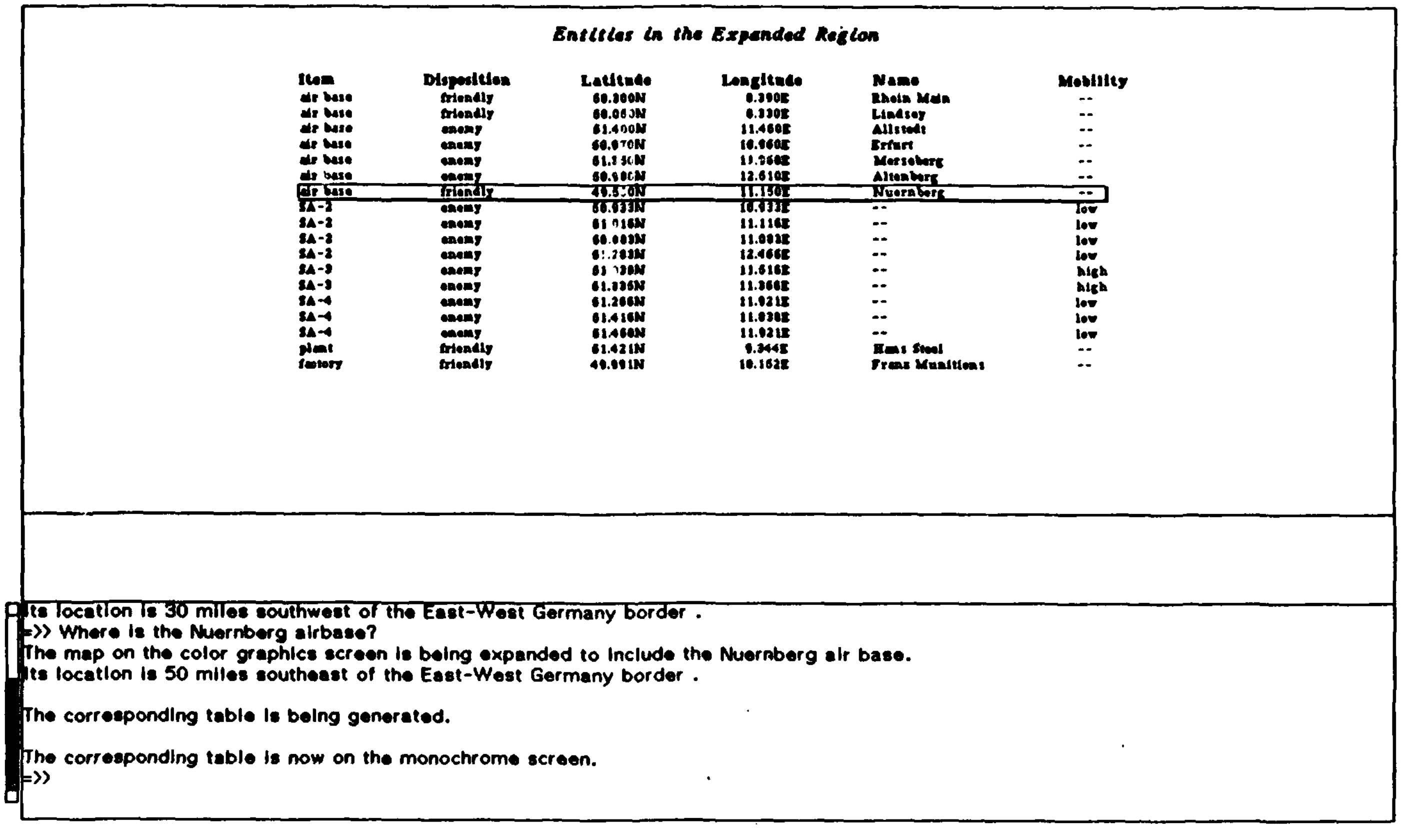


Figure 7-1a: Map and Table Maintaining Context and Consistency



Entities in the Expanded Region

Item	Disposition	Latitude	Longitude	Name	Mobility
air base	friendly	50.300N	8.390E	Rhein Main	--
air base	friendly	50.060N	8.330E	Lindsey	--
air base	enemy	51.400N	11.460E	Allstedt	--
air base	enemy	50.970N	10.960E	Erfurt	--
air base	enemy	51.350N	11.960E	Merseberg	--
air base	enemy	50.980N	12.610E	Altenberg	--
air base	friendly	49.500N	11.150E	Nuernberg	--
SA-2	enemy	50.933N	10.933E	--	low
SA-2	enemy	51.016N	11.116E	--	low
SA-2	enemy	50.883N	11.083E	--	low
SA-2	enemy	51.783N	12.466E	--	low
SA-3	enemy	51.130N	11.616E	--	high
SA-3	enemy	51.336N	11.366E	--	high
SA-4	enemy	51.266N	11.921E	--	low
SA-4	enemy	51.416N	11.838E	--	low
SA-4	enemy	51.466N	11.921E	--	low
plant	friendly	51.421N	9.344E	Hans Steel	--
factory	friendly	49.991N	10.162E	Franz Munitions	--

```
Its location is 30 miles southwest of the East-West Germany border .
=>> Where is the Nuernberg airbase?
The map on the color graphics screen is being expanded to include the Nuernberg air base.
Its location is 50 miles southeast of the East-West Germany border .

The corresponding table is being generated.

The corresponding table is now on the monochrome screen.
=>>
```

Figure 7-1b: Map and Table Maintaining Context and Consistency

Monochrome Graphics Display:

- Table of relevant entity attributes. Same table as before, but expanded to include the new entities added to the map covering the extended area.
- The table entry containing the attributes for the Nuernberg airbase is highlighted.

DISCUSSION:

As previously discussed, whenever possible the CUBRICON system prefers to present information to the user graphically with ancillary information presented simultaneously in an another modality. Since CUBRICON knows how to display an airbase graphically (it has an icon associated with the class in the knowledge base), and since each particular airbase in the knowledge base has an associated geographical location the Map Icon modality is selected. Then the system will display the airbase on the color-graphics map with additional information displayed in another modality. If the Nuernberg airbase is already displayed on the color map display, then the system would choose to blink the particular airbase icon as its way of pointing to the object and accompany this pointing action with a spoken response. If the Nuernberg airbase could be added to the current map, it would do so and direct the user's attention to the airbase icon as mentioned above. However, the Nuernberg airbase is outside of the region shown in the map display currently on the color CRT. Therefore the resources needed to present the Nuernberg airbase graphically are unavailable. The system must now decide how to modify the state of the resources to show the airbase. What map should be displayed?

In composing a new map on which to display the Nuernberg airbase, the system has some choices. These choices include: open a window on the color graphics display showing the area around the Nuernberg airbase, replace the old map on the CRT with a new area around the Nuernberg airbase, or compose a new map including both the old map and the region around the Nuernberg airbase.

An important guideline to which the CUBRICON system tries to adhere is to maintain the context of the user-computer dialogue. With regard to the graphic displays, this means that the system tries to retain the most recently discussed or mentioned objects on the displays so as to maintain continuity in the dialogue. The discourse focus space representations, discussed in Section 4.2.1 are the key knowledge sources in this process. The system composes a new map containing the objects that are on the old map as well as the Nuernberg airbase. The algorithm that the system uses to determine the boundary for a new map of this type is to determine the smallest rectangle that encloses the old objects on the current map as well as the new objects to be displayed and then add a small "border" area around all sides. This essentially extends the area shown to include both the old and new objects.

Another important guideline to which the CUBRICON system adheres is to maintain consistency throughout a display so as to prevent the user from making false inferences about what is or is not located within the region. In the case of our map display, this means that there should be consistency in the types of objects shown across the entire map. If SAMs are displayed in the old region, then they should be displayed in the newly added map area. Similarly for other types of objects. If this is not done, then the user would probably infer that there were no SAMs in the new area since he sees none on the display in the new area, when in reality there are SAMs in the new area. Figure 7-1 shows the new map display composed by CUBRICON in response to the user's input "Where is the Nuernberg airbase?" The rectangular outline within the map is used to indicate the previously displayed area. This provides graphic context: the new entities in the context of the previously displayed area.

Based on the information provided by the user/task model, CUBRICON knows the important attributes of each object. The table modality is selected to present this information. Guided by the consistency principle, the system also modifies the tabular presentation that is on the monochrome display to include the additional objects and their relevant attributes. The map and table displays are shown in Figure 7-1.

In this example, the CUBRICON system distinguishes between spoken and written (to a CRT display) NL. CUBRICON used graphic and deictic gestures with spoken NL only (not with written NL), since a pointing or graphic gesture needs to be temporally synchronized with the corresponding verbal phrase, allowing for multiple graphic gestures within any individual sentence. The coordination between a graphic gesture and its co-referring verbal phrase is lost if printed text is used instead of speech. Written NL was used however, when deictic/graphic expressions are not used, but, instead, definite descriptions are generated as noun phrased with sufficient specificity to hopefully avoid ambiguous references.

The user now asks the system a question phrased exactly like the previous question for purposes of comparison.

USER: "Where is the Stargard airbase?"

DEVICE CONFIGURATION:

The color graphics display contains a map displaying the Fulda Gap region and a table showing the important attributes of the objects is displayed on the map. The monochrome display contains a mission planning form.

CUBRICON:

Monochrome Display:

- No change

Speech Output with coordinated Color Graphics:

- The sentence "Its location is 120 miles east of the Fulda Gap region." is accompanied by the visual point gesture which blinks the window containing the Fulda Gap region.

This example illustrates the flexibility CUBRICON has in selecting from alternative presentation modalities and its ability to measure the relevance. Although this question is phrased exactly the same as the previous question, the CUBRICON response is totally different. The Stargard Airbase is well outside of the user's area of responsibility as represented in the CUBRICON knowledge base (ie. the task model). Therefore, CUBRICON judges that the Stargard airbase is less relevant than the current display and does not modify the color graphics display to present the information graphically. Instead, the Natural Language Prose modality is chosen and the system responds verbally without changing the current display.

8 Natural Language Output with Graphic Gestures and Graphic Expressions

Just as CUBRICON accepts natural language accompanied by deictic and graphic gestures during input, CUBRICON can generate multi-modal language output that combines natural language with *deictic gestures* and *graphic expressions* during output. An important feature of the CUBRICON design is that natural language and graphics are incorporated in a single language generator providing a unified multi-modal language with speech and graphics synchronized in real time. CUBRICON uses a GATN [Shapiro82] for multi-modal language generation. This GATN generates natural language from SNePS knowledge base structures.

An important aspect of the CUBRICON system is that it distinguishes between spoken and written (to a CRT display) NL. CUBRICON uses graphic and deictic gestures with *spoken NL* only (not with written NL), since a pointing or graphic gesture needs to be temporally synchronized with the corresponding verbal phrase, allowing for multiple graphic gestures within any individual sentence. The coordination between a graphic gesture and its co-referring verbal phrase is lost if printed text is used instead of speech. When using a pointing gesture, CUBRICON uses a terse NL phrase (e.g., "this SAM") accompanied by a pointing gesture to reference an object that is visible on one of the displays. When CUBRICON generates *written NL*, however, deictic/graphic expressions are not used, but, instead, definite descriptions are generated as noun phrases with sufficient specificity to hopefully avoid ambiguous references.

CUBRICON combines synchronized natural language and graphic expressions (1) for the presentation or expression of relative locative information and (2) for the presentation of space-time-dependent event sequences such as AF mission plans.

The next subsection reviews the component modalities before discussing the integrated use of natural language and graphics.

8.1 Component Modalities

This section discusses the generation of spoken and/or written natural language accompanied by graphic gestures and/or graphic expressions. These modalities are discussed briefly in the following subsections.

8.1.1 Speech

Speech output is produced by a stand-alone DECtalk speech synthesis device, controlled from the Symbolics Lisp machine via a serial port connection. The DECtalk system includes a full grapheme-to-phoneme conversion system for English and accepts normal orthography. The system provides for the specification of the pronunciation of individual words that need to be tailored for the specific domain (e.g., the lexical stress in "subMISsion" in comparison with "SUBmission"). The exception dictionary is uploaded at system initialization time and is stored in the synthesis device itself. We have used DECtalk's standard male voice for all speech output.

8.1.2 Written Language

Written natural language output is displayed mainly in the Natural Language Interaction Window (see Section 5.1.4) on the monochrome screen. This is the same window that is used to echo the user's multi-modal input. Natural language output is also displayed in dynamic text windows on the color graphics CRT during mission plan presentations.

8.1.3 Deictic Gestures

There are a number of deictic gestures that CUBRICON uses (see Section 9), depending on the type of object being pointed to, the dialogue context, and the modality in which the object is visually presented. The primitives that CUBRICON combines for deictic expressions are speech, blinking, highlighting, and graphic pointers. On a map window (see Section 10), *the system can highlight* individual icons by drawing a circle around them, intensifying their color, or flashing them. They may also be provided with a descriptive label. An icon can be pointed to by with the use of a "pointing text box." Regions of a displayed area can be pointed to by drawing a box around them or flashing the window border containing them. Pointing to an item on a table is done by drawing a box around the row containing it. Pointing to items on the form is accomplished by flashing the box around the appropriate field and putting its content in boldface type. More information on deictic gestures employed by the system can be found in Section 9).

8.1.4 Graphic Expressions

CUBRICON also uses graphic expressions in combination with natural language for information that is, at least partially, amenable to graphic presentation. In the current CUBRICON implementation, the type of information that falls in this category includes relative locative information and time-space-dependent event sequences. Graphic expressions are graphic illustrations that are displayed on the color graphics CRT.

8.2 Written Natural Language

In contrast to CUBRICON's use of spoken natural language for output, when written natural language (written to one of CUBRICON's CRT displays) is generated, deictic gestures are not used. If printed text is used instead of spoken output, the coordination between a graphic gesture and its co-referring phrase is lost. Written NL output is considered to be more permanent in nature than the transient spoken variety, and must therefore be more self-contained. Since the Natural Language Interaction window has a history mechanism, previous written natural language output can always be retrieved for reference purposes, and in this context there are no disambiguating deictic gestures available. Written natural language thus uses definite descriptions for noun phrases and locative phrases, so the intended referent can be determined from the language alone. In stead of saying "this SAM" accompanied by a pointing gesture, the system will generate a definite description as the noun phrase with sufficient specificity to avoid ambiguous references.

8.3 Natural Language with Deictic Gestures

Natural language and deictic gestures are controlled by a single multi-modal language generator which integrates and synchronizes them in real time. Just as the multi-modality of CUBRICON's input stream allows for cross-modal disambiguation of otherwise ambiguous phrases (Section 5.2.2.1), terse spoken phrases (e.g., "this SAM") generated by CUBRICON can be disambiguated by a simultaneous deictic gesture to the intended referent (e.g. an icon on a map window or an element of a table).

Deictic gestures are combined with appropriate natural language during output to guide the user's visual focus of attention. During language generation, in order to compose a reference for an object,

1. if the object is represented by an icon on the display, then CUBRICON generates a natural language expression for the object and a simultaneous coordinated graphic gesture that points to the icon.

   If the object has an individual name or identifier, then CUBRICON uses its name or identifier (e.g., "the Merseberg airbase") as the natural language part of the expression

   else CUBRICON generates an expression consisting of a demonstrative pronoun followed by the name of an appropriate class to which the object belongs (e.g., "this SAM", "these SAMs") as the natural language expression.

2. if the object (call it X) is not represented by an icon on the display, but is a component of such a visible object (call it Y), then CUBRICON generates a phrase that expresses

object X as a component of object Y and uses a combined deictic-verbal expression for object Y as described in the above case. For example, if CUBRICON is generating a reference for the runway of an air base called Merseberg and an icon for the air base is visible on the map (the air base as a whole is represented visibly, but not its parts), then the system generates the phrase "the runway of the Merseberg Airbase" with a simultaneous point gesture that is directed at the Merseberg air base icon on the map.

Frequently an object to which CUBRICON wants to point has a visible representation in more than one window on the CRTs. Therefore the system must select the visual representation(s) of the object (e.g., an icon, table entry, form slot entry) that it will use in its point gesture(s) from among the several candidates. The CUBRICON methodology for determining which of the object's visible representations to use in such a situation and the nature of the deictic gesture(s) generated are discussed in Section 9.

8.4 Natural Language with Graphic Expressions

The system combines *graphic expressions* with natural language output when the information to be expressed is, at least partially, amenable to graphic presentation. Information about the location of one object relative to the location of another falls into this category.

When generating relative locative information about some object (the figure object [Herskovits85]), CUBRICON selects an appropriate landmark as the ground object [Herskovits85], determines a spatial relationship between the figure and ground object, and generates a multi-modal expression for the locative information including the spatial relationship. When selecting the ground object, CUBRICON selects a landmark such as a city, border, or region, that is within the current map display (i.e., does not require a map transformation). If possible, the system uses a landmark that is in focus by virtue of its having been already used recently as a ground object. CUBRICON's discourse model (Section 4.2.2) includes a representation of the attentional focus space of the dialogue, including a main focus list of entities and propositions that have been expressed by CUBRICON or by the user via multi-modal language. If a new landmark must be used as a ground object, then the system selects the landmark that is nearest the figure object. CUBRICON derives a spatial relation between the ground object and figure object that it represents in its knowledge base. This relation includes (1) the direction from the ground object to the figure object and (2) the distance, if the distance is greater than 0.04 times the window width. If the distance is less than 0.04 times the window width, then the figure object appears to be right next to the ground object. This criterion for deciding whether to include distance as part of the relation reflects the tendency for people to omit a distance measure when the distance is small relative to the geographic area under discussion and to say something like "just northeast of" instead of stating a distance explicitly.

As an illustrative example, the user may ask about the location of a particular object, such as the Fritz Steel plant. CUBRICON then uses the steel plant as the figure object, selects a ground object, and derives a spatial relation between ground object and figure object as discussed above. The multi-modal response is given below.

USER: "Where is the Fritz Steel plant?"

CUBRICON: "The Fritz Steel plant is located here <point>, 45 miles southwest of Dresden <graphic-expression>."

The <point> consists of a gesture that points out the Fritz Steel plant icon to the user via a gesture that uses a combination of blinking, highlighting, circling the icon and the attachment of a pointing label-box that identifies the icon. The <graphic-expression> is a visual representation of the spatial relation between the figure object (Fritz steel plant) and the ground object (Dresden city), consisting of an arrow drawn from the Dresden city icon to the steel plant icon, a label stating the distance, and a label identifying the city (the steel plant should already be labeled).

When presented synchronously in real time, the combination of spoken natural language and graphic illustration forms a fluid multi-modal language for the presentation of spatially oriented information.

8.5 Multi-Modal Language Generation for Space-Time-Dependent Events

CUBRICON includes a discourse level generation component that structures a multi-sentence presentation that is designed for the explanation or presentation of events that are sequenced in space and time. The prototype implementation has been applied to the presentation of OCA missions that include flight path traversals with related events such as the striking of targets and the airborne refueling of strike aircraft. This multi-modal presentation component uses two critical concepts: a task hierarchy and granularity. These are discussed in the following subsections. The last subsection discusses the discourse structure and output composition.

8.5.1 Task Hierarchy

CUBRICON's multi-modal presentation of space-time-dependent events is dependent on the representation of a task hierarchy and the interrelationships and properties of the tasks. This information is used to define the concept of granularity in the CUBRICON system. For the mission planning application domain used in this project, the tasks represented in the hierarchy in CUBRICON's knowledge base are tactical AF missions. The CUBRICON multi-modal language components depend on the knowledge base structures discussed in Section 4.1. These knowledge base representations use generic structures and relations (e.g., sub-task, super-task) so that the multi-modal generation components are applicable to other domains and across different mission types within this domain. The same is true of the knowledge base representation of information about missions in the form of properties.

For this Air Force application domain, CUBRICON's knowledge base includes information about different types of missions and their interrelationships and properties. The primary mission types that are accounted for in CUBRICON's current knowledge base are shown in the hierarchy of Figure 8-1. The dashed lines in this figure represent the relation of parent to child (or super-task to sub-task). The SNePS knowledge base representation of this relation is discussed in Section 4.1. The knowledge base also includes information about the details of each mission. This information is represented in the form of properties of the different missions. For an OCA mission, for example, this information includes the the airbase from which the mission aircraft originate, the aircraft unit flying the mission, the type of aircraft used, the time of departure of the aircraft, and the target(s) of the mission.

```
                Package
                /     \
                /      \
     OCA mission        RFL mission
         /   \             \
        /     \             \
STK mission    \         STN mission
                \         /     \
                 \       /       \
                SVC mission     ORB mission
```

Figure 8-1: Hierarchy of Mission Types

A SVC (service) mission consists of the refueling of a strike aircraft by an orbiting refueling tanker. The properties of an SVC mission include the start time of the SVC, the length of time that the servicing takes (callcd the duration of the servicing), and the amount of fuel disbursed to the strike aircraft.

A RFL (refueling) mission consists of a tanker aircraft flying a certain route in order to service one or more strike aircraft.

An STN (station) mission consists of a tanker aircraft stationing itself, by orbiting for a certain period of time, at a certain location in order to service one or more strike aircraft.

A STK (strike) mission consists of a strike aircraft striking its target. Properties of the STK mission include the identifier of the target, target location, and the time of the target strike.

8.5.2 Granularity

If a system has a knowledge/data base that contains a voluminous amount of information, the system must have a method of selecting the appropriate information to present in response to user requests if the response is not well constrained by the request itself. A factor that can play an important role in selecting appropriate information is that of granularity.

One of the main factors used by the event presentation component is that of granularity. The CUBRICON design includes several levels of granularity: very coarse, coarse, default, fine, very fine, and local. Of these, coarse, default, and local are used by the multi-modal event presentation component. Coarse is the granularity level used for the introduction, the main body of the presentation uses default granularity, and local granularity is used for the presentation of ancillary events.

For any task (mission) type, granularity for presentation purposes is defined in terms of

1. the relative level of the tasks (missions) in the task (mission) hierarchy and
2. the importance of the properties of a task (mission).

The task (mission) hierarchy was discussed briefly in the previous section. By relative level in the hierarchy we mean the relation of parent, self, and child task (mission) with respect to the mission to be presented. The CUBRICON definition of granularity uses these three relations between tasks (missions) along with the classes of properties of a mission to be presented.

The properties of each task (mission) type are divided into two groups: major and minor. These two groups represent the more important properties of a mission and the lesser important properties, respectively. This categorization of properties is part of the task (mission) model. A special property of a task (mission) used in the definition of granularity is that of the name of the task (mission).

The following table defines the concept of granularity for an event sequence presentation. Each column of the table defines a granularity level or type. Let X represent the task (mission) to be presented. Each type (level) of granularity specifies the information to be presented as part of the presentation of task (mission) X. The solid bullet-marks identify the tasks (missions) related to X that are to be part of the presentation. For example, for coarse granularity, the parents of X and X itself are part of the presentation. The unfilled bullet-marks identify the types of properties to be presented for each of these related tasks (missions). Continuing the example for coarse granularity, for each parent of X, the name property and all major properties are presented as well as the name and major properties of X itself.

Granularity

Very coarse	Coarse	Default	Fine	Very Fine	Local
• Parents ○ name	• Parents ○ name ○ major	• Parents ○ name ○ major	• Parents ○ name ○ major	• Parents ○ name ○ major ○ minor	
• Node ○ name	• Node ○ name ○ major	• Node ○ name ○ major ○ minor	• Node ○ name ○ major ○ minor	• Node ○ name ○ major ○ minor	• Node ○ name ○ major ○ minor
	• Children ○ name	• Children ○ name	• Children ○ name ○ major	• Children ○ name ○ major ○ minor	

8.5.3 Discourse Structure and Output Composition

This section discusses the CUBRICON language generation component that composes multimodal output for *a collection of compound space-time-dependent events.* For the AF mission planning domain of this project, the compound events are OCA missions. These missions are composed of sub-missions with their flight paths forming *event sequences.*

The presentation of a collection of compound events is subdivided into two parts by the discourse grammar: an introduction and a main body. The introduction provides summary highlights of the events and the main body provides a detailed presentation of the event collection. Within the main body of the discourse, the main thread of the presentation is that of the space-and-time-dependent event sequences. For an OCA mission presentation, this main thread (event sequence) is the (flight) path traversal. A path traversal sequence is composed of events, each of which consists of traversing a leg (line segment) of the path (the path is represented as a polygonal path). We will call the endpoints of each of the line segments "waypoints", which seems to be the commonly used terminology in the military. Each of these segment traversal events takes place in both space and time: the waypoints of each segment have coordinates in terms of latitude/longitude and each waypoint has an associated time at which the aircraft traversing the path is scheduled to reach the waypoint. Figure 8-2 illustrates the knowledge base representation of a path (line) segment. The segments of the path are sequenced by their common endpoints. That is the last (second) endpoint of line segment S is the first endpoint of line segment $S + 1$.

Important ancillary events occur during the main space-time-dependent event sequence. The

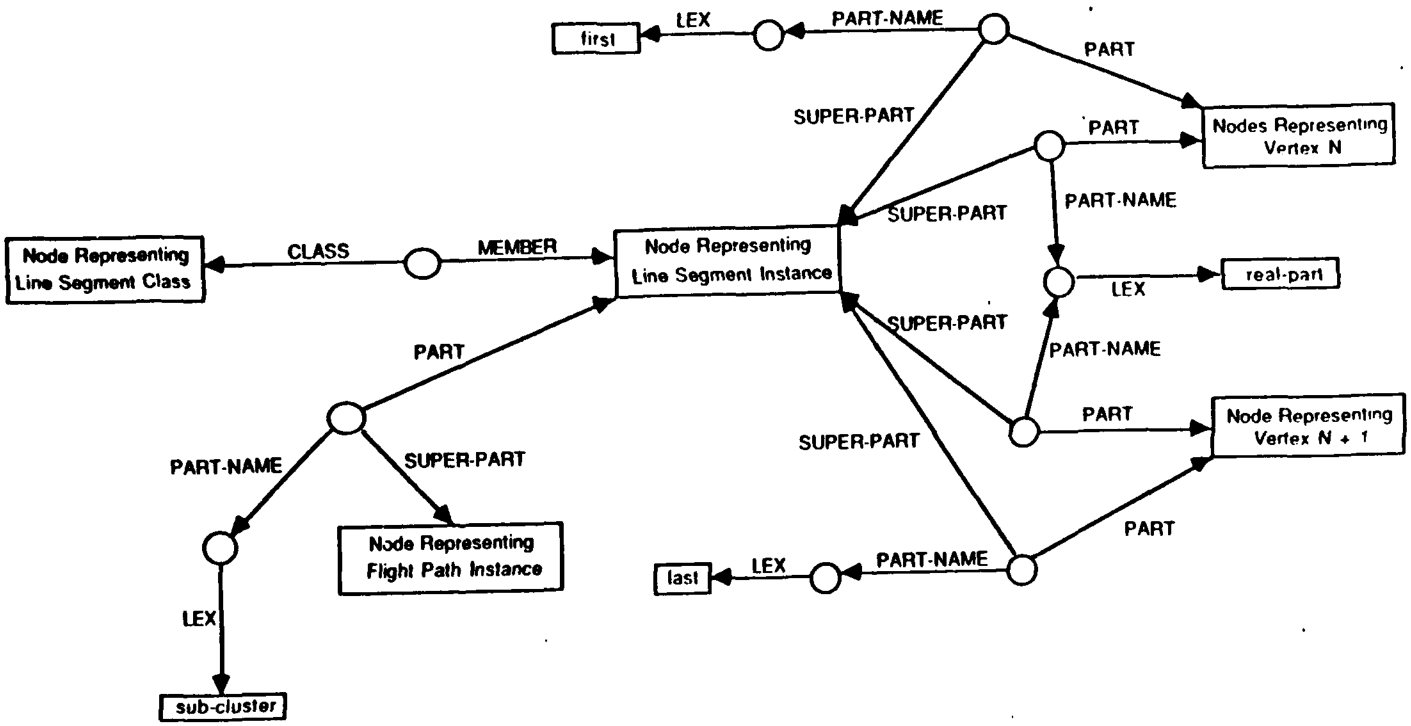

first
LEX
PART-NAME
PART
SUPER-PART
PART
Nodes Representing Vertex N
SUPER-PART
PART-NAME
Node Representing Line Segment Class
CLASS
MEMBER
Node Representing Line Segment Instance
LEX
real-part
SUPER-PART
PART-NAME
PART
PART
Node Representing Vertex N + 1
SUPER-PART
PART-NAME
SUPER-PART
Node Representing Flight Path Instance
PART
last
LEX
PART-NAME
LEX
sub-cluster

Figure 8-2: Path Segment Knowledge Base Representation

ancillary events are sub-missions or sub-tasks of the main OCA mission, including tasks such as the striking of targets and the airborne refueling of strike aircraft. After each leg (segment) of the path is presented, the NL generator determines whether there are any ancillary events whose occurrence time or start time precedes the occurrence time of the next waypoint. If so, these ancillary events are presented in order of their time of occurrence as represented in the knowledge base. Otherwise, the next leg or segment of the path is presented.

In order to select the appropriate information to present in the introductory presentation, the system uses the coarse level of granularity as defined in the previous subsection. In selecting the information for the main body of the event presentation, default granularity is used. When the system selects the appropriate information to present for the ancillary events, it uses the local granularity level. Thus, for example, when the presentation has reached the point at which the task of striking the target is to be presented, the system uses local granularity in retrieving the appropriate information from the knowledge base. Thus for a strike target task (STK mission), the system selects only the properties (both major and minor) of the STK mission itself, and says nothing about the parent task (mission) or child tasks (missions).

As stated previously, for the presentation of a collection of OCA missions, the main thread of the presentation is the presentation of the group of event sequences. For this application domain, each event sequence is a (flight) path traversal. The following pseudo-code summarizes the top-level processing logic of the portion of the multi-modal generation grammar that handles the presentation of a group of event sequences.

```
Express Collection of Space-Time-Dependent Event Sequences:
Retrieve the first event of each event sequence from the KB
Initialize current_event to the event from this group of events
        with earliest time of occurrence
Initialize other_events to the rest of this list of events
Initialize done to false
WHILE not done DO
    IF current_event is the last event of some event sequence
     AND there are no other_events THEN
        Express-Transition-Event
        Generate final output for the collection of event sequences
        Set done to true
    ELSE
        Express-Transition-Event
        Continue-Sequence
    ENDIF
ENDDO
```

The following pseudo-code expands on the logic of the grammar for expressing a transition event. For the OCA mission presentation application, the startup of each event sequence consists of expressing information about the aircraft departing the origin airbase in spoken natural language with accompanying graphic gestures shown on the map display. Relevant information includes the location of the origin airbase, the time of departure, and the unit that is flying the mission. For this OCA mission application, a transition event is the traversal of a leg of a flight path. The transition between the previous event (waypoint) and current event (waypoint) is presented in animated graphics showing an strike aircraft icon traversing the leg of the path leaving a trace represented by a growing directed line segment that extends as the aircraft icon moves. For each waypoint, the planned time at which the aircraft arrives at the waypoint is shown as a label near the waypoint icon. The final output for an event sequence consists of information about the arrival of the aircraft back at the origin airbase. This information is expressed in spoken natural language accompanied by graphic gestures shown on the color graphics map display. The following pseudo-code presents the logic of that portion of the grammar that handles the expression of transition events.

Express-Transition-Event:

```
IF current_event is the first event of some sequence
   Express the startup of the event sequence
ELSEIF current_event is the last event of some sequence
         Express transition between previous_event and current_event
         Generate final output for the sequence
    ELSE
         Express transition between previous_event and current_event
    ENDIF
ENDIF
```

After each transition event is expressed, the relevant and timely ancillary events are expressed. As indicated previously, for an OCA mission presentation, these ancillary events are the sub-tasks of striking the targets and the refueling of the strike aircraft. For each ancillary event, local granularity is used to select the information to present about each ancillary event. The information is generated in multi-modal language. The following pseudo-code summarizes the logic for continuing an event sequence after a transition event has been expressed.

Continue-Sequence:

```
Add successor of current_event to other_events
Set next_event to the event from other_events with earliest
      time of occurrence and remove it from other_events
```

```
IF the occurrence times of some ancillary events are before next_event THEN
    Generate output for the ancillary events
ENDIF
Set current_event to the value of next_event
Set previous_event to the predecessor of current_event.
```

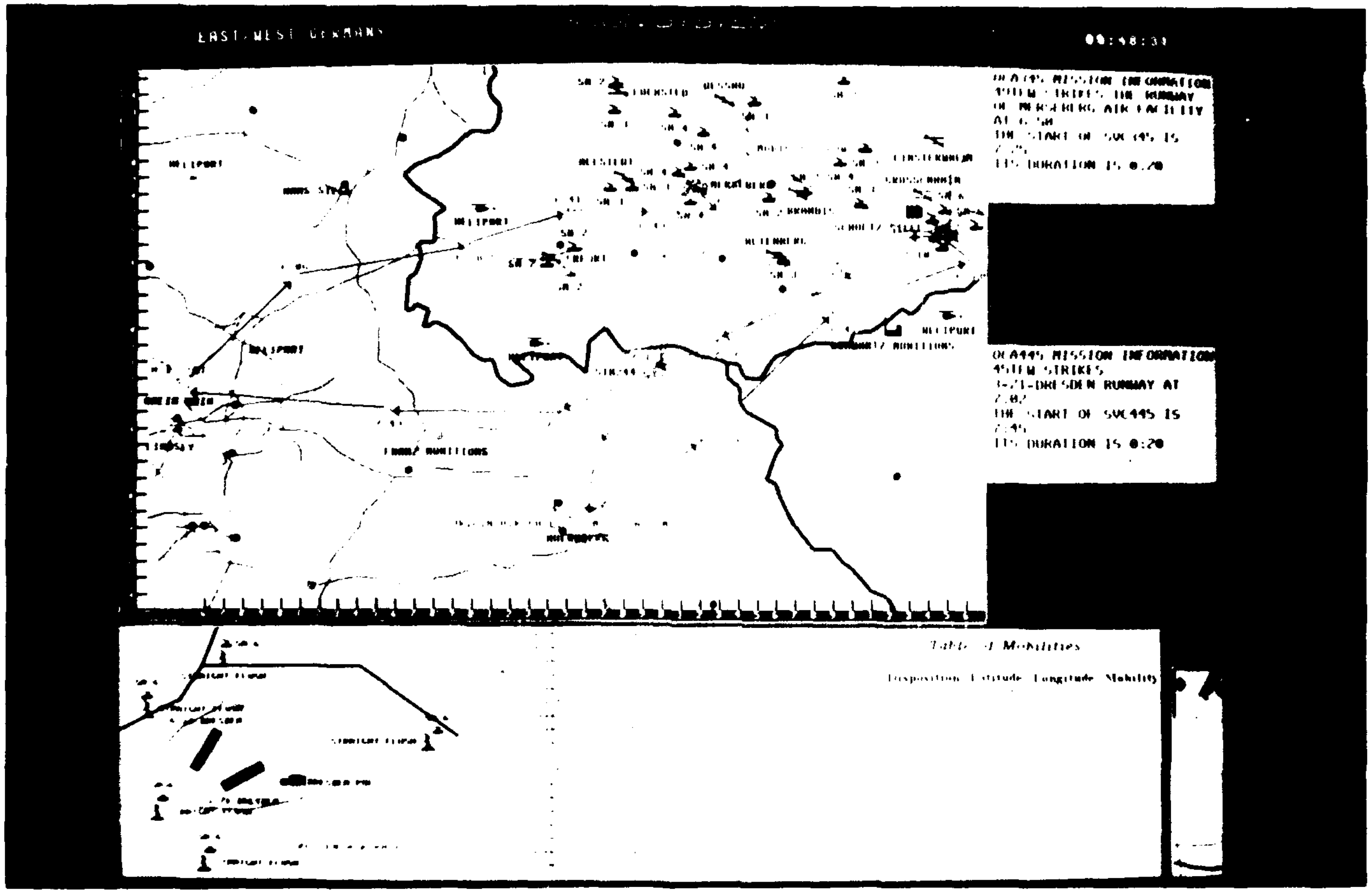
The CUBRICON natural language generator composes and presents both spoken and written (on the CRT) natural language. As discussed in the previous section, when graphic gestures and expressions are used, spoken NL is generated so that the natural language and graphics are temporally synchronized. The system also produces a written version of its NL utterances. During the multi-modal presentation of a collection of complex events such as mission plans, the written version of the natural language is presented in special dynamic text windows (one per OCA mission) on the color graphics CRT. Figure 8-3 shows the color graphics display after the conclusion of a multi-modal presentation of a collection of space-time-dependent events, which are OCA missions in this application.

The multi-modal generation grammar is designed to handle all of the different types of SNePS knowledge base structures that are in the CUBRICON knowledge base. Section 4.1 discusses the major CUBRICON knowledge base structures. For each type of KB structure (the KB structures are the input to the generation grammar), there is a portion of the generation grammar that generates natural language with synchronized graphic gestures and/or expressions for that type of structure. These knowledge base structures are generic so that the representations and multi-modal generation components are applicable to other domains and across different mission types within this AF mission planning domain.

The multi-modal presentation of a collection of OCA missions is fairly lengthy and complex. The following summarizes such a presentation from the perspective of what a viewer sees and hears:

1. As an introduction to the presentation, for each OCA (Offensive Counter Air) mission, its ID number, its package number (a package is a set of related missions), the origin (departure) air base, and the OCA's submissions (strike and refueling missions) are summarized in speech and written language (on the Natural Language Interaction Window), accompanied by temporally synchronized pointing gestures to the corresponding items (as they are presented) on the mission form, which is on the monochrome display. For each OCA mission, a mission information window is initialized on the color graphics display, next to the relevant map window. It will be used during the rest of the presentation to summarize important information in a written form.

2. Information about the mission flight departure is presented via speech with highlighting of the origin air base on the map window as it is mentioned and accompanied by the display of a label reading "origin airbase."



OCA495 MISSION INFORMATION
450TFW STRIKES THE RUNWAY
OF MERSEBERG AIR FACILITY
AT 6:50
THE START OF SVC495 IS
7:45
ITS DURATION IS 0:20

OCA445 MISSION INFORMATION
45TFW STRIKES
3-21-DRESDEN RUNWAY AT
7:02
THE START OF SVC445 IS
7:45
ITS DURATION IS 0:20

Table of Mobilities

Disposition Latitude Longitude Mobility

Figure 8-3: Color Graphics Display After Presentation of Missions

3. One by one, the segments making up the different (polygonal) flight paths are displayed in an animated manner on the map window so as to simulate simultaneous flight path traversal. Each flight path segment is presented at its appropriate time according to the time on waypoint for its second endpoint. For each flight path, an aircraft icon moves from waypoint to waypoint as a directed line segment grows to represent the particular leg of the flight path. Each waypoint is labeled so as to indicate the time on waypoint.

4. For each mission, when the target of the mission is reached, it is identified via spoken NL with a synchronized deictic gesture consisting of blinking/highlighting its icon on the map window, highlighting the information on the form window, and any tables that include the information. The time on target is also presented. The target information is also summarized in the mission information window on the color graphics CRT.

5. The presentation continues with flight path segments as before.

6. For each mission, when the refueling location is reached on the map window, information about the refueling mission presented in spoken natural language with synchronized simultaneous deictic gestures pointing to the relevant information representations on the various windows (i.e., map, form, and tables). The information is also summarized in the mission information window.

7. The presentation continues with flight path segments as before.

8. For each mission, when the aircraft arrives back at the origin air base, the completion of the presentation is announced in speech.

9 Deictic Gestures

Multi-modal deictic expressions can be used very effectively by a human-computer interface system to assist the user in locating and tracking the visual focus of attention. Deictic expressions can consist of speech, printed natural language, various graphic techniques, or combinations thereof. Such expressions can target one or more small objects such as icons on a map, or larger objects such as a window on a display or a particular CRT. A deictic expression can consist of one or more sentences or may be a phrase within a sentence. This section describes the use of deictic gestures in multi-modal output generation. The use of deictic gestures in multi-modal language understanding is described in Section 5.

9.1 Primitives for Deictic Expressions

CUBRICON is able to present information in a variety of combinations of modalities (eg. maps, tables, forms, written natural language, spoken natural language and pointing gestures), as discussed in Section 7. The visual types of objects to which CUBRICON can point are: CRT, window, geographic region, table row or element, form entry, and icon group, discussed in Section 9.2.

The following *primitive deictic gestures* are utilized by CUBRICON individually or in various combinations in appropriate circumstances. The methodology for generating deictic gestures composed of these primitives is discussed in Section 9.2.

- *Speech.* A verbal pointing expression, such as "Look at the color-graphics screen", can be used to draw the user's attention to this particular CRT.
- *Blinking.* The following applications of blinking are used as primitive deictic gestures: the simultaneous blinking of a group of one or more icons, the blinking or flashing of the border of a window, the blinking of an entry in a form.
- *Highlighting.* An object can be highlighted by changing its color, intensifying its color, enclosing it with a circle and/or a rectangle.
- *Graphic pointer.* A graphic pointer in the form of an arrow or other graphic symbol can be used as a primitive deictic gesture. CUBRICON's primitive, called a *labeled-pointer* or *text-box-pointer*, attaches a label or text-box to an arrow which is directed at the visual representation of the target of the point gesture. This type of graphic pointer can, in fact, have more than one arrow emanating from the same text-box for pointing to objects that are of the same type or have some attribute in common. Figure 9-1 illustrates such a *labeled-pointer* to call attention to the Nuernberg airbase on a cluttered map display.

9.2 Generating Deictic Gestures

The following paragraphs list the techniques that CUBRICON uses to point at different *types of presentation objects*. In general, CUBRICON's deictic gestures can take the form of speech output alone, graphics alone, or speech output accompanied by synchronized graphic gestures. The latter is the most frequently used. In general, CUBRICON uses deictic gestures whenever the information or object to be presented (referenced) is visible in the form of a presentation object on one of the displays. For each of the following presentation objects, the technique for pointing at it is listed.

- *CRT*. CUBRICON produces a spoken deictic expression (e.g., "Look at the color-graphics screen") to direct the user's attention to the appropriate screen (CRT) when a new window is created on one of the CRTs containing information requested by the user.

- *Window*. When the information requested by the user takes up most of the area of an existing window, then the system directs the user's attention to the particular window by a spoken deictic expression (e.g., "This window contains the aimpoints within Dresden.") with an accompanying visual gesture briefly blinking the border around the window and then leaving the border in a highlighted state.

- *Region within a window*. When the information requested by the user is presented in a subarea of a window, the system directs the user's attention to the area using speech (eg. "This window contains the Fulda Gap region.") with an accompanying visual gesture which consists of highlighting a rectangle that is temporarily displayed on the window around the subarea.

- *Map Icon group*. CUBRICON can point at map icons by blinking the icons, highlighting them via the various techniques listed in the previous subsection, or by using text-box-pointers (also mentioned in the previous subsection). As discussed in Section 8, map icon pointing typically accompanies natural language generation when objects to be referenced are visible on one or more of the map displays. Currently,

  1. if an object is represented by an icon on the display, then CUBRICON generates a NL expression for the object and a simultaneous coordinated graphic gesture that points to its icon.

     If the object has an individual name or identifier, then CUBRICON uses its name or identifier (e.g., "the Merseberg airbase") as the NL expression

     else CUBRICON generates an expression consisting of a demonstrative pronoun followed by the name of an appropriate class to which the object belongs (e.g., "this SAM", "these SAMs") as the NL expression.

   2. if the object (call it X) is not represented by an icon on the display, but is a component of such a visible object (call it Y), then CUBRICON generates a phrase that expresses object X as a component of object Y and uses a combined deictic-verbal expression for object Y as described in the above case. For example, if CUBRICON is generating a reference for the runway of an airbase called Merseberg and an icon for the airbase is visible on the map (the airbase as a whole is represented visibly, but not its parts), then CUBRICON generates the phrase "the runway of the Merseberg airbase" with a simultaneous point gesture that is directed at the Merseberg airbase icon on the map.

- *Form Slot.* When the information requested by the user is presented in one of the Package Worksheet form slots, then CUBRICON highlights the slot. This type of pointing accompanies natural language generation if an object to be referenced has a visible representation in one or more of the form slots.

- *Table Row or Element.* When the information requested by the user is presented in the form of a table row/element, CUBRICON highlights the row/element. This type of pointing can accompany natural language generation if an object to be referenced has a visible representation as a row of one or more tables on the displays.

It is frequently the case that an object to which CUBRICON wants to point has a visible representation in more than one window on the CRTs. Therefore the system must select the visual representation(s) of the object (e.g., icon, table entry, form slot entry) that it will use in its point gesture(s) from among the several candidates. The current CUBRICON methodology is to point out all the object's visible representations, but to use a strong pointing gesture (e.g., blink the icon to attract the user's attention and add a pointing text-box) for the most significant or relevant representations and weak non-distracting gestures (e.g., just highlight the visible representation) for the less significant ones. In order to select the most relevant visible representations from among all the candidates, CUBRICON:

1. selects all the windows which contain a visible representation of the object.

2. filters out any windows which are not active or not exposed.

3. if there are exposed windows containing a visible representation of the object, then CUBRICON uses all of these representations as objects of weak deictic gestures and selects the visible representation in the most important or salient window as the target of a strong deictic gesture. The selection of most important window when generating deictic gestures is based on both the numeric value assigned to each window which measures importance (see Section 13.3.4) and how important a window is to the user's request. For example, if the user requests locative information, then map windows are considered the window type most appropriate for the strong deictic gesture and the map window with the maximum window importance value is selected.

4. if there are no exposed windows displaying the object's visible representation, then CUBRICON determines the most important active (see Section 13.3.4) de-exposed window displaying the object. CUBRICON exposes this window and uses the representation of the object in this window in a strong deictic gesture.

9.3 Multi-Modal Examples

9.3.1 First Example

In the first example, spoken deictic expressions are used as well as a multi-modal deictic expression to present locative information on the map and weak deictic gestures are presented on a table row/element. For this example, suppose that the color graphics CRT shows a map of the Fulda Gap region and that the Nuernberg airbase is not on the displayed map since it is outside of the boundary of the region. Figure 9-1 shows the displays as they appear at the end of this example.

USER Spoken or Typed Input: "Where is the Nuernberg airbase?"

CUBRICON:

Decision Processing: By retrieving the location of the Nuernberg airbase from the knowledge base and consulting the discourse/display model, it is determined that the airbase is not on the current map display. To maintain context and continuity in the user-computer dialogue, the current map is expanded to include Nuernberg rather than replace it with a different map of the area around Nuernberg. Section 10 discusses the CUBRICON methodology for expanding a map (zoom out).

Spoken Deictic Expression: "The map on the color graphics screen is being expanded to include the Nuernberg airbase."

Color Graphics Display: Map of the Fulda Gap region is expanded to include the area around the Nuernberg airbase. A rectangular polygon (see Figure 9-1) is added to the display to show the boundary of the previously displayed region in order to show the new area in the context of the previously displayed area.

Spoken Deictic Expression with coordinated Color Graphics to direct the user's attention to the Nuernberg airbase: "Its location is here <point> 50 miles southeast of the East-West Germany border." The word "here" is accompanied by a visual point gesture in the form of blinking the airbase icon, then highlighting it and enclosing it with a circle and adding a pointing text box labeled with "Nuernberg Airbase", a text-box identifying the East-West Germany border and a pointing arrow indicating the shortest distance from the East-West Germany border to Nuernberg airbase. Section 8 discusses CUBRICON's presentation of relative locative information in more detail.

Decision Processing: Guided by the consistency principle, modify the tabular presentation on the monochrome display so that all the objects on the map display are listed. The discourse/display model is updated with the new contents of this window.

Spoken Deictic Expression: "The corresponding table is being presented on the monochrome screen."

Monochrome Graphics Display: A table containing relevant entity attributes is displayed. This is an expanded version of the table that accompanied the previous display of the Fulda Gap region. It has been expanded to include the new entities added to the map covering the extended area.

Table Row/Element Deictic Expression: The table row/element containing the Nuernberg airbase location is highlighted.

9.3.2 Second Example

This example illustrates the generation of deictic gestures for an icon group and a table row/element.

USER Spoken or Typed Input: "What is the mobility of these <point>$_1$<point>$_2$<point>$_3$?" where each <point>$_i$ points to a SAM icon on the map via the mouse device. Note: For CUBRICON's treatment of multi-modal input, see Section 5.2.

CUBRICON:

Decision Process: Suppose that the first SAM referenced by the user has high mobility, the second has low mobility, and the third has high mobility according to CUBRICON's knowledge base. Natural language prose is selected as the primary modality for the response using the CUBRICON modality selection criteria (see Section 7.1). According to the generation methodology of Section 9.2 for icon groups, since the SAMs do not have individual names and are visible on the display, CUBRICON generates "these SAMs" or "this SAM" with a simultaneous visible point gesture for each SAM group mentioned in the response. For the user's subsequent reference, a terse summary version of the response information is printed on the display near each SAM icon. In addition, CUBRICON determines if any of the SAMs referenced by the user are visible in any of the tables in thc form of row/elements. If so, then CUBRICON highlights each of these row/elements.

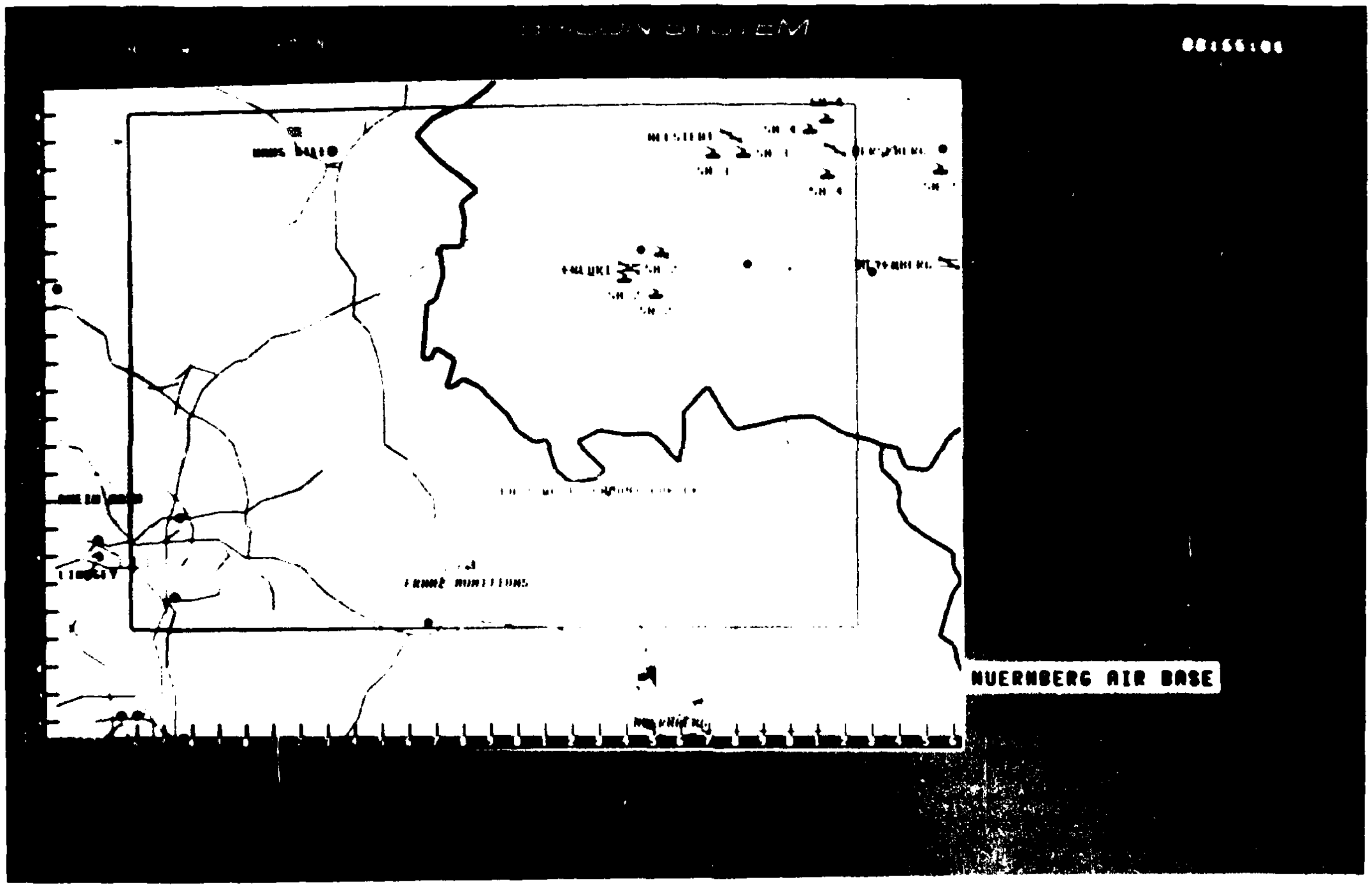
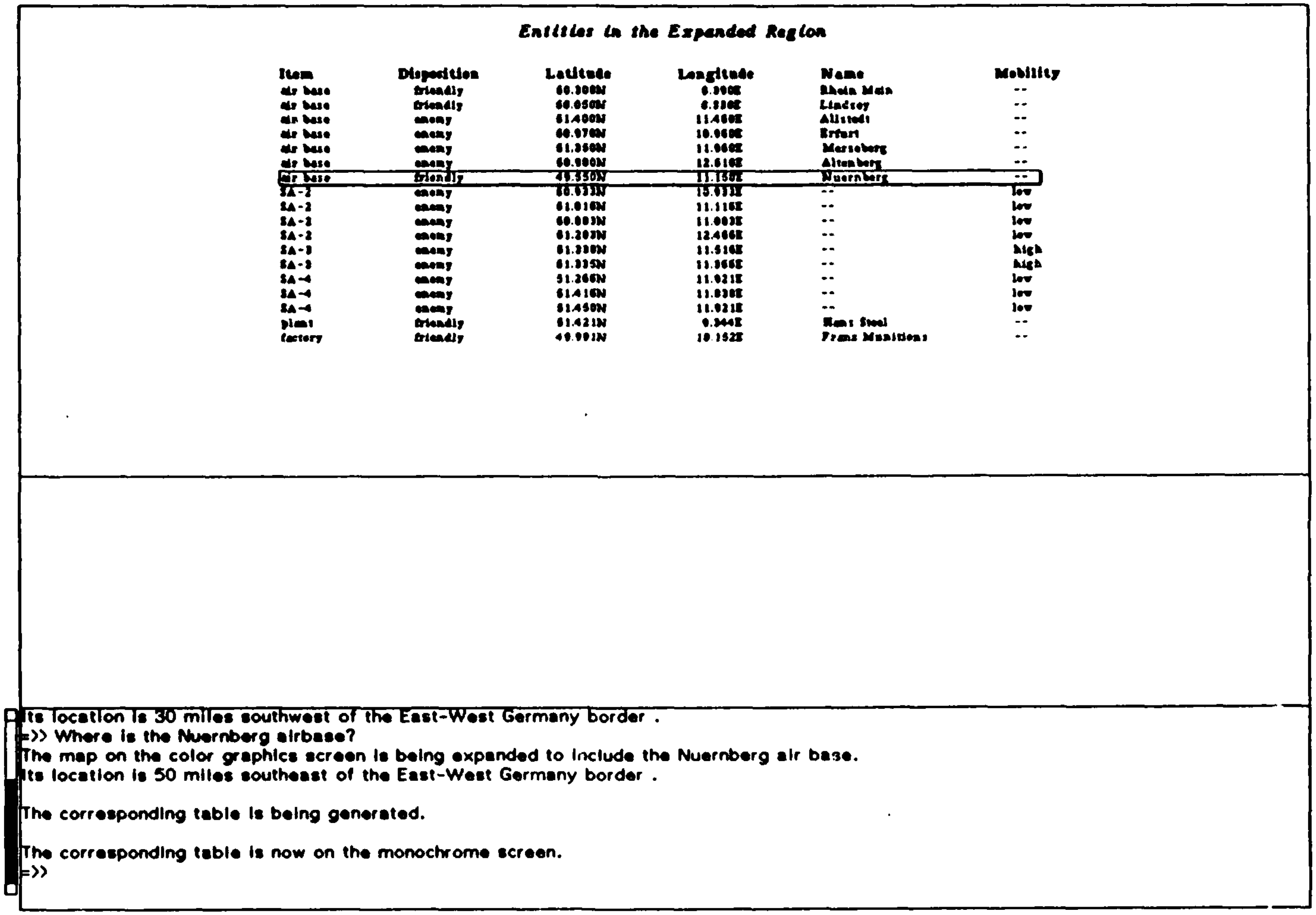


Figure 9-1a: The Displays After a Location Request



Entities in the Expanded Region

Item	Disposition	Latitude	Longitude	Name	Mobility
air base	friendly	50.100N	8.590E	Rhein Main	--
air base	friendly	50.050N	8.330E	Lindsey	--
air base	enemy	51.400N	11.460E	Allstedt	--
air base	enemy	50.970N	10.960E	Erfurt	--
air base	enemy	51.350N	11.960E	Merseberg	--
air base	enemy	50.980N	12.610E	Altenberg	--
air base	friendly	49.550N	11.150E	Nuernberg	--
SA-2	enemy	50.933N	15.033E	--	low
SA-2	enemy	51.016N	11.116E	--	low
SA-2	enemy	50.883N	11.003E	--	low
SA-2	enemy	51.203N	12.466E	--	low
SA-3	enemy	51.330N	11.516E	--	high
SA-3	enemy	51.335N	11.366E	--	high
SA-4	enemy	51.266N	11.021E	--	low
SA-4	enemy	51.416N	11.830E	--	low
SA-4	enemy	51.450N	11.921E	--	low
plant	friendly	51.421N	9.344E	Hans Steel	--
factory	friendly	49.991N	10.152E	Frans Munitions	--

```
Its location is 30 miles southwest of the East-West Germany border .
=>> Where is the Nuernberg airbase?
The map on the color graphics screen is being expanded to include the Nuernberg air base.
Its location is 50 miles southeast of the East-West Germany border .

The corresponding table is being generated.

The corresponding table is now on the monochrome screen.
=>>
```

Figure 9-1b: The Displays After a Location Request

Speech Output with Synchronized Graphics: "The mobility of these <point-to-icon>$_1$ <point-to-icon>$_3$ SAMs are high and the mobility of this <point-to-icon>$_2$ SAM is low." Each <point-to-icon>$_i$ represents a pointing gesture consisting of briefly blinking the map icon and then leaving the icon highlighted and circled. Next to each SAM icon, a small text box is added containing the expression "mobility: high" or "mobility: low", as appropriate for the particular SAM.

Table Row/Element Deictic Expression: The appropriate rows of any tables including the information are highlighted also.

9.3.3 Third Example

The third example illustrates CUBRICON's output generation process in the event that an entity to be expressed has no representation that is visible on one of the displays, but it *is part of* such an entity. This contrasts with the previous example in which the entity itself had a visible representation. Furthermore, in this example the entity has an individual name. The user has asked CUBRICON to "Present the OCA345 mission plan" and, as part of this presentation, CUBRICON must explain what the target of the STK345 sub-mission is. In this case, the target is the runway of an airbase. The runway itself has no visible representation on the map display, but the airbase, of which it is a part, is visible in the form of an airbase icon on the map. Therefore, CUBRICON expresses the information as follows:

"49tfw strikes the runway of the Merseberg air facility <point-to-air-facility> at 6:50."

The <point-to-air-facility> consists of blinking the airbase icon and then leaving the icon in a highlighted state. This natural language response is produced via speech output with the point gesture being produced simultaneously with the noun phrase "the Merseberg air facility". Speech output and graphics are always synchronized in the CUBRICON system. This output generation is shown in Figure 8-3. The printed version of the text is shown in the text window on the upper right side of the screen.

10 Map Modality

Color graphics is preferred by CUBRICON and is selected whenever the information to be represented includes spacial relationships. In particular, for the current CUBRICON implementation, this means that the map modality is preferred for entities that have regional or coordinate attributes and the existence of iconic symbols associated with the entities to be presented.

The map modality results in the creation or transformation of one of two types of maps: (1) geographic maps or (2) part-whole decomposition maps. A geographic map displays regional and coordinate information, whereas a part-whole decomposition map is a schematic diagram depicting the components of an object. Several operations are performed on geographic maps: map creation, zoom out, zoom in, and pan. Part-whole decomposition maps are created and not modified afterwards, however.

10.1 Geographic Maps

An important aspect of CUBRICON's processing capability is its decision-making logic for deciding when and how to create and transform geographic map displays. CUBRICON dynamically composes geographic map displays including the determination of the boundary of the region to be displayed and selection of the relevant entities to display in the region.

10.1.1 Map Composition

After selecting a map modality CUBRICON, must decide when and how to compose map displays. The steps involved in composing a map display are:

1. Determine what type of map transformation to perform.
2. Determine the objects to display on a map.
3. Determine the boundary of the region to be displayed.
4. Extend the boundary of the region vertically or horizontally so that one vertical kilometer is displayed as the same distance as one horizontal kilometer.
5. Display the map in the appropriate window with the appropriate icons, colors, and labels.

Relevancy is a critical factor for an HCI to consider when determining what information to present and how to present it to the user. In CUBRICON's map manipulation process, relevancy plays an important role determining both what type of map transformation to perform and what objects to display on a map.

10.1.1.1 Determining the Map Transformation One aspect of relevancy in the CUBRICON system pertains to the user's task. CUBRICON keeps track of the task that the user is working on and registers the user's transition from one task to another. CUBRICON's process for deciding on an appropriate map transformation takes into consideration whether or not the user's task has just changed. If the user's task has not changed, then CUBRICON assumes that the current main map configuration is still relevant and tries to keep it in the same window, subject to some expansion or contraction. If the user's task has just changed, however, then CUBRICON will move the map from the main window to a secondary window and "repaint" the main window with a new appropriate map area.

The following list provides the criteria that CUBRICON uses to select each of the map transformations.

- **Create**

  Selected if (1) no map is on the screen and (2) either objects with geographic locations are to be presented or a geographic region is to be presented.

- **Expand or Zoom Out**

  Selected if (1) the user's task has not changed and (2) *objects with geographic locations* are to be presented or a geographic region is to be presented and (3) the objects/region to be presented are/is not located within the boundary of a currently visible map.

- **Minor Zoom In**

  Selected if (1) the user's task has not changed and (2) the user requests the system to zoom in on a certain area.

- **Major Zoom In**

  Selected if (1) the user's task has changed and (2) the user requests the system to zoom in on a certain area.

- **Pan**

  Selected if (1) the user's task has changed and (2) objects with geographic locations are to be presented or a geographic region is to be presented and (3) the objects/region to be presented are/is not located within the boundary of a currently visible map.

If CUBRICON is to perform an expand or zoom out, then CUBRICON must decide which of the existing maps to expand and/or modify to accommodate the new objects and the area containing the new objects. The criteria used to determine which of the existing maps to transform are listed below in ranked order.

1. Expand the window containing the greatest number of objects requested by the user.
2. Expand the map which is closest to the smallest rectangle enclosing the objects requested by the user.
3. Expand the map with the greatest geographic area.

10.1.1.2 Determining the Relevant Objects Relevancy is also important in selecting the objects to display in a map region. Frequently, sophisticated application systems include one or more massive databases and, indeed, the databases may be shared by more than one application system. When a system such as CUBRICON selects objects from the database for display on a map, it should be discriminating in its selection. Not all the available objects should be selected from the database for display, since this could result in an unnecessarily cluttered and confusing map. Instead, only the relevant objects should be displayed. Relevant objects are objects which are (1) specifically requested by the user, (2) relevant to the dialogue and support dialogue continuity, and/or (3) relevant to the user's task.

Continuity and relevance are key factors in discourse. Without these factors, people find discourse disconcerting and unnatural. The attentional discourse space representation [Grosz78; Grosz86; Sidner83; Grosz85] is the key knowledge structure used in determining which objects to display in order to maintain dialogue continuity and relevance. The representation of the discourse focus space is in two structures (1) a main focus list and (2) a display model (discussed in Sections 4.2.1 and 4.2.2).

The technique used in the CUBRICON system to determine the objects relevant to the user's task relies on the use of the *entity rating system* of the user model (discussed in Section 4.3). When composing maps, CUBRICON displays only those objects above the critical importance threshold for the user's current task. Thus, for an Offensive Counter Air (OCA) planning task, CUBRICON would display all airbases, SAM sites, critical factories and plants, but not objects such as schools or minor industry.

10.1.1.3 Determining the Region to Display After determining the appropriate map transformation and objects to display in a map region, based on relevancy, CUBRICON must delimit the boundary of the region to be displayed. The coordinates of the boundary are determined by the smallest rectangle enclosing both the objects to be displayed and the existing map to be expanded, if expansion of an existing map is relevant. This boundary is then enlarged to include a small border area.

10.1.1.4 Scaling the Region One of the human factors guidelines incorporated into CUBRICON is to maintain consistency across displays [Smith86]. An object presented on more than one display should have the same shape. To accomplish this a geographic map must display one vertical kilometer as the same distance as one horizontal kilometer. Therefore, the the boundary of the region is extended vertically or horizontally (if necessary), so that when the region is displayed in the window provided by the Intelligent Window Management system, a vertical kilometer and a horizontal kilometer are the same distance.

10.1.1.5 Displaying the Map Having determined the type of map transformation, the objects to display on the map, the appropriate map boundary and properly scaled the map, the map is displayed in the appropriate window with the appropriate icons, colors, labels, etc. The composition and presentation of the map in the appropriate window is performed by the Color Graphics system.

10.1.2 Map Operations

The operations on geographic maps are listed below. In general, CUBRICON's decision-making process has been designed with the goal of maintaining context for the user and helping the user understand the transition from one map to another. For each of the map transformations, CUBRICON presents the new map in the context of the previously displayed map. In communicating the map transitions, CUBRICON uses a "region boundary box" to outline or highlight a region that is a sub-region of another. Objects to be displayed on each new map are selected according to their importance to the user's task, as discussed above.

- **Map Creation**

  A new map is created and displayed in a window on the color-graphics screen.

- **Zoom Out**

  The area shown in a map window is extended to include appropriate additional area of interest to the user. The criteria used to determine which map to extended was described in the previous section. A "region boundary box" is superimposed on the new map to show the boundary of the map that was previously displayed. This helps the user unde.stand the transformation from previous map to new map display. Figure 10-1 shows the CUBRICON color-graphics screen after a zoom out operation has been performed.

- **Major Zoom In**

  A sub-region (specified by either the user or the system) of the current main map is enlarged. The map transition is performed as follows: CUBRICON first moves

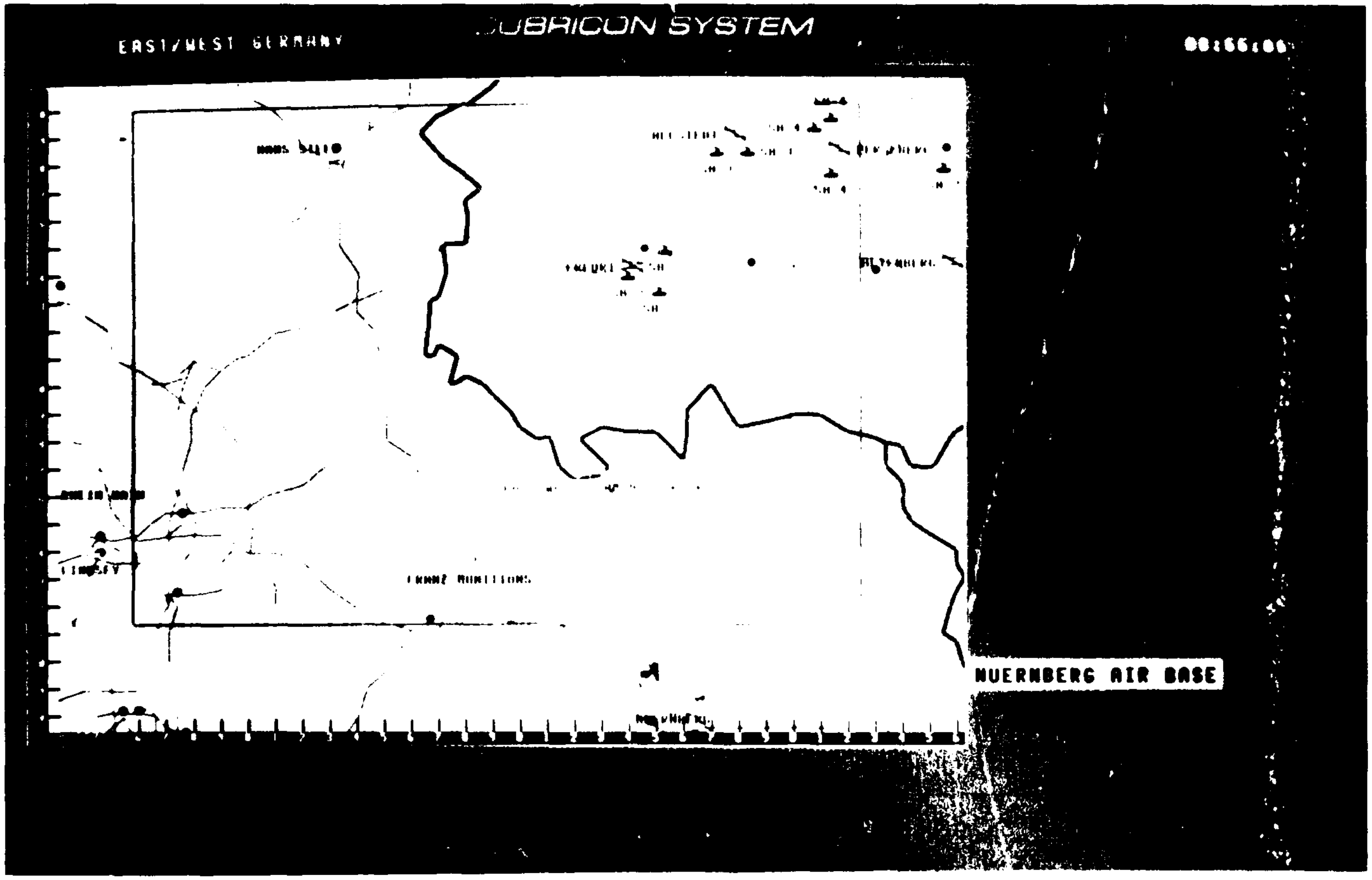


Figure 10-1: Color Graphics Screen After Zoom Out Operation

the map currently displayed in the main window to a secondary window and adds a "region boundary box" to this secondary window showing the sub-region that is to be enlarged. The enlarged version of the sub-region is then displayed in the main window. Figure 10-2 shows the CUBRICON color-graphics screen after a major zoom in operation has been performed.

- **Minor Zoom In**

  A sub-region (specified by either the user or the system) of the current main map is enlarged. The map transition is performed as follows: CUBRICON superimposes a "region boundary box" on the main map showing the outline of the region to be enlarged. An enlarged version of the designated region is then displayed in a secondary window of appropriate size. Figure 10-3 shows the CUBRICON color-graphics screen after a minor zoom in operation has been performed.

- **Pan**

  Pan to a new region. The map transition is performed as follows: in one of the secondary windows, CUBRICON displays a map region whose boundary is the smallest rectangle enclosing the old map in the main window and the new region to be displayed; CUBRICON then shows region boundary boxes designating (1) the old region that was in the main window and (2) the new region to be displayed. The new region is displayed in the main map window. Figure 10-4 shows the CUBRICON color-graphics screen after a pan operation has been performed.

10.2 Part-Whole Decomposition Maps

The map modality is the preferred modality used to represent the parts of an object, such as an airbase. An airbase's parts might be objects such as runways, radars and SAMs. The type of map generated is a part-whole decomposition map (see Section 13.1) which is a schematic diagram depicting the parts of the object and their relative locations.

In selecting the objects to display on a part-whole decomposition map, all parts of an object are relevant in the current CUBRICON system. When representing parts of an object the existing knowledge base contains only those objects which highly relevant to the OCA planning tasks defined. Therefore, the determination of the type of map transformation and objects to display in the map do not apply to part-whole decomposition maps. As one or more large databases are accessed by CUBRICON there will be a need to discriminate among the parts, displaying the objects which are most relevant. The decision-making logic used to select the relevant information to display on geographic maps, described above, will be used to select relevant information to display on part-whole decomposition maps.

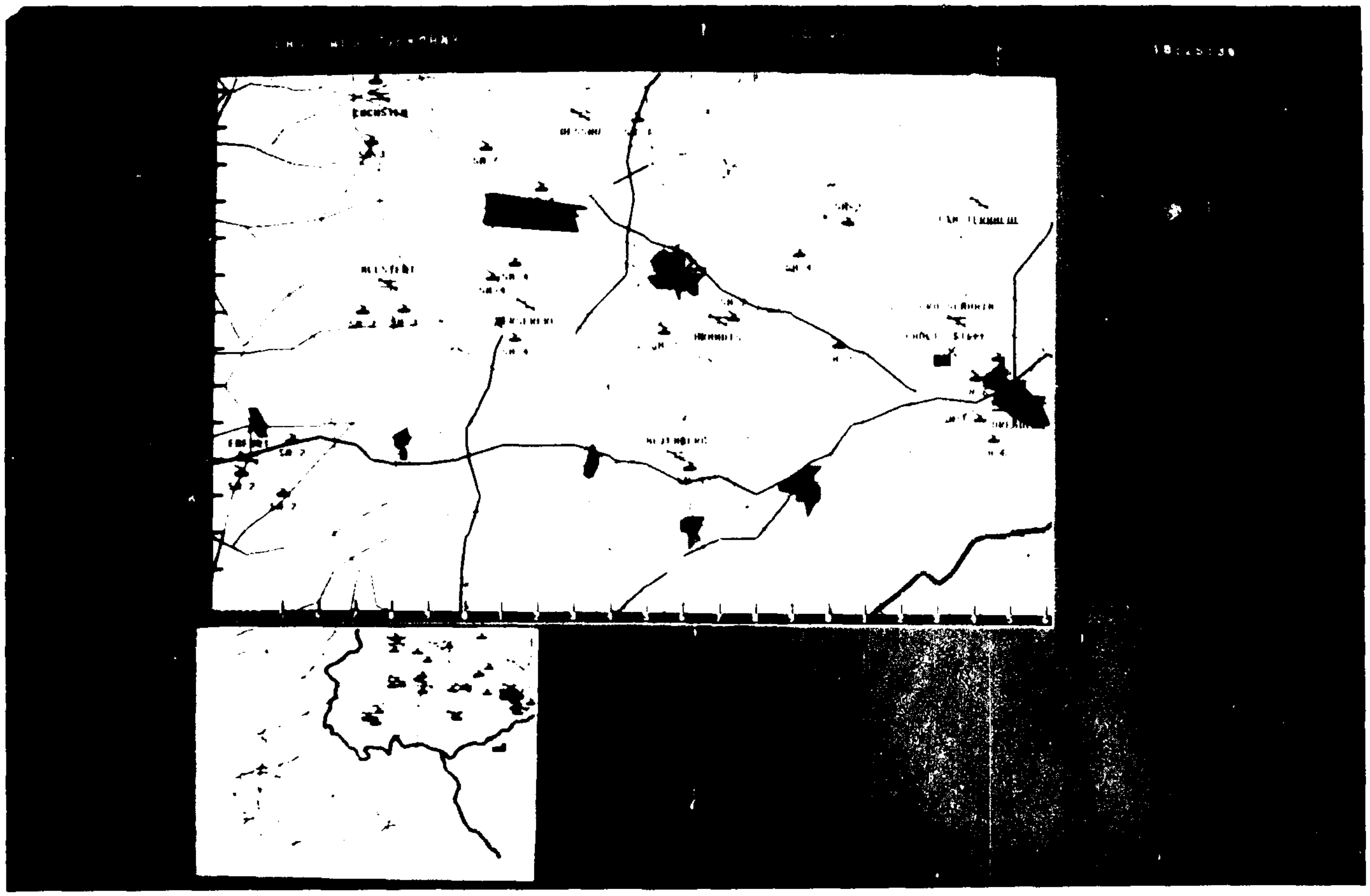


Figure 10.2: Color Graphics Screen After Major Zoom In Operation

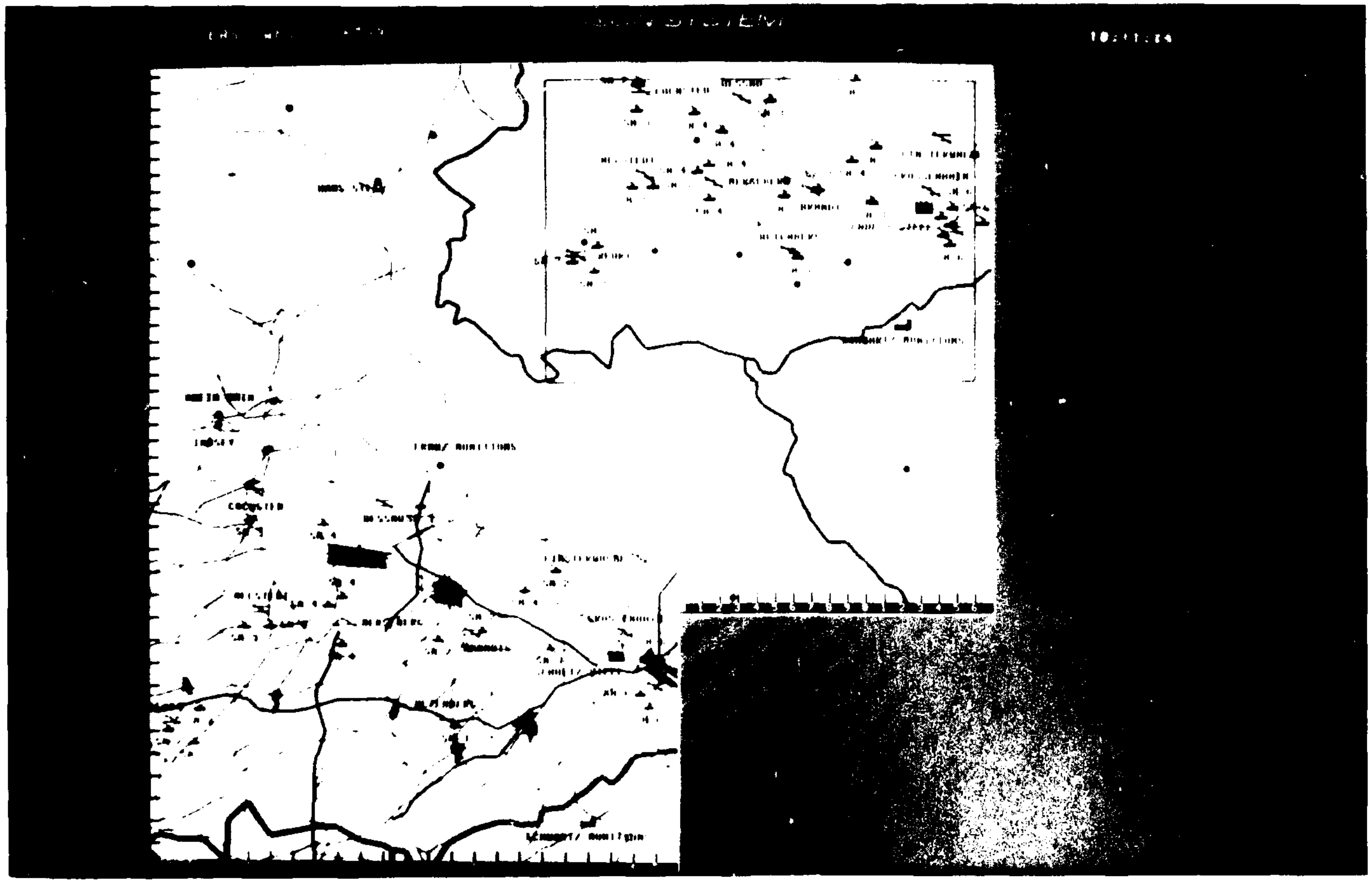


Figure 10-3: Color-Graphics Screen After Minor Zoom In Operation

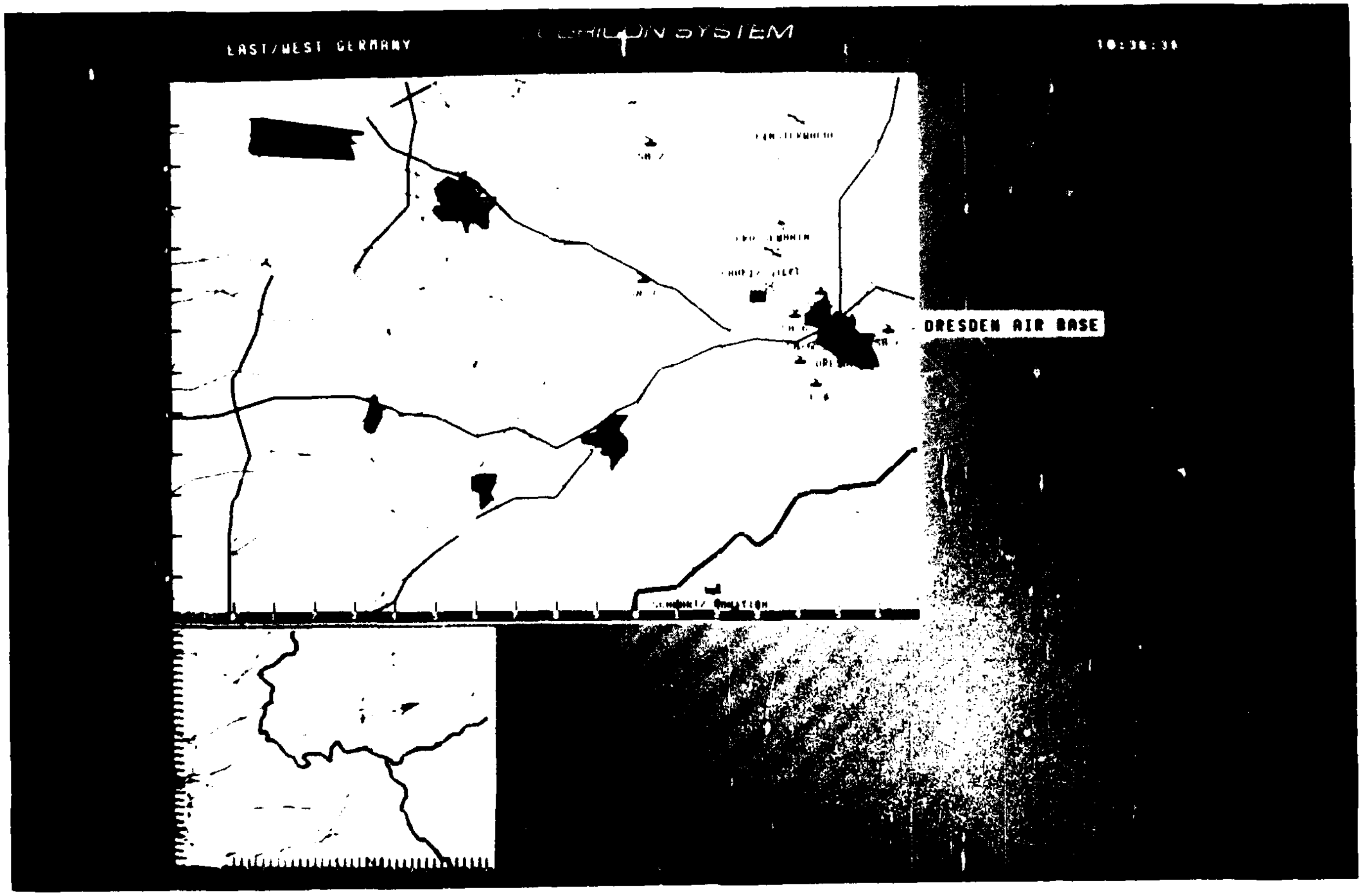


Figure 10-4: Color-Graphics Screen After Pan Operation

11 Table Modality

The table modality is selected by CUBRICON for information presentation when the information consists of numerous tuples that represent common relations between various objects or concepts. Most frequently CUBRICON uses the table modality when the values of common attribute(s) of several entities must be expressed. The table modality actually includes two modalities: (1) tables displayed on the monochrome screen and (2) tables displayed on the color-graphics screen. A detailed explanation of the criteria for selecting the table modalities is in Section 7.1.3, Table Modality Selection Criteria. A detailed explanation of the table window placement algorithm that covers placement on both the monochrome and color displays is in Section 13.3.

11.1 Determining the Relevant Objects and Properties

Relevancy and consistency are critical factors for an HCI to consider when determining what information to present and how to present the information to the user. In CUBRICON's table generation process, both relevancy and consistency play important roles in determining which objects and properties to display on a table.

Frequently, sophisticated application systems include one or more massive databases and, indeed, the databases may be shared by more than one application system. When a system such as CUBRICON selects objects from the database for display on a table, it should be discriminating in its selection. Not all the available objects should be selected from the database for display, since this could result in an unnecessarily large table. Therefore, only the relevant objects and properties should be displayed. Relevant objects and properties are those which are (1) specifically requested by the user, (2) relevant to the dialogue and support dialogue continuity, (3) relevant to the user's task and (4) maintain consistency between related displays. In order to make intelligent decisions with respect to these issues, CUBRICON uses its knowledge sources (see Section 4). In order to judge relevancy with respect to the dialogue, CUBRICON uses its discouse model, discussed in Section 4.2.2. In order to judge relevancy with respect to the user's task, CUBRICON uses its user model, discussed in Section 4.3. For maintaining consistency, CUBRICON primarily uses the discourse model.

The technique used in the CUBRICON system to determine the objects and properties relevant to the user's task relies on the use of the *entity rating system* of the user model (discussed in Section 4.3). If the system chooses to augment the requested information with additional relevent information, it bases its relevancy decisions on the knowledge it has represented in the entity rating list. For example, if the user asks *"What ac-pools are at the Nuernberg Airbase?"*, the system does not just list the names of the ac-pools, but additional attributes of these entities which it deems relevant.

One of the situations in which CUBRICON selects the table modality for information presentation is when additional information is to be presented to augment a map display. Frequently, when a map modality has been selected as the primary modality for information presentation, the map modality alone is not sufficient to present all the relevant information, since there are often many important attributes associated with objects and their presentation on the map would severely clutter the map. Therefore, a table is generated which lists all of the objects that are visible on the map along with their relevant properties. It is important to maintain consistency between these displays, so every object on the map is represented on the table. Associated with each map window is a list of its contents as part of the display model (which is part of the discourse model). CUBRICON uses this list in determining the objects to include on the table. When determining the relevant properties to display in the table, CUBRICON consults the entity rating list (see Section 4.3) of its user model. The entity rating list is a task dependent rating of all the entity types in CUBRICON's knowledge base, weighted so as to indicate their importance to the user's task. Also included in the entity rating list are the relevant properties for each entity type, listed in order of their importance to the particular task. CUBRICON uses the relevant properties listed for each entity type in the user model when creating a table. For an OCA mission planning task, for example, the properties that are be displayed for airbases are: name, location, and disposition (friendly or enemy), but the property nationality (which is defined in the knowledge base) is not.

As stated previously, maintaining consistency across displays is an important principle in the *CUBRICON* design. *In order to maintain consistency* between a map display and its related table, whenever the map is modified (e.g., by the addition of new objects), then the related table is similarly modified.

11.2 Composing a Table

In composing a table, the rows and columns are ordered by their relevance to the user's task. Rows are displayed with the most important entity types listed first. If multiple objects of the same type are displayed, then these objects are listed alphabetically by name. Columns are displayed with the most important properties listed left to right. The relative importance of the different entity types and the relative importance of the properties per type are defined in the user model.

CUBRICON generally highlights appropriate rows of a table when certain entities are especially significant. This would be the case, for example, if the information to be presented is a subset of an existing table. If such a table is so large that it is not entirely visible in the window that it occupies and not all the highlighted entities or rows are visible, then CUBRICON scrolls the table so that all the highlighted rows are visible or, if this is impossible, then it scrolls the table so that the first highlighted row is at the top of the window.

11.3 Examples

Map-Related Table Example

In the first example, a table is generated which is related to a geographic map. The entities and properties selected for presentation in the table are chosen to comply with the user request, to present information that is relevant to the user's task, and to maintain consistency between the table and its related map.

USER Spoken or Typed Input: "Display the Fulda Gap region."

CUBRICON:

Decision Processing:

Since, according to CUBRICON's knowledge base, a region can be presented graphically, the map modality is selected for the presentation of the Fulda Gap region. Section 10 discusses the CUBRICON methodology for creating and composing a map display. The map modality alone is not sufficient, however, since there are important attributes associated with the objects, which would severely clutter the map if presented there. Since the number of objects on the map is voluminous (more than four) and there is at least one attribute common to all of the objects, the table modality is selected to display the attributes of the objects on the map. When creating the table related to the map display, the content list (part of the display model) of the map window is used to determine which objects to include in the table. For each type of object, the relevant properties to include in the table are retrieved from the user model. The larger table of Figure 11-1 is an example of a table on the monochrome screen that was created to augment a map display that is on the color-graphics screen.

The Table as Primary Modality for Information Presentation

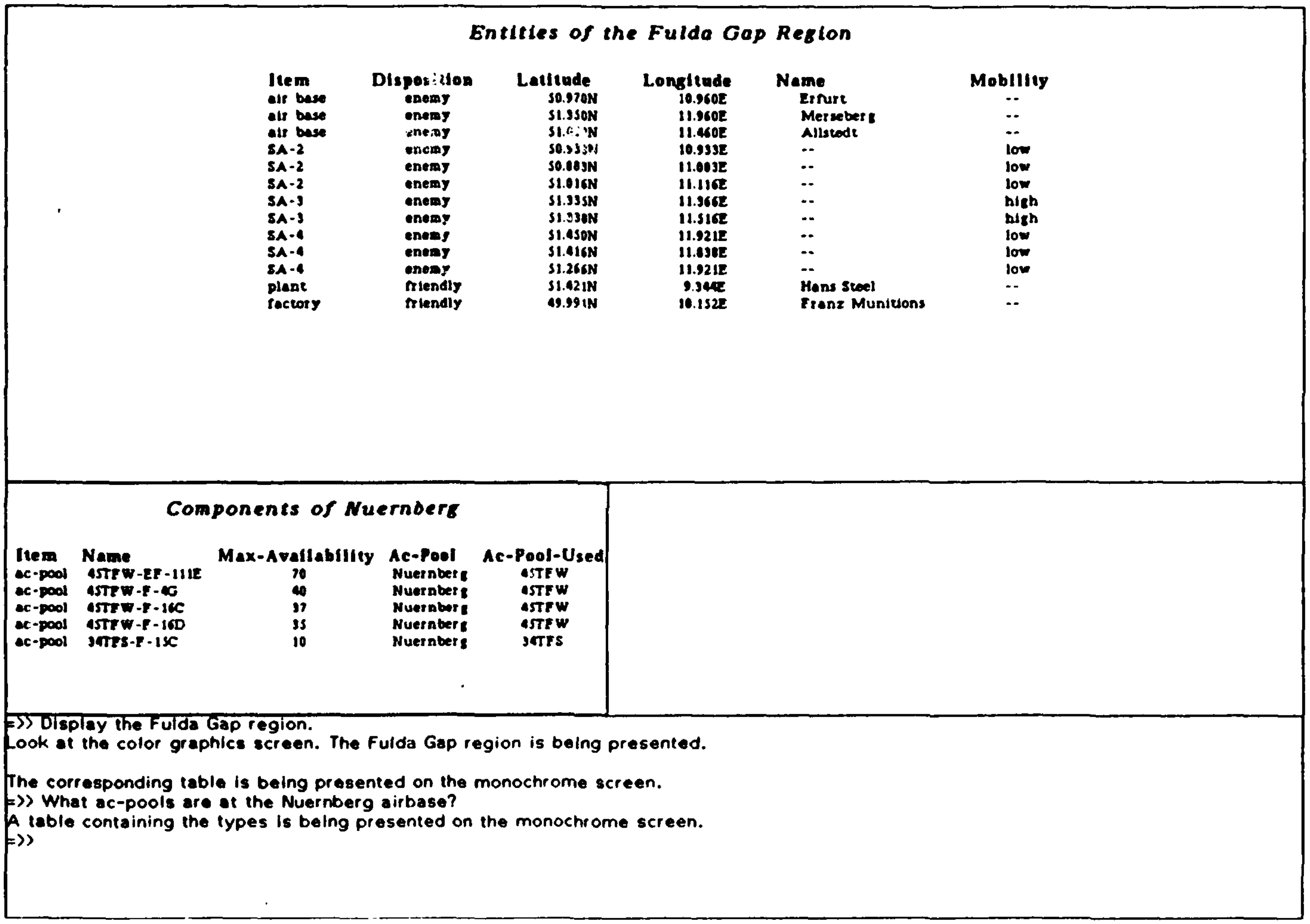
In the second example, the information to be presented in response to the user's request consists of a voluminous amount of information that cannot be presented graphically and the information items include common attributes. Therefore, the table modality is selected as the primary modality for presentation of the information.

USER Spoken or Typed Input: "What ac-pools are at the Nuernberg airbase?"

CUBRICON:

Decision Processing:

Since the information to be presented as a result of this request cannot be presented graphically, the map modality is rejected. This information is voluminous (more than 4



Entities of the Fulda Gap Region

Item	Disposition	Latitude	Longitude	Name	Mobility
air base	enemy	50.970N	10.960E	Erfurt	--
air base	enemy	51.350N	11.960E	Merseberg	--
air base	enemy	51.[illegible]N	11.460E	Allstedt	--
SA-2	enemy	50.533N	10.933E	--	low
SA-2	enemy	50.883N	11.003E	--	low
SA-2	enemy	51.016N	11.116E	--	low
SA-3	enemy	51.335N	11.366E	--	high
SA-3	enemy	51.330N	11.516E	--	high
SA-4	enemy	51.450N	11.921E	--	low
SA-4	enemy	51.416N	11.830E	--	low
SA-4	enemy	51.266N	11.921E	--	low
plant	friendly	51.421N	9.344E	Hans Steel	--
factory	friendly	49.991N	10.152E	Franz Munitions	--

Components of Nuernberg

Item	Name	Max-Availability	Ac-Pool	Ac-Pool-Used
ac-pool	45TFW-EF-111E	70	Nuernberg	45TFW
ac-pool	45TFW-F-4G	40	Nuernberg	45TFW
ac-pool	45TFW-F-16C	37	Nuernberg	45TFW
ac-pool	45TFW-F-16D	35	Nuernberg	45TFW
ac-pool	34TFS-F-15C	10	Nuernberg	34TFS

```
=>> Display the Fulda Gap region.
Look at the color graphics screen. The Fulda Gap region is being presented.

The corresponding table is being presented on the monochrome screen.
=>> What ac-pools are at the Nuernberg airbase?
A table containing the types is being presented on the monochrome screen.
=>>
```

Figure 11-1: Presentation of a Map-Related Table and a Primary Table

objects) and there is a least one property common to all the objects, therefore the table modality is selected. The relevant properties to include for each object are determined by the user model. Figure 11-1 shows the table generated as a result of this user request.

Temporary Table Presentation

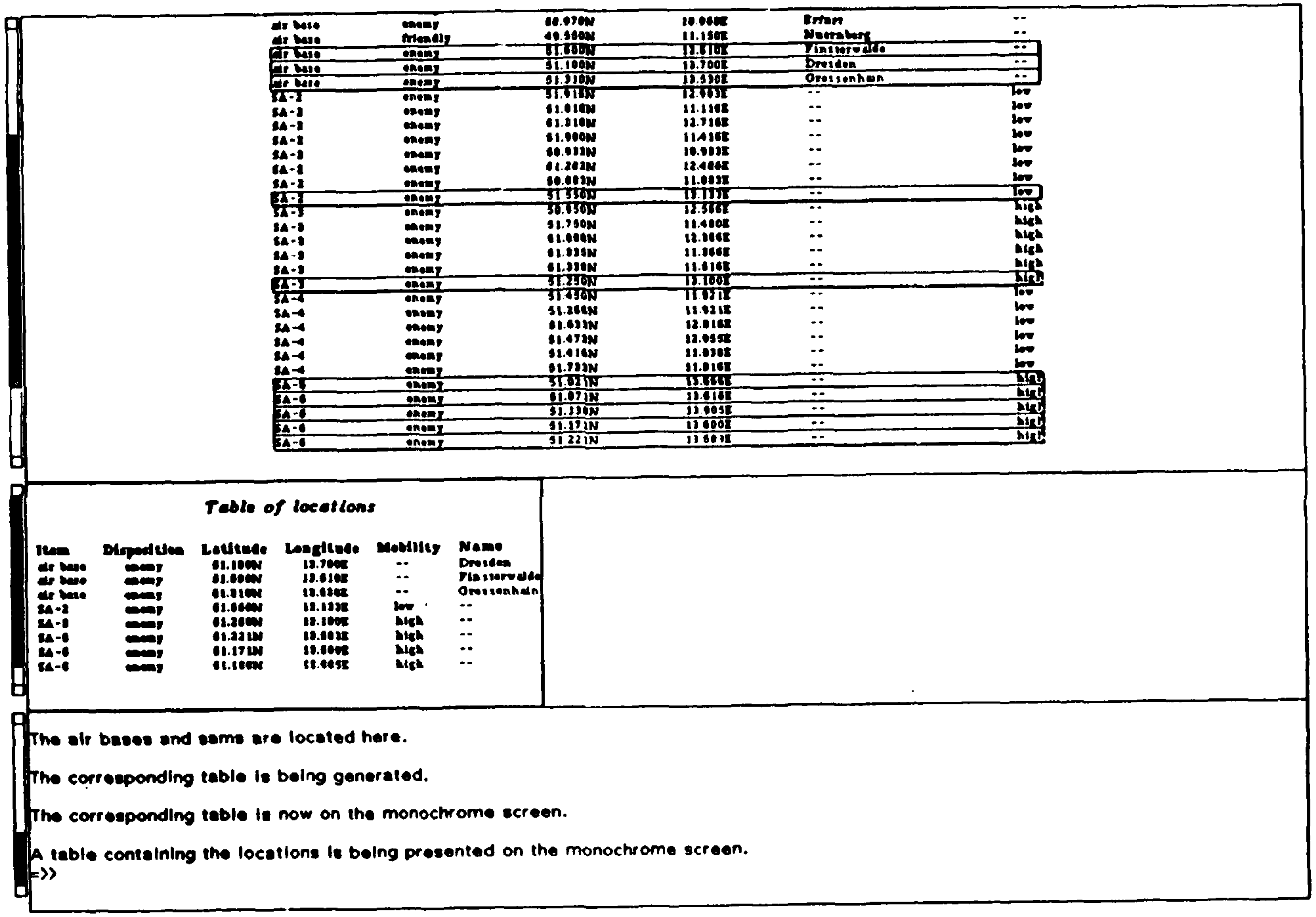
In the third example, the user requests information about the threats around the Dresden Airbase. These threats are presented to the user using the table modality since the information consists of a voluminous number of objects with common attribute types. The decision making process for deciding to present a temporary table (a table that is displayed for one dialogue cycle) is explained below.

USER Spoken or Typed Input: "What are the threats around the Dresden Airbase?"

CUBRICON:

Decision Processing:

Since the requested information is already visible on an existing table which is related to a map display, CUBRICON highlights the information on the existing table. However, the table is large and numerous table entries (more than four) must be highlighted. Also, since the relevant rows are not contiguous, the user may find it difficult to compare information, particularly if the information is not visible on one screen. *Therefore,* a temporary table is created which contains the information requested by the user, making it easy to view the requested information contiguously. This table is called a temporary table in the CUBRICON system since it is only displayed until output is generated in response to the next user request. The task-relevant properties to display for each object type are listed in the user model. CUBRICON uses these properties in composing the table. Figure 11-2 shows the monochrome screen after the table has been presented.



air base	enemy	50.970N	10.960E	Erfurt	--
air base	friendly	49.500N	11.150E	Nuernberg	--
air base	enemy	51.600N	13.610E	Finsterwalde	--
air base	enemy	51.100N	13.700E	Dresden	--
air base	enemy	51.310N	13.530E	Grossenhain	--
SA-2	enemy	51.916N	12.903E	--	low
SA-2	enemy	51.016N	11.116E	--	low
SA-2	enemy	51.816N	12.716E	--	low
SA-2	enemy	51.900N	11.416E	--	low
SA-2	enemy	50.933N	10.933E	--	low
SA-2	enemy	51.283N	12.486E	--	low
SA-2	enemy	50.883N	11.083E	--	low
SA-2	enemy	51.550N	13.133E	--	low
SA-3	enemy	50.850N	12.566E	--	high
SA-3	enemy	51.750N	11.400E	--	high
SA-3	enemy	51.000N	12.366E	--	high
SA-3	enemy	51.335N	11.866E	--	high
SA-3	enemy	51.330N	11.616E	--	high
SA-3	enemy	51.250N	13.100E	--	high
SA-4	enemy	51.450N	11.921E	--	low
SA-4	enemy	51.366N	11.921E	--	low
SA-4	enemy	51.633N	12.016E	--	low
SA-4	enemy	51.473N	12.055E	--	low
SA-4	enemy	51.416N	11.038E	--	low
SA-4	enemy	51.733N	11.016E	--	low
SA-5	enemy	51.621N	13.666E	--	high
SA-6	enemy	51.071N	13.616E	--	high
SA-6	enemy	51.130N	13.905E	--	high
SA-6	enemy	51.171N	13.600E	--	high
SA-6	enemy	51.221N	13.603E	--	high

Table of locations

Item	Disposition	Latitude	Longitude	Mobility	Name
air base	enemy	51.100N	13.700E	--	Dresden
air base	enemy	51.600N	13.610E	--	Finsterwalde
air base	enemy	51.310N	13.530E	--	Grossenhain
SA-2	enemy	51.550N	13.133E	low	--
SA-3	enemy	51.250N	13.100E	high	--
SA-6	enemy	51.221N	13.603E	high	--
SA-6	enemy	51.171N	13.600E	high	--
SA-6	enemy	51.130N	13.905E	high	--

The air bases and sams are located here.

The corresponding table is being generated.

The corresponding table is now on the monochrome screen.

A table containing the locations is being presented on the monochrome screen.
=>>

Figure 11-2: Temporary Table Display

12 Form Modality

People in all walks of life are accustomed to filling out forms of various types such as Federal and State Income Tax forms, motor vehicle registration forms, loan applications, and insurance applications. By a form we mean a visually organized format that provides a person with blanks or slots in which data is to be entered and each blank or slot is accompanied by a specification of the type of data to be entered in the slot. Typically forms provide an efficient means of data collection since the form provides the user with an overview of what is expected (although the specific details for particular data items may be lacking) and with a logically organized sequence of data requests.

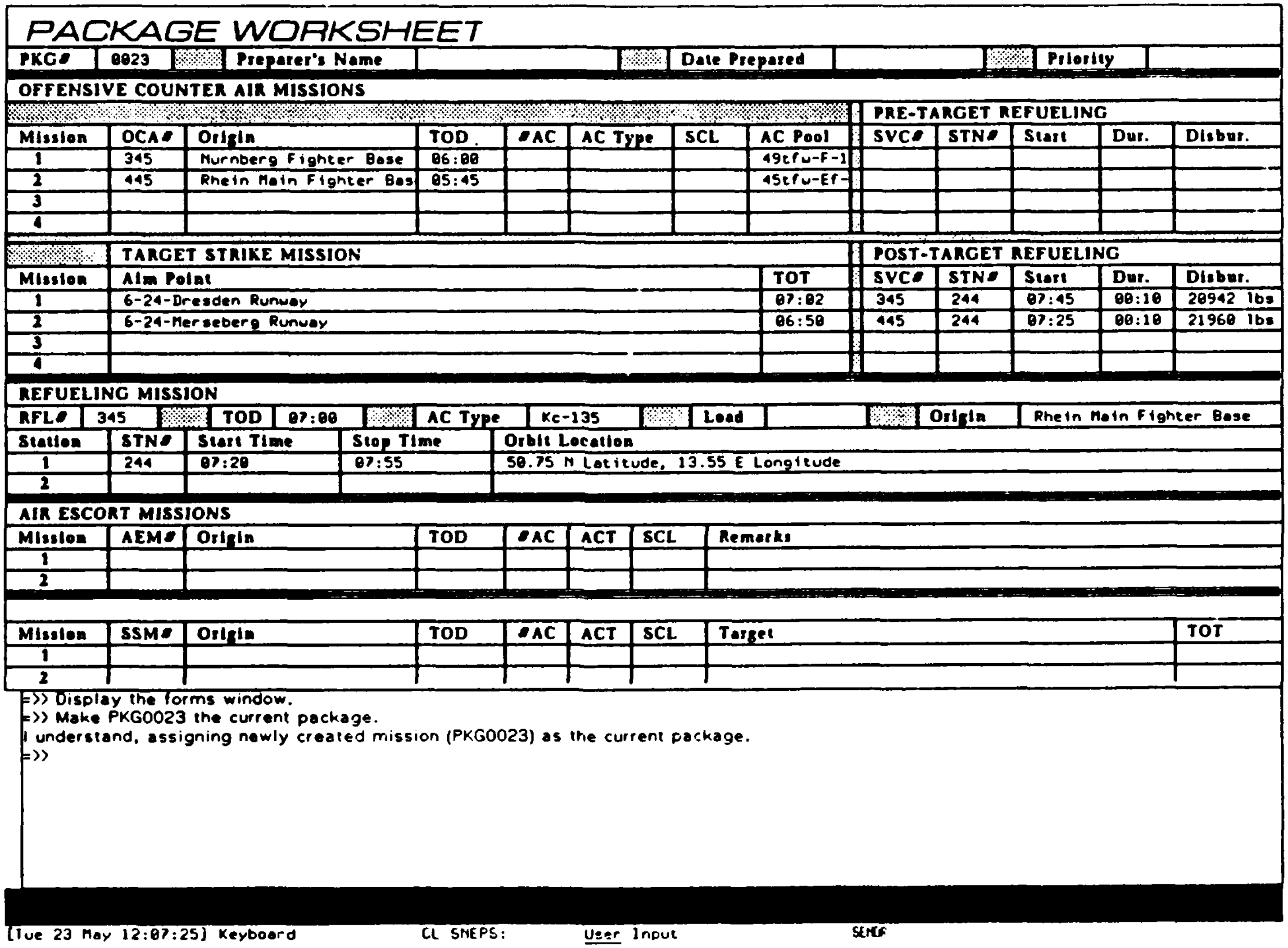
For the application domain of this project, a Package Worksheet form was developed for the CUBRICON prototype implementation. This form is meant to be representative of forms that mission planners use, although not faithful in detail to any particular form. The form developed for the CUBRICON project provided sufficient coverage of mission planning data to be realistically challenging without expending a disproportionate amount of project resources on the form modality alone. The Package Worksheet is shown in Figure 12-1.

The CUBRICON Package Worksheet form is one means of communicating information between user and computer. It is one modality in the CUBRICON suite of modalities, to be used individually or in combination with other means of communication (e.g., speech input accompanied by pointing to other displays). The form is used to create or modify OCA mission plans. The user can enter data into the form using a variety of natural language input types such as:

- *"Enter the Nuernberg Airbase as the origin airbase for OCA123."*
- *"Enter the value 06:30 here <point-to-time-slot>."*
- *"Enter this <point-map-icon> here <point-form-slot>."*

These natural language inputs can be typed or spoken. Although not yet implement, the CUBRICON design provides the user with the ability to type or speak directly into any slot of the form.

Completing a form via speech input can be very efficient for the user. The combination of terse spoken inputs and pointing gestures can make form filling very fast and easy. Although not yet implemented, the CUBRICON input understanding capability can be extended to handle terse inputs such as a point gesture to select a particular form slot accompanied by the spoken entry such as "Nuernberg Airbase". Another desirable extension to the CUBRICON system is the capability of using terse "yank" and "enter" commands. If this were available, a user could point to an object (e.g., the Nuernberg Airbase icon) on a map display while saying "yank" in order to tell the system to retrieve a copy of the object and then tell the



PACKAGE WORKSHEET

PKG#	0023	Preparer's Name		Date Prepared		Priority	

OFFENSIVE COUNTER AIR MISSIONS

								PRE-TARGET REFUELING				
Mission	OCA#	Origin	TOD	#AC	AC Type	SCL	AC Pool	SVC#	STN#	Start	Dur.	Disbur.
1	345	Nurnberg Fighter Base	06:00				49tfw-F-1					
2	445	Rhein Main Fighter Bas	05:45				45tfw-Ef-					
3												
4												

	TARGET STRIKE MISSION		POST-TARGET REFUELING				
Mission	Aim Point	TOT	SVC#	STN#	Start	Dur.	Disbur.
1	6-24-Dresden Runway	07:02	345	244	07:45	00:10	20942 lbs
2	6-24-Merseberg Runway	06:50	445	244	07:25	00:10	21960 lbs
3							
4							

REFUELING MISSION

RFL#	345	TOD	07:00	AC Type	Kc-135	Load		Origin	Rhein Main Fighter Base

Station	STN#	Start Time	Stop Time	Orbit Location
1	244	07:20	07:55	50.75 N Latitude, 13.55 E Longitude
2				

AIR ESCORT MISSIONS

Mission	AEM#	Origin	TOD	#AC	ACT	SCL	Remarks
1							
2							

Mission	SSM#	Origin	TOD	#AC	ACT	SCL	Target	TOT
1								
2								

```
=>> Display the forms window.
=>> Make PKG0023 the current package.
I understand, assigning newly created mission (PKG0023) as the current package.
=>>
```

```
[Tue 23 May 12:07:25] Keyboard          CL SNEPS:          User Input          SEMF
```

Figure 12-1: The CUBRICON Package Worksheet Form

system to enter the object in a particular form slot by simple saying "enter" accompanied by a point gestures to select the slot. Another advantage to spoken input is that users can avoid typographical errors. However, users may encounter problems with the speech recognition system if they are not accustomed to its limitations.

As discussed in previous sections, the CUBRICON interface system provides a unified system in which various displays and presentations reflect a single integrated underlying reality. The different modalities, including the Form, provide different ways of communicating between human and computer about the same conceptual objects, namely mission plans, targets, airbases, etc. Thus the Form is a visualization of information that is stored in the knowledge base. User entries into the Form create or modify knowledge base structures. These knowledge base structures are, of course, semantic network representations of mission planning knowledge and instantiations of mission plans. Section 4.1 discusses the knowledge base representations in some detail.

More specifically, underlying the Form is a Form Model (see Section 4.2.3) which consists of three data structures:

- The Mission Template and associated KB mission instantiations: the mission planning knowledge stored in the KB along with KB representations of individual mission plans that have been developed by the user.
- The Form Window Object: the Lisp window object whose visualization is the Package Worksheet display and which contains information about how to display the form on the CRT.
- The Information Data Structure (IDS): the data structure which keeps track of the information entered or displayed in the Form and which serves as a short term working memory for the mission plans being developed by the user. The IDS is part of the Form Window Object and the mission information stored in the IDS is in the form of KB objects.

13 Intelligent Window Manager

The Intelligent Window Manager (IWM) automatically performs all window placement and manipulation functions within the CUBRICON system. The decision to automate window management functions was based on the premise that this would reduce the efforts required of the user for window management, and thus free the user's mental and temporal resources for task domain activities. The goal was to automatically perform window management functions well enough so that the user would not need to manipulate the windows directly. The window management functions performed by the CUBRICON window manager include window: creation, placement, sizing, movement, and removal.

The fact that the time spent manipulating windows in a windowing system consumes a significant portion of overall problem solving time has been demonstrated experimentally [Davies85; Bly86], at least for certain types of tasks. Davies et al. found that for tasks requiring supplemental information relative to a primary task, the windowing environment allowed more error-free performance but took significantly longer. Their study indicates that the additional time spent, was due to window management operations (e.g., displaying and positioning windows, scrolling to desired locations within windows). Their data also indicates that the reduction in errors was not simply the result of having spent more time on the task. The time differential was evident even when all errors had to be corrected. Apparently, the overhead of window management adds a significant time burden.

Bly and Rosenberg [Bly86] studied the relative tradeoffs between *overlapping and tiled windowing systems* (see Section 13.2). They found that overlapping systems are good at optimally sizing windows (to contain the desired information), but are more difficult to manage. The CUBRICON window management methodology is a hybrid of tiled and overlapping approaches. The default configuration is tiled, but windows can overlap when necessary to avoid overly cluttered windows. Four pre-defined tiled window positions are available on each display. These overlap adjacent windows when necessary. If more than four windows are requested, the least important window is iconized and removed (i.e., removed and displayed displayed as an icon). Although redisplay of iconized windows is not implemented at this time, this feature will eventually allow the recall of windows that were recently removed.

13.1 Window Types

The types of windows managed by the Intelligent Window Management system are:

- map,
- table,
- mission planning form,

- text, and
- dynamic text windows.

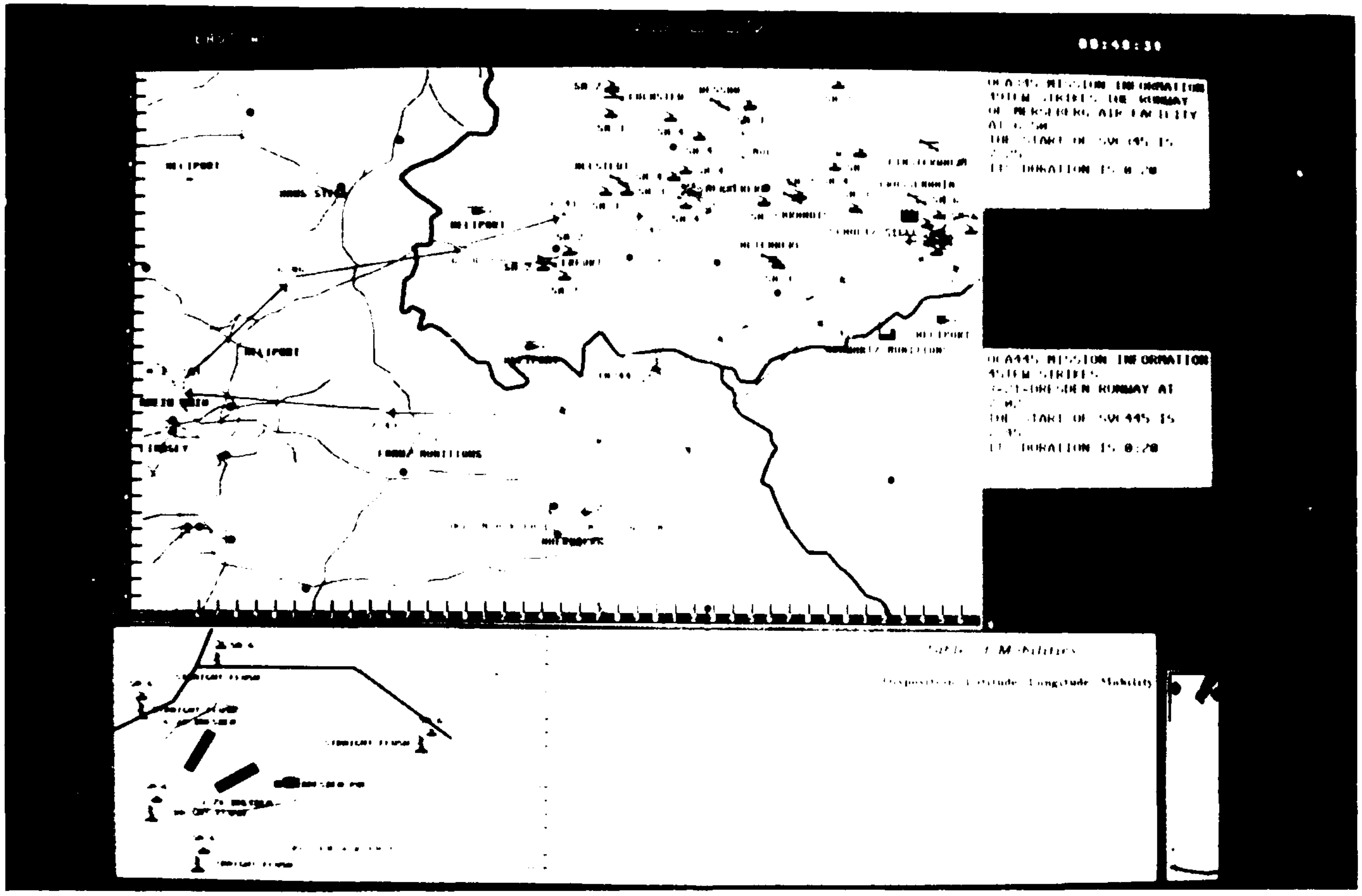
Figure 13-1 shows a color graphics display containing map, table, text, and dynamic text window types.

Map windows are used by CUBRICON to display geographic information. All of the window manager functions (i.e., creation, placement, sizing, removal, and movement) can be performed on map windows. Map windows are placed only on the color graphics display. There are two types of map windows:

- geographic map windows and
- part-whole decomposition map windows.

Geographic map windows are composed of background maps overlaid with relevant application domain information. The background maps include objects such as national borders, roads, rivers, and cities which are displayed using the MAP Display System [Hilton87]. *Part-whole decomposition map windows* are used to display the decomposition of an object into its component parts. For example, CUBRICON's knowledge base contains information about the composition of certain airbases in terms of their parts such as radars, SAM systems, runways, fuel storage facilities, etc. Using this knowledge, CUBRICON can display an airbase in terms of its parts in a part-whole decomposition map window. The part-whole decomposition map displaying the Dresden airbase is shown in Figure 13-1. This map contains the boundary of the airbase and an iconic representation of each object which is a part of the airbase. The part-whole decomposition map does not utilize a background map.

Table windows are used to display numerous tuples that represent common relations between various objects or concepts. The operations performed on table windows are creation, placement, sizing, and removal. Tables can be *temporary* or *permanent.* Generally, if the information requested by the user is already contained in an existing table, the relevant table entries are highlighted. However, if the relevant table entries are located at various positions in the table that are widely distributed throughout the table, then the user may find it difficult to assimilate the information, particularly if the table entries are not all simultaneously visible on the window (i.e., scrolling is required to view them). Therefore, in this case, CUBRICON creates a temporary table containing only the requested information, enabling the user to view the information in contiguous rows of a smaller table. This type of table is referred to as a temporary table since it is displayed for one dialogue cycle. CUBRICON leaves permanent windows on the display until there is contention for the display space, when the system decides to display another window on the same CRT screen. The manner in which CUBRICON handles this issue is discussed in Section 13.3. In general, permanent table windows can be placed either on the monochrome display or the color graphics display, whereas temporary table windows are placed only on the monochrome display. In addition,



OCA395 MISSION INFORMATION
395TH STRIKES THE RUNWAY
OF MERSEBERG AIR FACILITY
AT [illegible]
THE START OF SVC495 IS
[illegible]
ITS DURATION IS 0:20

OCA445 MISSION INFORMATION
451FM STRIKES
[illegible]-DRESDEN RUNWAY AT
[illegible]
THE START OF SVC445 IS
[illegible]
ITS DURATION IS 0:20

Table of Mobilities

Disposition	Latitude	Longitude	Mobility

Figure 13-1: Types of Windows

permanent tables can be related to another display, such as a map. Tables are frequently used by the system in combination with maps to display information that is relevant to the user. Such a map-related table augments the map display by showing a additional important attributes of the displayed objects. The CUBRICON window placement and importance algorithms for related and unrelated table windows are discussed in Section 13.3.

Text windows are used to display information pertaining to one or more icons on a map. Currently, the information displayed in this type of window is text identifying the name or type of the relevant icon(s). If the system is unable to place the text window next to the relevant icon(s) (due to clutter on the map), then a pointing arrow is used to show the correlation between the appropriate icon(s) and the text window. Figure 13-1 shows a text window identifying the Nuernberg airbase as the origin of an OCA mission.

Dynamic text windows are used to display natural language text on the color graphics display. Such a window is dynamic in that the window is sized to fit the text and if text is added to the window, then the size is enlarged. Figure 13-1 shows two dynamic text windows containing natural language text describing the highlights of an OCA mission plan presentation.

13.2 Window Layout

The default window configuration of each CUBRICON screen consists of four pre-defined window positions. Figure 13-2 shows the pre-defined window positions. Positions 1-4 are used for the tiled windows. Positions 5-9 are used for iconized windows. An iconized window is a window that has been removed from the main part of display and redisplayed (symbolically) in the form of a small icon in the lower right hand side of the CRT screen.

The configuration of the windows is a hybrid window layout combining tiled and overlapping approaches. Table 13-1 shows the pros and cons of tiled and overlapping window layouts. This table lists various characteristics of window layouts and identifies the type of window layout which is superior in each category with a star. The strengths of a tiled window layout are that it requires little window manipulation by the user [Bly 86], displays windows in an organized and uncluttered manner, supports multi-tasking on the part of the user, and allows standardized window locations. The strength of an overlapping window configuration is the ability of windows to conform to their contents, thereby maximizing the visibility of these contents [Bly 86]. CUBRICON uses a tiled windowing approach as a default, but allows the "tiled" windows to overlap adjacent windows when necessary based on window contents. This allows CUBRICON to realize the advantages of both types of windowing systems. An overlapping window configuration is shown in Figure 13-3.

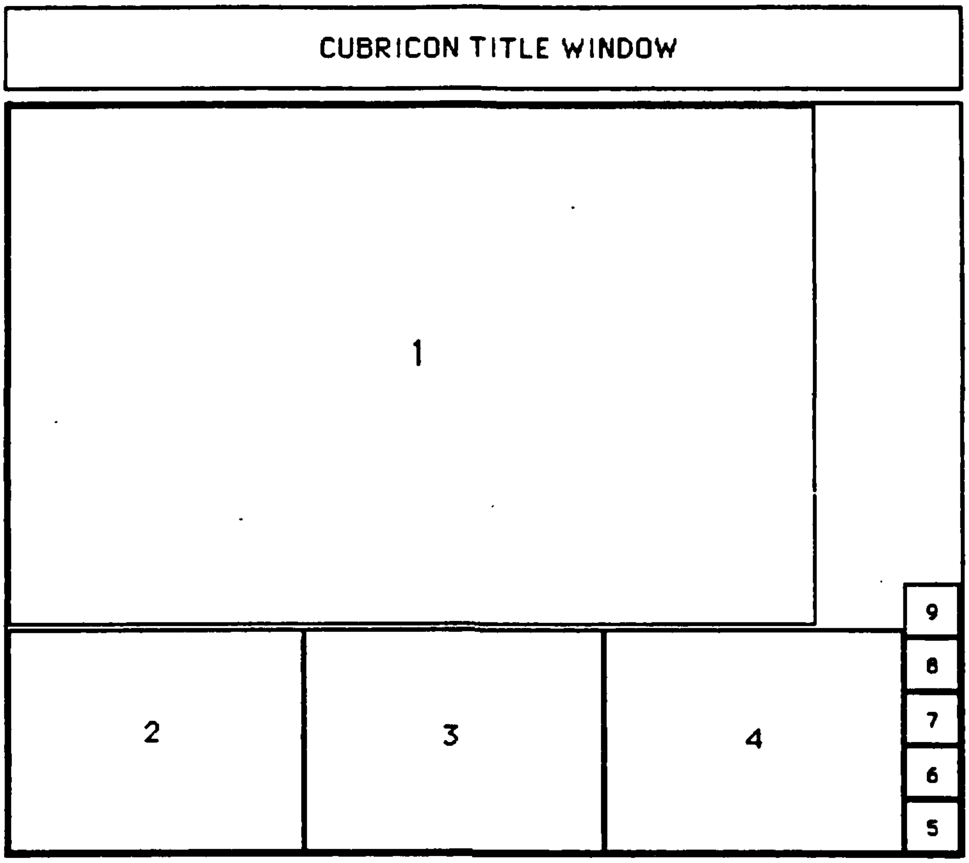


Figure 13-2: The Basic Tiled Window Layout.

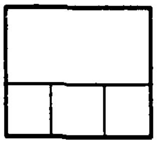
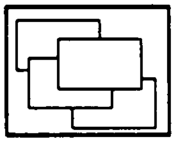
	TILED 	OVERLAPPED 
ABILITY TO OPTIMALLY SIZE WINDOWS	FAIR	☆ GOOD
NUMBER OF WINDOWS ON SCREEN	LOW	HIGH
WINDOW MANIPULATION REQUIRED OF THE USER	☆ LOW	HIGH
DISPLAY APPEARANCE	☆ ORGANIZED UNCLUTTERED	DISORGANIZED CLUTTERED
SUPPORTS MULTI-TASKING ON THE PART OF THE USER	☆ HIGH	LOW
CAN STANDARDIZE WINDOW LOCATIONS	☆ YES	TO A LIMITED EXTENT

Table 13-1: The pros and cons of the tiled and overlapping window layouts.

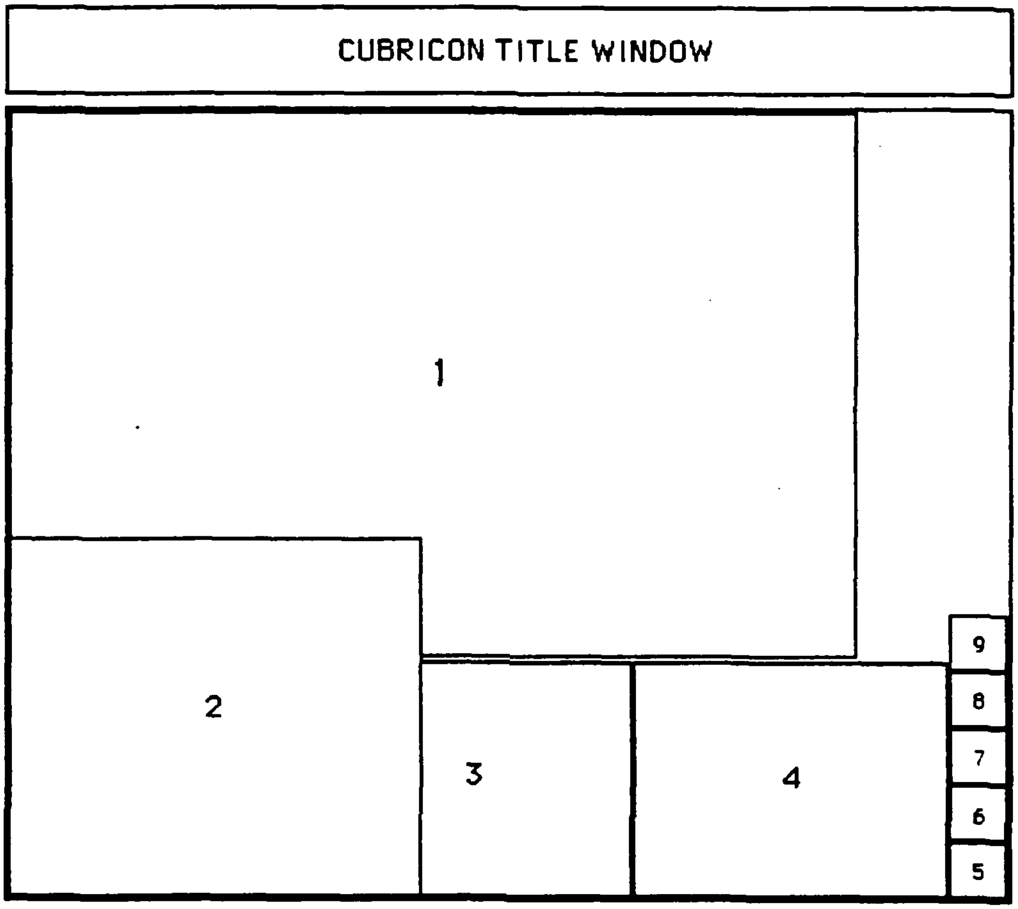


Figure 13-3: The System Provides for Overlapping Windows

One type of window in CUBRICON often requiring an overlapping configuration is the map window. The size of a map windo\v is determined by an algorithm described in Section 13.4. This algorithm computes the proper size of a map window based on clutter analysis which considers the density of icons and labels. Using this algorithm, the size of a window may exceed the size of the pre-defined window position, requiring it to overlap adjacent windows.

Another type of window often requiring an overlapping configuration is the table window. If the columns of a table exceed the default width of the pre-defined window position, then the table may overlap other windows horizontally.

13.3 Intelligent Window Placement

The placement of windows by the window management system combines heuristics for each type of window and generalized window placement logic. The type of window determines the media on which to place the window as well as the positioning of each window on a display. The placement of map and table windows in one of the four pre-defined window positions on the color graphics CRT is based primarily on the window positions available and the relative importance of the window being placed compared with the windows currently on the display. A detailed description of the window importance algorithm appears in a later section.

13.3.1 Map and Table Placement

Maps and tables are the two window types currently placed in one of the four pre-defined window positions on the color graphics CRT. Maps are placed only on the color graphics display, while permanent table windows are placed either on the color graphics or monochrome display. The monochrome display is preferred for permanent table placement in order to leave the color screen available for map displays. However, if there is not an available window position on the monochrome display the table is usually placed on the color graphics display.

As previously stated, the placement of windows in one of the four pre-defined window positions on the color graphics CRT is based primarily on the window positions available and the relative importance of the window being placed compared with the windows currently on the display. In general, if there is a window position vacant, then the window will be placed in an available window position. Generally, window positions are filled in the following order; top, lower left, lower middle, then lower right. If a permanent table window would normally be put on the monochrome display, but all of the pre-defined window positions are filled and there is space for the window on the color graphics display, then the window manager places the window on the color graphics display. If a window is to be placed on the color graphics display, but all of the pre-defined window positions there are filled, then the window manager removes the least important window, transforming it into an icon, to make space for the new window. The algorithm used to determine the importance of a window is in Section 13.3.4

The window placement alogorithm for the monochrome display has not been developed to the same extent as the window placement algorithm on the color graphics display. For example, new windows are allowed to be placed over existing monochrome windows and the ability to iconize monochrome windows has not been implemented. A future enhancement to CUBRICON is to further devlop the monochrome placement algorithm to include the iconization of windows, as has been implemented for the color graphics CRT. One type of table which can be placed over an existing monochrome window is the temporary table (see Section 13.1). A temporary table is displayed by CUBRICON for just one dialogue cycle. Therefore, if all of the monochrome window positions are occupied, rather than occupying a color graphics position, the table is placed over an existing monochrome window. Another case in which monochrome windows are placed over an existing window is when all window positions on both the monochrome and color graphics displays are occupied and the window to be placed is more important than the current monochrome windows. Finally, the mission planning form window is always placed over existing monochrome windows.

The remainder of this section describes the placement alogorithms in detail. The algorithm for placing windows on the color graphics display is:

```
PlaceOnColorGraphicsCRT:
IF the new window is a part-whole decomposition window
  OR a table related to a part-whole decomposition window THEN
       IF one of the window positions 2-4 is available THEN
              Place window in the next available window position other than
                window position 1
       ELSE (positions 2-4 are filled)
              Determine the least important window in position 2-4
              Iconize the least important window
              Place window in this available window position
       END IF
ELSE (not part-whole decomposition window AND
  not a table related to a part-whole decomposition window)
       IF a window position is available THEN
              Place the window in the next available window position
       ELSE (window position is not available)
              Determine the least important window
              Iconize the least important window
              Place the new window in this available window position
       END IF
END IF (part-whole decomposition window or related table)
```

The algorithm which decides whether to place a table on the monochrome display, on the color display, or whether to allow this window to overlap other monochrome windows is:

PlaceTableWindow:

```
IF a monochrome window position is available THEN
        PlaceOnMonochromeCRT
ELSEIF a color graphics window position is available
  AND the table is not a temporary table THEN
        PlaceOnColorGraphicsCRT
ELSEIF the table is more important than the current monochrome windows
  OR the table is temporary THEN
        Place the table in the bottom right corner of the main display
ELSE
        PlaceOnColorGraphicsCRT
ENDIF (table window position cases)
```

The algorithm for placing tables on the monochrome display when a window position is available is:

```
PlaceOnMonochromeCRT:
IF there is a color graphics window related to this table window THEN
        Place table on the monochrome display in the same relative position as the
          position occupied by the related window on the color display
ELSE IF the table can be placed one of the in monochrome positions 2-4 without
  overlapping another window THEN
        Place the table in the available tiled monochrome window position 2-4
ELSE
        Place the table in window position 2
ENDIF (table window placement cases)
```

13.3.2 Mission Planning Form Placement

A mission planning form window requires nearly an entire display, and is the most important window type in the CUBRICON system. Since mission planning forms do not require a color graphics display, they are placed strictly on the monochrome display, so as not to occupy the entire color graphics device. Due to their relative importance, a request to display a mission planning form is never rejected.

13.3.3 Window Importance Algorithm

The factor used to determine which windows to place in one of the four pre-defined window positions on the color graphics display is *window importance*. If all the window positions are on the color graphics display are already occupied by windows and a new window is to be placed on the display, then the system compares the importance of the new window to the importance of the windows already on the display. If there is a less important window on the display than the one to be added, then the system removes (iconizes) the least important window and adds the new window in its place. If the importance of the new window is less than the importance of each of the windows on the display, then the system does not add the new window. In this case, the Output Planner would need to select a different means (modality) of (for) communicating the information to the user. Window importance is also used, to a lesser extent, in placing tables on the monochrome display (see Section 13.3.1).

The calculation of window importance is based on the following items:

- the time the window is created,
- the contents of the window,
- the time elapsed since the last interaction with the window,
- the frequency of interactions with the window, and
- the context of the window.

The time of the last interaction and the frequency of window interactions are based partly on the total number of window interactions. A *window interaction* occurs whenever a window is accessed either by the user or by the system. A maximum of one interaction can be accumulated per window per dialogue cycle. A dialogue cycle consists of a user input and a system response. The following is the formula that the system uses to calculate importance for a window. The coefficient of each term indicates the relative weight that CUBRICON assigns to the particular component of window importance. The components of window importance are discussed in the following subsections.

$$\begin{aligned}&\mathit{Importance} =\\&3.5\,\mathit{TimeOfCreation} + 3\,\mathit{Contents} + 1.5\,\mathit{TimeSinceLastInteraction} + \mathit{FreqOfInteractions} + \mathit{Context}\end{aligned}$$

13.3.3.1 Time of Creation This factor measures the importance of a window based on how recently it was created. The importance of a window, as perceived by the user decreases with time. This is a derivative of the law of exponential decay of information in short-term memory. The value of *TimeOfCreation*, shown below, is an exponential function of the *number of user interactions with* CUBRICON since the window was created divided by 10. The exponential power is divided by 10 in order to get a smooth and reasonable decrease in the value of *TimeOfCreation*.

$$TimeOfCreation = e^{-\frac{CurrentTime - TimeCreated}{10}}$$

The variable *CurrentTime* is the current dialogue cycle number and *TimeCreated* is the dialogue cycle number at the time the window was created. When the system is initialized, the dialogue cycle number is 1 and the dialogue cycle number increments by 1 for each occurrence of user input followed by system output.

13.3.3.2 Contents This factor measures the importance of the window contents with respect to current mission or task. All objects represented by icons on a map or rows in a table are not equally important. For example, when planning an OCA mission SAM systems are more important than heliports. The value of the window contents is the average of the importance of the objects contained in the window, as show below. The importance value of an object is defined in the User Model (see Section 4.3).

$$Contents = \frac{\sum_{i=1}^{NumberOfObjects} ImportanceOfObject}{NumberOfObjects}$$

13.3.3.3 Time Since Last Interaction The *TimeSinceLastInteraction* measures the importance of the window based on how recently the window was accessed, by either the user or the system. The more recently a window is accessed the more important it is. *NumOfDialogCyclesSinceLastInteraction* is the total number of dialogue cycles that have occurred since the last interaction with the window. The *TimeSinceLastInteraction* is an exponential function to ensure a relatively rapid affect of the time that has elapsed since last interaction with the window.

$$TimeSinceLastInteraction = e^{-NumbOfDialogCyclesSinceLastInteraction}$$

13.3.3.4 Frequency of Interaction This factor measures the importance of a window based on the frequency of interactions with the window. As mentioned previously, the access of a window by either the user or the system counts as a window interaction. The more often a window is accessed the more important it is. The frequency of interaction is the percentage of dialogue cycles that involved an interaction with the window. The following formula is used to measure the frequency of interaction.

$$FreqOfInteractions = \frac{NumbOfInteractionsWithWindow}{NumbOfDialogCyclesSinceCreation}$$

The variable *NumbOfInteractionsWithWindow* is the total number of interactions with this window. The variable *NumbOfDialogCyclesSinceCreation* is the total number of dialogue cycles that have occurred since the the window was created. For a window created during the current user interaction, the value of *FreqOfInteractions* is set to 1. This factor balances against the weight assigned to the *TimeSinceLastInteraction* defined above. For a window

Purpose of the Window	Command Example	Context Value
Mission Planning Form	Display the Form.	1.0
Zoom in on a target area	Zoom in	1.0
Geographic Map	Display the FG region.	0.75
Part-Whole Decomposition Map	What are the aimpoints within the Dresden airbase?	0.75
Zoom in in a non-target area	Zoom in	0.5
Unrelated table	List the packages.	0.5
Related table	automatically generated	0.0
Context window	automatically generated	0.0

Table 13-2: Context value included in the importance factor.

that was previously frequently accessed, if *TimeSinceLastInteraction* indicates that it should be removed, *FreqOfInteractions* says "lets not be too hasty." That is, it causes windows that have been important, in terms of their frequency of use, to be removed after more careful consideration of their ongoing use.

13.3.3.5 Context of a Window This factor measures the importance of a window to the current mission planning task. Certain windows are more critical to the mission planning task than others. For example, map windows are more critical when planning a mission than table windows. Also, maps containing the target of a mission are more critical than maps that do not. Table 13-2 identifies the values assigned to the context of a window based on the window's functionality.

13.4 Sizing Map Windows

One of the functions of the Intelligent Window Manager is to decide the minimum acceptable size for a map window. The minimal size of a map is based on the density of display objects (e.g., icons, labels). This is important because when display objects are packed too closely together, the map becomes difficult to read (i.e., the ease of extracting information from the map is reduced). For example, it becomes difficult to tell which labels go with which icons and some icons may overlap making them difficult to recognize. This reduction in readability can be measured as increased time or decreased accuracy for tasks using the map. These measures of reduced usability are said to indicate clutter [Potash77].

The CUBRICON map window sizing methodology is based on a measure of clutter for map windows. The clutter analysis algorithm is based on the following premise: For the map displays of various sizes that would fit on a typical high-resolution workstation screen (the CUBRICON workstation screens have a usable display area that is approximately 10.5 inches by 13.5 inches), a map would be perceived as too cluttered if there is any circular subregion S, with radius between approximately 0.25 inches and 1.25 inches, such that the clutter measure of subregion S exceeds the clutter tolerance for an area of its size. This means that it is not sufficient to compute the percentage of the *whole* map that is occupied by iconic display objects, but that (ideally) every subarea of the map of every possible size should be examined to determine whether it exceeds the clutter tolerance for its size. Furthermore, the clutter tolerance (maximum value of the clutter measure) for subregions of a map display varies inversely with the area of the subregion, but the proportionality is not constant.

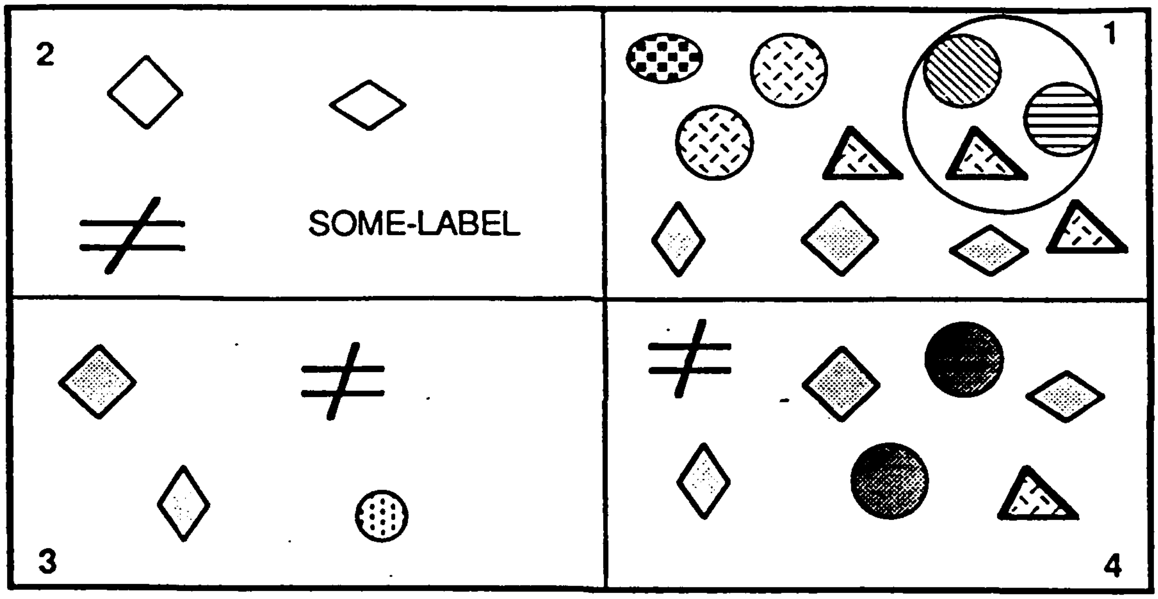


Figure 13-4: Hypothetical Map Display

Consider the hypothetical map display of Figure 13-4 which has been subdivided into four quadrants. Quadrant 1 appears to be too cluttered for its size even though any circular subarea of the size shown in the figure would be acceptable. Thus, it does not seem to be sufficient to use a pre-defined fixed area size for the purposes of determining acceptability of map displays with respect to clutter. That is, it does not seem appropriate to use an algorithm which simply subdivides the map into pieces of size S, where S is pre-defined, and count the number of objects per unit area for each piece of size S. For the map of Figure 13-4, the map would probably be acceptable if S were of a size equal to the entire map, unacceptable if S were of a size equal to a quarter (quadrant) of the map, and acceptable if S were of a size equal to the area enclosed within the circle of Figure 13-4. Thus, it seems as if the results would be inconsistent. To account for this problem, our clutter analysis

Radius in Inches for Critical Area	*Acceptable Clutter in Number of Objects*
0.25	4.00
0.75	10.00
1.25	16.00

Table 13-3: Clutter Measure Table

algorithm is based on the premise that, ideally, every subarea of the map of every possible size should be examined to determine whether it exceeds the clutter tolerance for its size.

The maximum tolerable clutter measure for areas of different sizes is displayed in Table 13-3. The results shown in Table 13-3 are based on the engineering judgement of our human factors research personnel and need to be subjected to experimental testing. The results seem reasonable based on extensive study of the types of map displays used on this project.

In counting objects within an area, labels are weighted more heavily than icons since the size of a label is approximately 1.5 times the size of an icon. In computing the clutter for any area A of a map in the CUBRICON system, the following formula was used:

$$ClutterMeasure = IconsInCA + 1.5LabelsInCA$$

The variable *IconsInCA* represents the number of icons in the critical area and the variable *LabelsInCA* represents the number of labels in the critical area. An icon is considered to be within the area if the area intersects the extent of the icon (the smallest rectangle enclosing the icon). A label is considered to be within the area if the area and any point on the label intersect. To simplify this discussion we assume that all icons are approximately of the same size.

The CUBRICON clutter analysis algorithm uses the *critical area radii* along with their corresponding acceptable (maximum) clutter measures shown in Table 13-3. The evaluation conducted during this effort (see Section 15) indicates that the clutter algorithm based on these areas and acceptable clutter measures is currently performing pretty well. These values can be adjusted if this appears to be warranted based on experience in using CUBRICON, or as a result of research designed for this purpose.

The CUBRICON clutter algorithm entails up to four levels of clutter analysis for any given map. These are zero or more of the following levels: the quadrants, circular areas of radius 1.25, circular areas of radius 0.75, circular areas of radius 0.25.

CUBRICON decides whether to perform clutter analysis for each critical area size on a quadrant-by-quadrant basis. This is done based on *screen density* which is based on the number of display objects present in the quadrant. If there are only a few display objects in a map quadrant, for example, there is no need to perform clutter analysis for the quadrant based on the larger areas. If many display objects are present, then all critical unit areas must be applied. This is the approach CUBRICON uses to determine which critical area sizes (radii) to use in its clutter analysis within each quadrant.

As noted above, this algorithm is applied on a quadrant-by-quadrant basis. This decision was based on the judgement of the human factors engineers on the project. It is intended to minimize the number of unnecessary calculations performed without sacrificing functionality. It recognizes the fact that icon density is not likely to be evenly distributed across an entire map display.

Once the critical unit areas to be applied for each quadrant are determined, they are applied on an icon-by-icon basis. That is, the degree of clutter in the immediate area (i.e., using one of the critical radii to determine the area) around each icon is computed. When the most crowded (i.e., cluttered) icon is found for each critical unit area, it is compared to the predetermined criteria from Table 13-3. If the acceptable clutter measures are not exceeded for these worst-case icons, then the map is not too cluttered. If the acceptable clutter measures are exceeded, the map size (i.e., the window size) is increased to a size that reduces clutter factor for the worst-case icon to within the acceptable range. Since this resizing is based on worst-case icons, it ensures that all other icons will also be brought within the acceptable range. The amount of resizing needed is calculated directly from the clutter factor calculation.

13.4.1 The Map Sizing Algorithm

The first step in determining map size requirements is to determine whether the planned map is too cluttered. As described in the preceding subsection, this determination is made by analyzing clutter within each map quadrant. An appropriate criteria to be used in the analysis is selected for each quadrant based on the number of display objects contained within the quadrant. One or more "critical area sizes" may be used. If a quadrant is found to be too cluttered, the entire map size is increased to a level where clutter is within acceptable levels. The following algorithm accomplishes this:

```
Calculate four equal quadrants for the map
FOR each quadrant $Q$
    Calculate $ScreenDensity_Q$ for this quadrant
    Table-Lookup the critical area radii, if any, to use based on screen density
      (see Table 13-4)
    Assign these radii to set $Radii_Q$
END FOR
FOR each critical area radius $r$ in the set $\{0.25, 0.75, 1.25\}$
    Initialize $MaxClutterMeasure_r$ with $AcceptableClutterMeasure_r$ (see Table 13-3)
END FOR
FOR each quadrant $Q$
    FOR each $r$ in $Radii_Q$
        FOR each icon $X$ in quadrant $Q$
            Calculate $ClutterMeasure(X,r)$
            IF $ClutterMeasure(X,r) > MaxClutterMeasure_r$ THEN
                $MaxClutterMeasure_r \longleftarrow ClutterMeasure(X,r)$
            END IF
        END FOR (each icon)
    END FOR (each $r$)
END FOR (each quadrant)
Initialize $MaxIncreaseFactor$ to 1
FOR each value of $r$ in the set $\{0.25, 0.75, 1.25\}$
    Calculate $IncreaseFactor_r$ for increasing the map size
END FOR (each value of $r$)
Assign $MaxIncreaseFactor$ the maximum of 1 and the values of $IncreaseFactor_r$ for $r$ in
  the set $\{0.25, 0.75, 1.25\}$
Calculate the new map length and width
```

The map boundary to be used for clutter analysis is the boundary of the window in which the new map will be placed. If a zoom out operation is being performed, then the new map replaces an existing map and the boundary of the window containing the existing map is used for the clutter analysis. If a new map is being created, a new window is created and a default window boundary will be used for clutter analysis. The default window boundary is dependent on the position in which the map will be placed. Section 13.2, Window Layout, describes the available window positions in detail.

The screen density value approximates the overall clutter of a window. This value is used solely to determine which of the critical area sizes to use in the clutter analysis for the quadrant. Screen density for each quadrant is calculated as follows:

$$ScreenDensity = NumberOfIcons + .5NumberOfLabels$$

The variables NumberOfIcons and NumberOfLabels represent the total number of icons and the total number of labels on the map, respectively. In this calculation, labels have less weight than icons. This is due to the difference in icon and label placement methodologies. CUBRICON utilizes an intelligent label placement algorithm which labels an icon only if

Range for the Screen Density Value	Use the Following Critical Area Sizes (Radii)
0 to 5	Not required
5.5 to 10	.25in.
10.5 to 15	.25in., .75in.
15.5 and higher	.25in., .75in., 1.25in.

Table 13-4: The selection of the critical area sizes for a quadrant is based on the screen density in that quadrant.

space is available. Therefore, a label will not overlap an icon or another label. An icon, however, is placed at a particular location and may overlap another icon. Therefore, it is assumed that labels contribute to clutter half as much as icons contribute to clutter. Table 13-4 indicates which critical area sizes are to be used for the different values of screen density.

The clutter algorithm determines the actual clutter measure as a function of the critical area size. As discussed previously, the formula for computing the ClutterMeasure is:

$$ClutterMeasure = IconsInCA + 1.5LabelsInCA$$

Labels are weighted more heavily than icons since the size of a label is approximately 1.5 times the size of an icon in the CUBRICON system. The variable *IconsInCA* represents the number of icons in the critical area and the variable *LabelsInCA* represents the number of labels in the critical area. An icon is considered to be within the critical area if the area intersects the extent of the icon. A label is considered to be within the critical area if the area and any point on the label intersect.

The optimal map size is based on the maximum clutter value and the acceptable clutter value for each critical area size. These values are used in calculating the percentage by which to increase the pre-defined map area. A candidate multiplier to be applied to the pre-defined map area is calculated as follows for each critical area radius *r*:

$$IncreaseFactor = 1 + \frac{MaxClutterMeasure - AcceptableClutterMeasure}{AcceptableClutterMeasure}$$

The maximum value of *IncreaseFactor* for the different critical area radii is used to calculate the final boundary values. The length and width of the map are increased as follows.

$$NewLength = Length\sqrt{IncreaseFactor}$$

$$NewWidth = Width\sqrt{IncreaseFactor}$$

The map sizing methodology presented in this section seems to be performing pretty well, based on the evaluation conducted during this project. However, it needs to be subjected to additional experimental testing to evaluate its effectiveness. The values used for the critical area radii and acceptable clutter measures are based on the engineering judgement of the human factors research personnel on this project after extensive study of the various types of map diplays used for the CUBRICON system. We believe that this map sizing methodology (or a refined version of this methodology) has significant potential for complex military applications.

14 Knowledge Base Builder Tool

Relational Database Management Systems (RDBMS) have become widespread in recent years, and many large relational data base (RDB) systems have been built. These systems, however, usually do not provide advanced intelligent interfaces and their use requires training in a data base (DB) language. The interfaces usually provided with such RDB systems generally consist of computer forms with simple record search and updating capabilities. Recent advances in human computer interface (HCI) technology provide the potential to overcome such limitations and make RDBs more readily accessible to personnel without special training. However such advanced HCI systems are often built to work with KBs and not RDBMSs.

This section discusses a tool, the Knowledge Base Builder, that was constructed to support the integration of the RDB underlying the AMPS Mission Planning system, and the KB underlying an intelligent multi-media interface (IMMI) system, CUBRICON. It is a hybrid tool in the sense that it operates on a knowledge base *and* a database system.

The KB used by the CUBRICON system is implemented in a Semantic Network Processing System (SNePS). The construction of KB's using this system involves the use of an editor (ZMACS in our case) to write files which contain semantic network building statements written in the SNePS user language (SNePSUL). This is a labor intensive process and does not easily support changes to the KB.

Without the aid of a tool such as the KB Builder here the process of building a KB in SNePS that corresponds to the data in AMPS would be a large effort in itself. It involves working from listings of the AMPS tables. For each piece of data in the listings the person doing the translation would type the corresponding SNePSUL statements. This transformation of the AMPS data into SNePSUL statements is not straight forward and requires thought on the part of the translator. The translation process does not condense the amount of text required but increases it. This means that is the AMPS DB were to be manually translated in this way the resulting text file of SNePSUL statements would be much larger than the original listings of the tables of the AMPS DB.

The KB Builder supports the integration of the KB and the RDB by providing the ability to construct KBs which are *linked* with the AMPS RDB. The tool does this by providing four main capabilities, a RDB browsing capabilities, a link construction capability, semi-automatic KB generation capabilities, and an interactive KB editing capability.

These capabilities are provided by the tool through a direct manipulation interface (windows, icons, menus, and pointing). This type of interface was chosen because of its easy of use *and* its ease of construction. Ultimately tools of this sort may have IMMI interfaces such as the CUBRICON system. However, while the development of such an interface would be a worth while research project, it would not be an appropriate part of the CUBRICON project.

The link construction capability enables the definition of links between CUBRICON concepts (implemented in SNePS) and data in the AMPS RDBMS. Once these links are defined they can be used to build concepts in the KB from RDB data. This is the KB generation capability. In addition to the linkage and KB generation capabilities an interactive KB editing capability and a RDB browsing capability have also been provided to support the generation of skeletal class structures and the specification of KB structures for which no data exists in the KB.

The process by which the capabilities of the KB Builder are used to develop a KB and link it to a RDB is as follows:

1. Build a skeletal class structure in the SNePS KB using the interactive editing capabilities of the tool.
2. Build KB structures called *links* and place them in the skeletal class structure. Using the tool this is done in a interactive semi-automatic manner or by using the KB editing capabilities.
3. Identify instances of the skeleton classes from data in the RDB and build KB nodes representing them.
4. Build KB structures associating RDB information with each instance node.
5. Build additional KB structures manually using the KB editing capabilities.

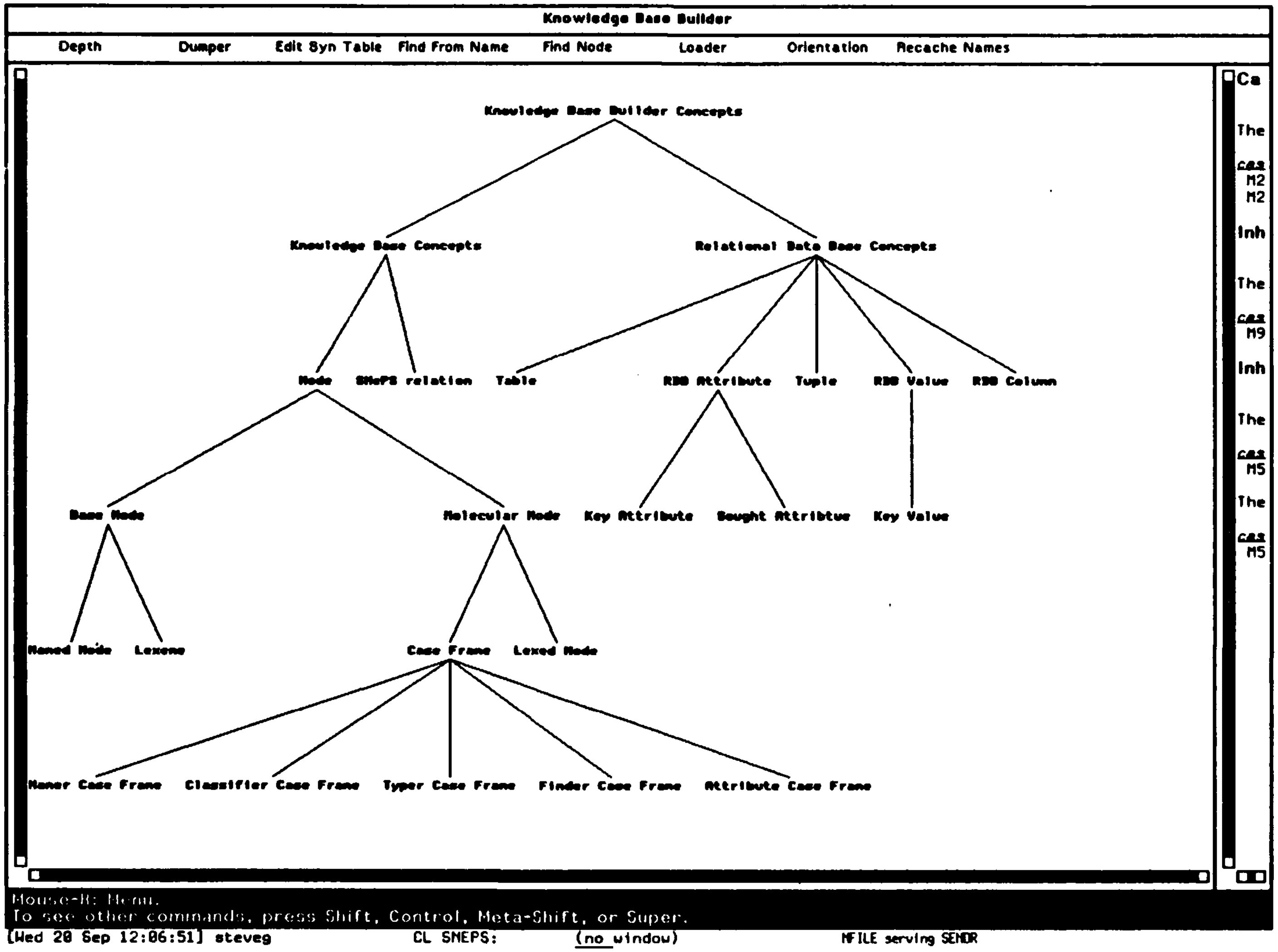
An understanding of the capabilities provided by the tool requires an understanding of the concepts used in RDBM's and the SNePS system. Additionally a specialized concept of a RDB-SNePS link must also be discussed. This concept forms the basis for understanding the linkage definition and KB generation capabilities of the KB Builder.

14.1 System Concepts

The concepts that are involved in the use of the KB Builder are presented in Figure 14-1. The SNePS based concepts are discussed in detail in Section 4.1 and the RDB concepts are described in the literature. In this section we will examine two concepts which are essential to an understanding of the KB Builder's capabilities, the link concept, and the case frame type concept. Later sections discussing the tools capabilities and displays will rely on an understanding of these concepts.

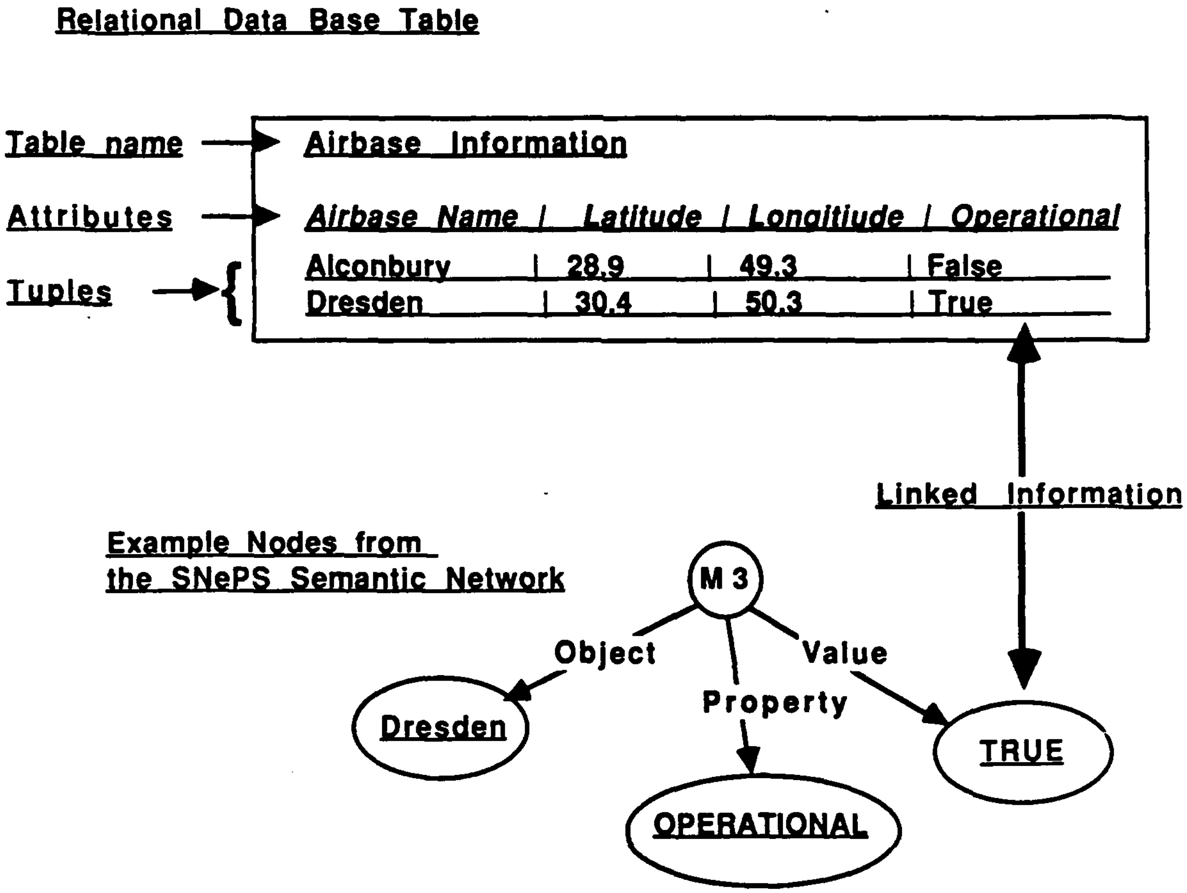
14.1.1 The Link

The link associates information from a RDB table with nodes in a semantic network that represent instances of a given class. The relationship between the instance and the node



```
Mouse-R: Menu.
To see other commands, press Shift, Control, Meta-Shift, or Super.
[Wed 20 Sep 12:06:51] steveg          CL SNEPS:      (no window)          NFILE serving SENOR
```

Figure 14-1: Hierarchy of Knowledge Base Builder Concepts



Airbase Name	Latitude	Longitude	Operational
Alconbury	28.9	49.3	False
Dresden	30.4	50.3	True

Figure 14-2: An Example Linkage

containing the linked information is accomplished through a case frame consisting of molecular node and a set of arcs (see Section 4.1). Figure 14-2 illustrates an example of such a linkage.

Please note that the figures of this section show abbreviated versions of the semantic networks. Nodes such as the Dresden node of Figure 14-2 do not really have the name "Dresden" but rather a node containing the name "Dresden" is related to the node representing dresden through a special case frame called the namer case frame. Such KB structures are provided so that the KB can distinguish between the concept of the dresden air base itself and the concept that the air base's name is Dresden. Section 4.1 provides many examples of the complete form of such networks. However, there is not enough room in our figures for all of the nodes and arcs that would result if all these naming structures were included. Such naming structures will be understood when a name in a node found in a figure is underlined.

The link itself is stored in the semantic network as a case frame (cf. Figure 14-3) which

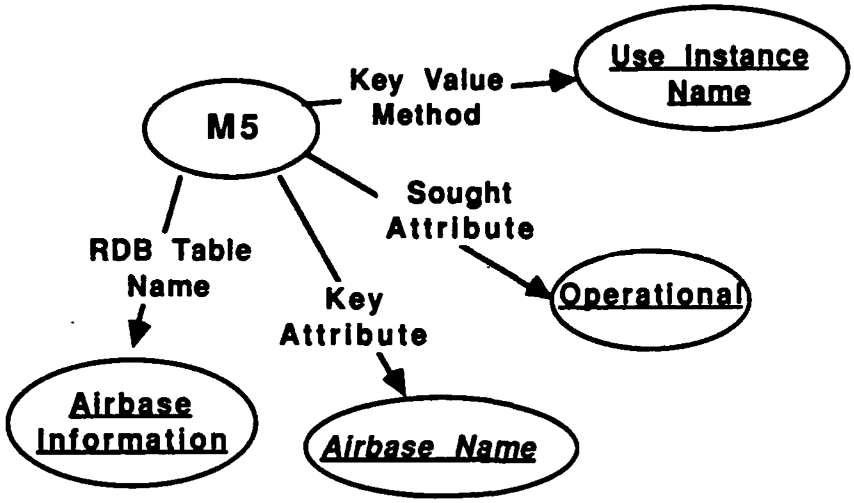


Figure 14-3: An Example Link

relates four pieces of information together.

- the name of a RDB table from which to extract information,
- the name of a RDB table attribute which is to be used as a key,
- the name of another attribute which is to be used to extract values from the table, and
- an identification of the manner in which the key values are to be generated.

The link is related to a class node in the network through another case frame such as in Figure 14-4. In this figure the relating case frame is represented by molecular node M2 which has three arcs descending from it, object, property, and value. The form of the case frame that links the class node to the node representing the link has a special purpose. The same case frame is used to relate the information found in the RDB to instances of the given class. This is illustrated in Figure 14-4 by the case frame represented by M3 which is of the same form as the case frame represented by node M2.

The use of links to generate KB structures from the contents of the RDB proceeds as follows.

- the RDB table is identified using information stored in the link.
- the name of a key attribute (stored in the link) is used with the name of the instance (Dresden in Figure 14-4) to identify the relevant tuples from the table.

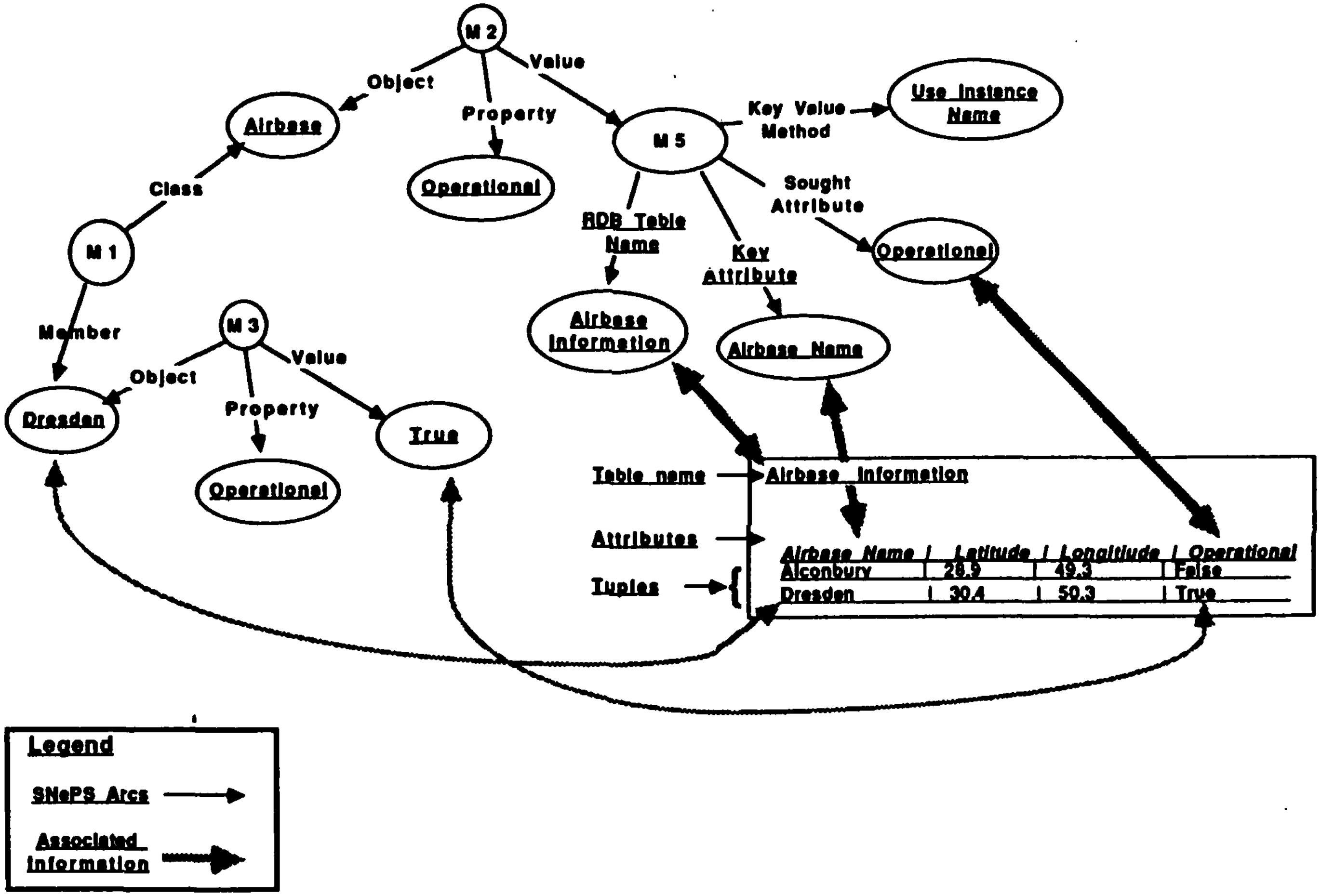


Figure 14-4: The Full Context of a Link

- the name of the sought attribute (Operational in the figure) is used to extract a set of values from the table.

- Case frames (such as the one represented by M3) are created which relate the extracted information to the node representing the instance. This case frame is patterned after the relationship between the class node and the link node.

14.1.2 The Case Frame and Case Frame Type concepts

The case frame concept as used by the CUBRICON and KB Builder system refers to a standard pattern of molecular nodes and arcs which relate a set of nodes together. Examples of such case frames are seen in Figures 14-5, and 14-6. For reasons of flexibility and SW development economy the tool's ability to manipulate case frames has been organized around the concept of abstract case frame types. This concept and the associated concepts such as related-nodes and the namer case frame are discussed in this subsection.

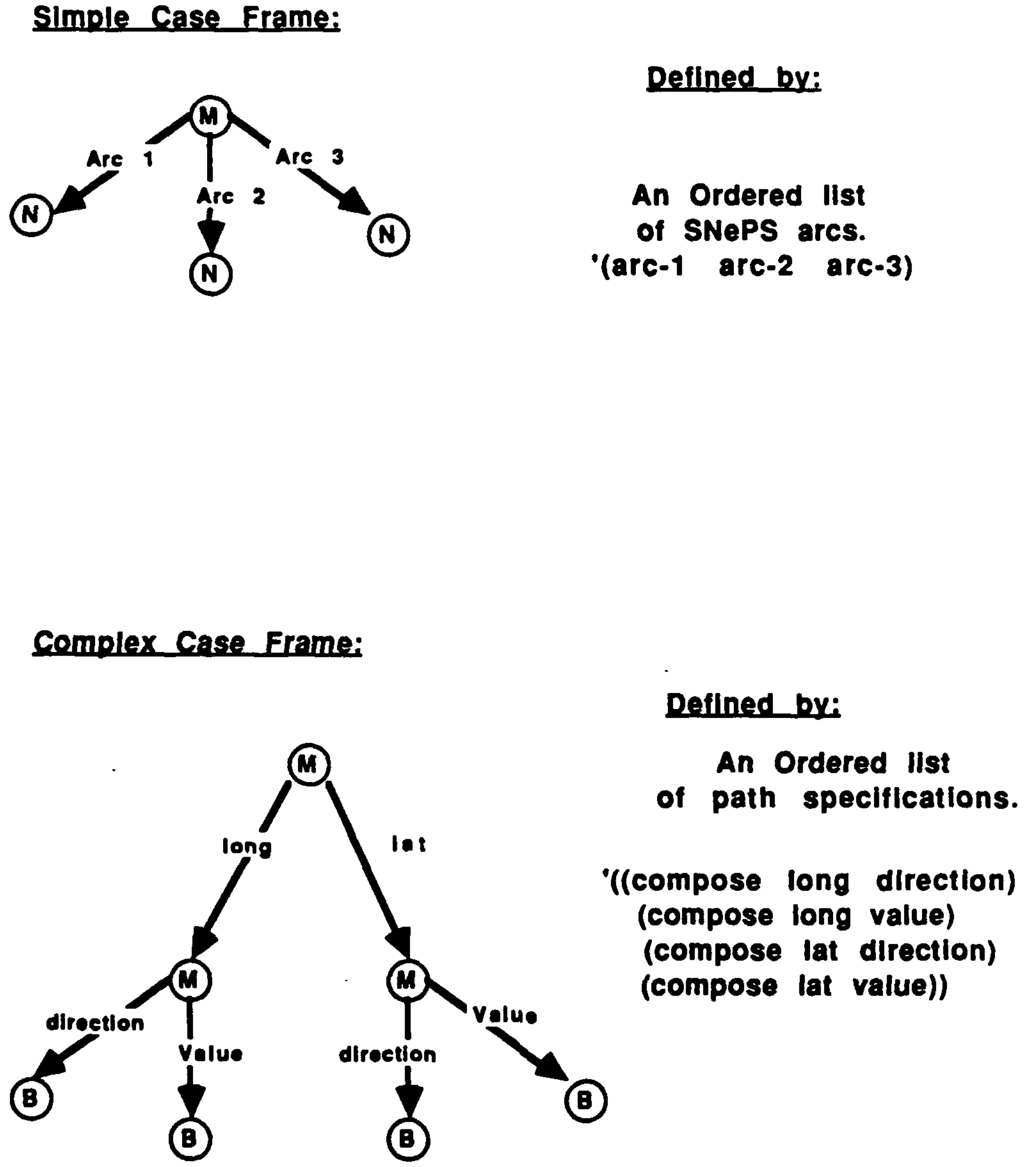
There are two basic requirements that lead to this approach. The first is a requirement to be able to dynamically specify new case frames and to modify existing ones without having to modify or extend the tool. The need for this became apparent during the course of the development of the KB Builder tool as requirements to manipulate new case frames were incrementally added in conjunction with the evolution of the CUBRICON system. In the context of these requirements an examination of the functions that are required to manipulate the case frames was performed which revealed commonality that could be exploited. This led to the definition of a case frame type concept within the system which enabled the case frame manipulating functions to exploit the commonality. Structuring the tool to manipulate case frames of declaratively defined types has the following advantages:

- The KB-builder tool software becomes independent of the choice of case frame representations that were chosen for the KB.

- The ability to examine and manipulate new case frames can be quickly added to the system.

- The organization of the software around abstract case frame type definitions is very compact. In previously investigated approaches each case frame type required its own file containing definitions of the case frame and functions to access, parse, display, edit, and create the case frame.

The approach chosen involved the development of presentation types and accessor/manipulator functions for the SNePS based concepts of a node, a base node, a molecular node, case frames



Simple Case Frame:

Defined by:

An Ordered list
of SNePS arcs.
'(arc-1 arc-2 arc-3)

Complex Case Frame:

Defined by:

An Ordered list
of path specifications.

```
'((compose long direction)
  (compose long value)
  (compose lat direction)
  (compose lat value))
```

Figure 14-5: Simple and Complex Case Frames

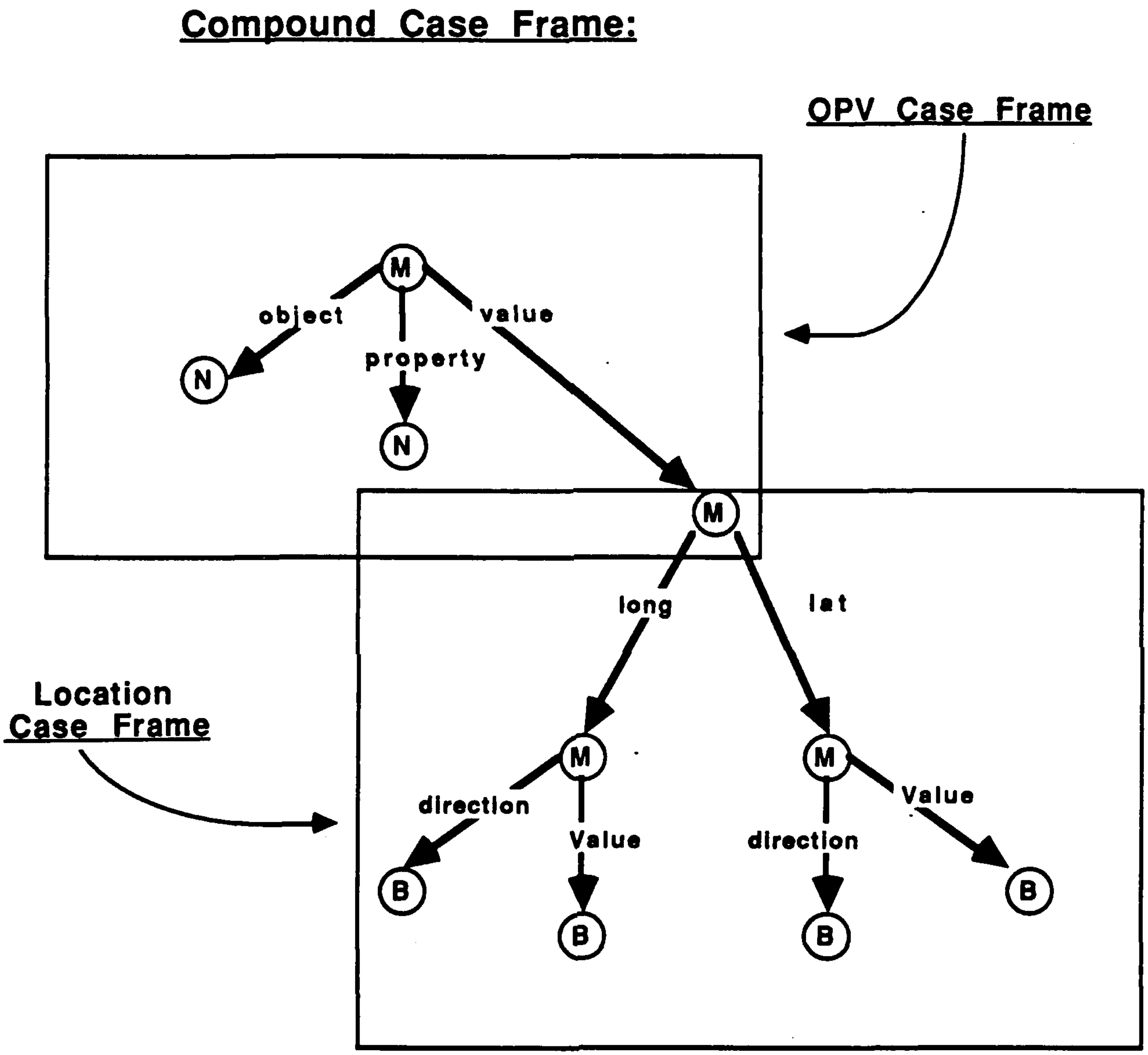


Figure 14-6: Compound Case Frames

in general, "a kind of"(AKO) nodes, and named nodes. It was not necessary to develop special code for each type of case frame used by the system (e.g. the location case frame, the OPV case frame, the PART case frame, ...).

The requirements for a "WYSIWYG" editing capability led to a requirement to be able to easily "rewire" the arcs of the KB. This capability is supported by the rewire function and the **related-nodes** concept which identifies two nodes and an arc that relates them. This is exactly the information that is needed to replace a node in a case frame. The first node of a **related-nodes** instance is often a "non leaf" case frame node, and the second node is often a "leaf node" of the same case frame. When a case frame leaf node is presented on the screen (most of the information in the knowledge base is viewed in this way) a mouse action which can edit this part of the case frame must be able to refer to a related-nodes instance and not just the part itself. If this is not done then the information that the rewire-nodes function needs to replace the part is not available.

The two Symbolics flavors that have been defined to support the definition of case frame types and related nodes are presented in Figure 14-7.

The **case-frame-type** flavor collects all of the information the system needs to:

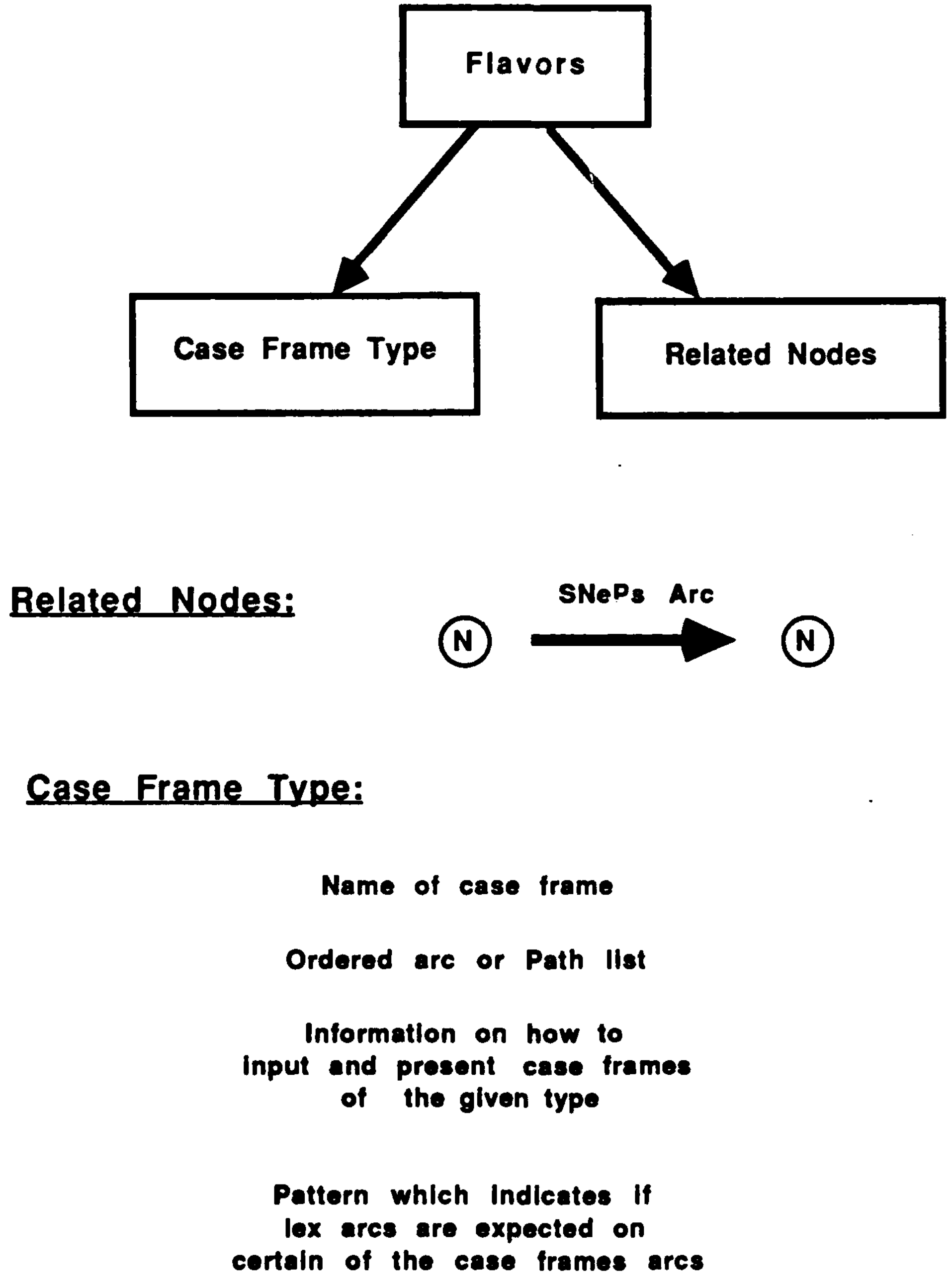
- define the arc structure and order for the case frame type, and
- **scl::accept** and **scl::present** such case frames,

Note that while the arcs in SNePS case frames are unordered we have chosen to add an order to them for display purposes.

For each type of case frame known to the system an instance of the flavor **case-frame-type** is built and stored on the list ***case-frame-list***. The code which manipulates case frames utilizes these definitions. In this way the case frame related information is declarative and not procedural. Several of the case frames thus declared are given special designations. The special case frames include:

- the namer case frame,
- the super-class/sub-class case frame,
- the class-member case frames, and
- the link case frames.

The namer case frame is used to indicate the name associated with a base node. Such a base node is called a named node. By convention the first arc or path of the namer case frame



Related Nodes:

Case Frame Type:

Name of case frame

Ordered arc or Path list

Information on how to input and present case frames of the given type

Pattern which indicates if lex arcs are expected on certain of the case frames arcs

Figure 14-7: Flavors Based Concepts

type definition refers to the named node the second path refers to the node **name** and the third path refers to the name itself.

The super-class/sub-class case frame, and the class-member case frames, are used to define inheritance paths for finders and case frames. They also are used to display the AKO tree found in the left pane of the KB Builder screen. This ability to draw a tree of nodes related by case frames can easily be generalized. Currently the trees are only derived from the super-class/sub-class case frame, and the class-member case frame. In the generalized version the user would choose what case frames are to be used in the generation of the tree. This would allow the graphing of "a part of" trees, chain of authority trees, etc.

The Link case frame is used by the system to define the mappings between the RDBMS and the SNePS KB (cf. Section 14.1.1). Since the link is represented by a case frame embedded in another case frame no special purpose code is required to manipulate or edit these structures. A link is just another type of case frame on the ***case-frame-list*** list.

The system often has to manipulate "compound case frames" such as the case frame of Figure 14-3 or Figure 14-6. Such compound case frames may be nested several layers deep. The display software will handle this by recursively displaying the parts of the case frame.

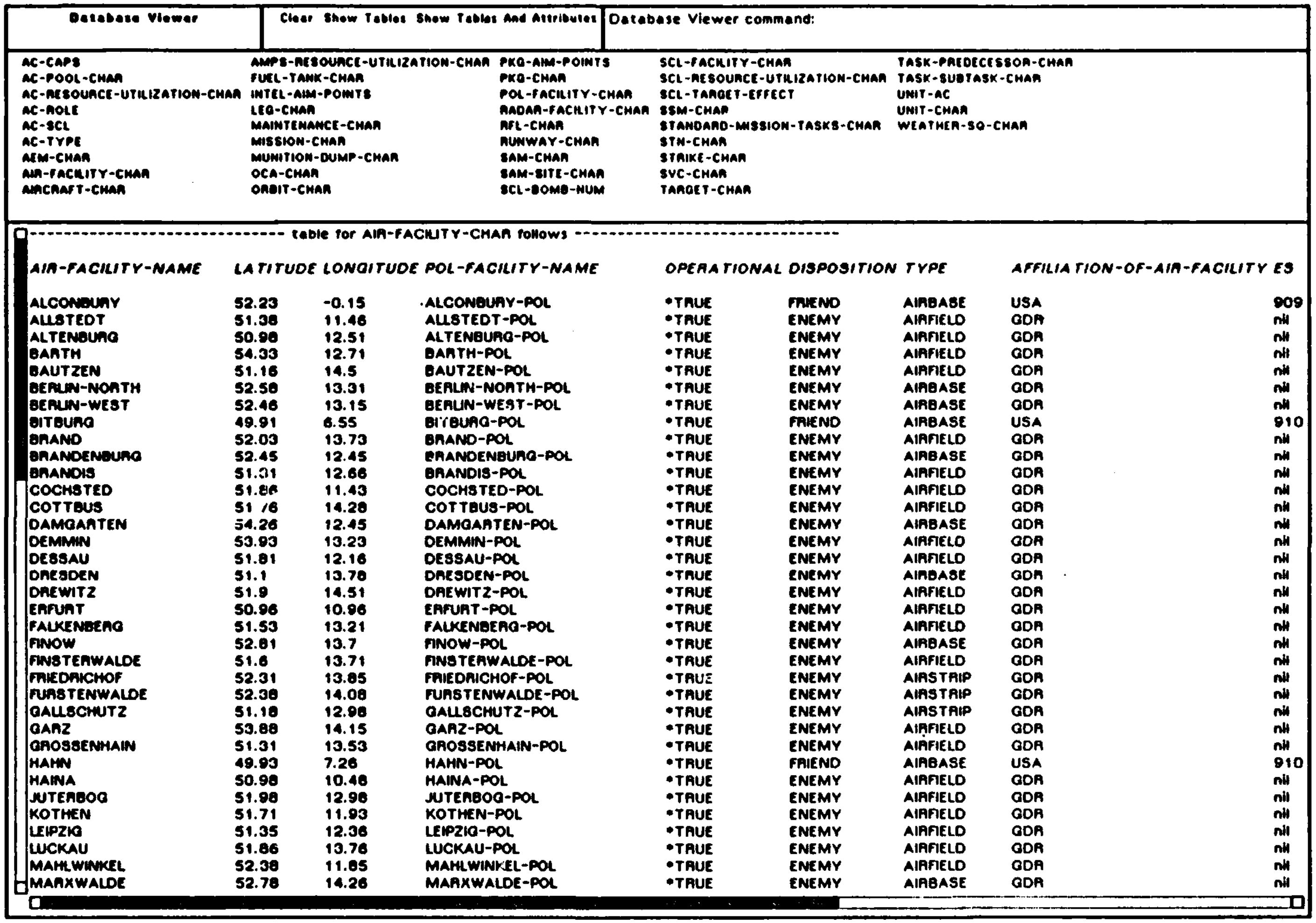
For very deeply nested compound case frames the displays may be more extensive than desired. (This has not been the case in our use of the tool). If such deeply nested case frames are to be represented in the system a limit to the nesting level of such displays can be included. The remaining levels can be displayed in a pop up window via a mouse gesture upon user request.

14.2 KB Builder Human Computer Interfaces

The four main capabilities of the tool are provided through two main interactive direct manipulation displays. One display, the Data Base Viewer, is oriented primarily toward the support of AMPS RDB browsing. The other display is orientated toward, the generation of linkages, the generation of KB structures from the RDB, and editing the KB. Figures 14-8 and 14-9 show the appearance of these displays.

Both displays have a menu of commands at the top and support *keyboard accelerators* for those same commands. The keyboard accelerators provide the means to invoke a command by entering the first letter of each command. This capability is in addition to pointing to the command in the command menu with the mouse.

In addition to the commands presented on the top of the display every object presented in the display can be referred to with the mouse to invoke additional operations specific to that object. Nearly all of the supported activities are performed through such "point and click" references using the mouse. In all cases the operations that are available on the

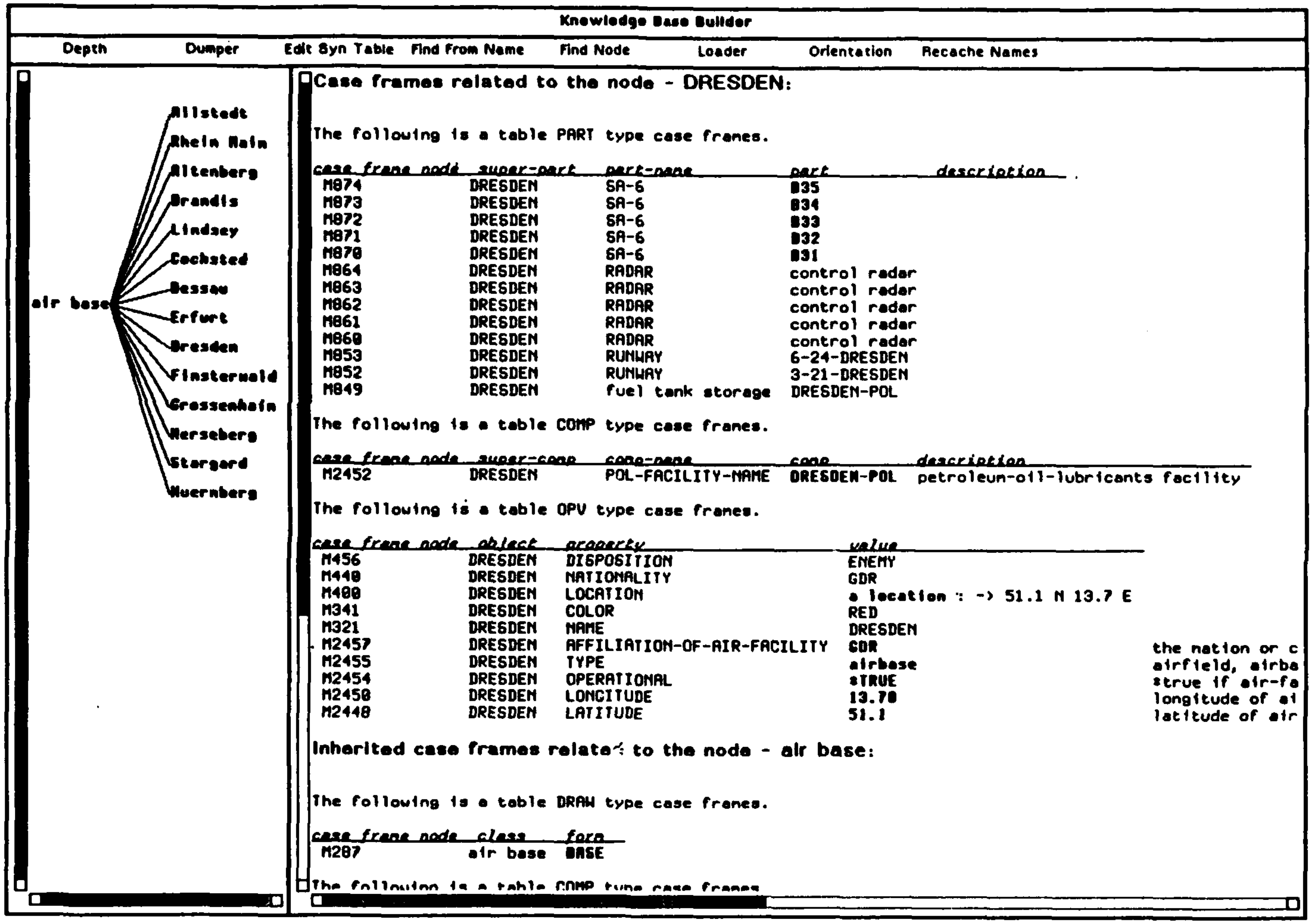


```
AC-CAPS                        AMPS-RESOURCE-UTILIZATION-CHAR PKG-AIM-POINTS     SCL-FACILITY-CHAR              TASK-PREDECESSOR-CHAR
AC-POOL-CHAR                   FUEL-TANK-CHAR                 PKG-CHAR           SCL-RESOURCE-UTILIZATION-CHAR  TASK-SUBTASK-CHAR
AC-RESOURCE-UTILIZATION-CHAR   INTEL-AIM-POINTS               POL-FACILITY-CHAR  SCL-TARGET-EFFECT              UNIT-AC
AC-ROLE                        LEG-CHAR                       RADAR-FACILITY-CHAR SSM-CHAR                      UNIT-CHAR
AC-SCL                         MAINTENANCE-CHAR               RFL-CHAR           STANDARD-MISSION-TASKS-CHAR    WEATHER-SQ-CHAR
AC-TYPE                        MISSION-CHAR                   RUNWAY-CHAR        STN-CHAR
AEM-CHAR                       MUNITION-DUMP-CHAR             SAM-CHAR           STRIKE-CHAR
AIR-FACILITY-CHAR              OCA-CHAR                       SAM-SITE-CHAR      SVC-CHAR
AIRCRAFT-CHAR                  ORBIT-CHAR                     SCL-BOMB-NUM       TARGET-CHAR
```

------------------------------ table for AIR-FACILITY-CHAR follows ------------------------------

AIR-FACILITY-NAME	LATITUDE	LONGITUDE	POL-FACILITY-NAME	OPERATIONAL	DISPOSITION	TYPE	AFFILIATION-OF-AIR-FACILITY	ES
ALCONBURY	52.23	-0.15	ALCONBURY-POL	*TRUE	FRIEND	AIRBASE	USA	909
ALLSTEDT	51.38	11.46	ALLSTEDT-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
ALTENBURG	50.98	12.51	ALTENBURG-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
BARTH	54.33	12.71	BARTH-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
BAUTZEN	51.16	14.5	BAUTZEN-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
BERLIN-NORTH	52.58	13.31	BERLIN-NORTH-POL	*TRUE	ENEMY	AIRBASE	GDR	nil
BERLIN-WEST	52.46	13.15	BERLIN-WEST-POL	*TRUE	ENEMY	AIRBASE	GDR	nil
BITBURG	49.91	6.55	BITBURG-POL	*TRUE	FRIEND	AIRBASE	USA	910
BRAND	52.03	13.73	BRAND-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
BRANDENBURG	52.45	12.45	BRANDENBURG-POL	*TRUE	ENEMY	AIRBASE	GDR	nil
BRANDIS	51.31	12.66	BRANDIS-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
COCHSTED	51.86	11.43	COCHSTED-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
COTTBUS	51.76	14.28	COTTBUS-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
DAMGARTEN	54.26	12.45	DAMGARTEN-POL	*TRUE	ENEMY	AIRBASE	GDR	nil
DEMMIN	53.93	13.23	DEMMIN-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
DESSAU	51.81	12.16	DESSAU-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
DRESDEN	51.1	13.78	DRESDEN-POL	*TRUE	ENEMY	AIRBASE	GDR	nil
DREWITZ	51.9	14.51	DREWITZ-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
ERFURT	50.96	10.96	ERFURT-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
FALKENBERG	51.53	13.21	FALKENBERG-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
FINOW	52.81	13.7	FINOW-POL	*TRUE	ENEMY	AIRBASE	GDR	nil
FINSTERWALDE	51.6	13.71	FINSTERWALDE-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
FRIEDRICHOF	52.31	13.85	FRIEDRICHOF-POL	*TRUE	ENEMY	AIRSTRIP	GDR	nil
FURSTENWALDE	52.38	14.08	FURSTENWALDE-POL	*TRUE	ENEMY	AIRSTRIP	GDR	nil
GALLSCHUTZ	51.18	12.98	GALLSCHUTZ-POL	*TRUE	ENEMY	AIRSTRIP	GDR	nil
GARZ	53.88	14.15	GARZ-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
GROSSENHAIN	51.31	13.53	GROSSENHAIN-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
HAHN	49.93	7.26	HAHN-POL	*TRUE	FRIEND	AIRBASE	USA	910
HAINA	50.98	10.46	HAINA-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
JUTERBOG	51.98	12.98	JUTERBOG-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
KOTHEN	51.71	11.93	KOTHEN-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
LEIPZIG	51.35	12.36	LEIPZIG-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
LUCKAU	51.86	13.76	LUCKAU-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
MAHLWINKEL	52.38	11.85	MAHLWINKEL-POL	*TRUE	ENEMY	AIRFIELD	GDR	nil
MARXWALDE	52.78	14.26	MARXWALDE-POL	*TRUE	ENEMY	AIRBASE	GDR	nil

Figure 14-8: The Data Base Viewer Display



```
Case frames related to the node - DRESDEN:

The following is a table PART type case frames.

case frame node  super-part  part-name           part             description
 M874            DRESDEN     SA-6                B35
 M873            DRESDEN     SA-6                B34
 M872            DRESDEN     SA-6                B33
 M871            DRESDEN     SA-6                B32
 M870            DRESDEN     SA-6                B31
 M864            DRESDEN     RADAR               control radar
 M863            DRESDEN     RADAR               control radar
 M862            DRESDEN     RADAR               control radar
 M861            DRESDEN     RADAR               control radar
 M860            DRESDEN     RADAR               control radar
 M853            DRESDEN     RUNWAY              6-24-DRESDEN
 M852            DRESDEN     RUNWAY              3-21-DRESDEN
 M849            DRESDEN     fuel tank storage   DRESDEN-POL
The following is a table COMP type case frames.

case frame node  super-comp  comp-name           comp          description
 M2452           DRESDEN     POL-FACILITY-NAME   DRESDEN-POL   petroleum-oil-lubricants facility
The following is a table OPV type case frames.

case frame node  object      property                       value
 M456            DRESDEN     DISPOSITION                    ENEMY
 M440            DRESDEN     NATIONALITY                    GDR
 M400            DRESDEN     LOCATION                       a location : -> 51.1 N 13.7 E
 M341            DRESDEN     COLOR                          RED
 M321            DRESDEN     NAME                           DRESDEN
 M2457           DRESDEN     AFFILIATION-OF-AIR-FACILITY    GDR                                   the nation or c
 M2455           DRESDEN     TYPE                           airbase                               airfield, airba
 M2454           DRESDEN     OPERATIONAL                    *TRUE                                 *true if air-fa
 M2450           DRESDEN     LONGITUDE                      13.70                                 longitude of ai
 M2440           DRESDEN     LATITUDE                       51.1                                  latitude of air
Inherited case frames relate to the node - air base:


The following is a table DRAW type case frames.

case frame node  class       form
 M287            air base    BASE
The following is a table COMP type case frames
```

Figure 14-9: The KB Builder Display

mouse gestures are show in the *mouse documentation line* at the bottom of the display. Since different commands are available when the mouse is on different types of objects or commands the mouse documentation line changes dynamically as the mouse moves from object to object.

In a few cases pop up menus appear which request the user to type in names or numhers. Whenever possible user typing is supported by computer completion of the typed input and computer enumeration of the remaining input possibilities.

14.3 Supported Activities

The two main displays support a variety of activities, data base browsing, link generation, KB generation, and WYSIWYG editing. The details of how the tool supports these activities are discussed below.

14.3.1 Data Base Browsing Using the Data Base Viewer

The Data Base Viewer is intended to help the KB builder examine and become familar with the the AMPS database. It is easy to use and can display the entire contents of the AMPS RDB.

The data is displayed by table, the central concept in a RDBMS. The user can display the list of all tables by clicking on the **show tables** command in the command menu at the top of the display. This results in the display of all of the table names in the window pane underneath the command menu. Each table name is sensitive to mouse gestures and can be used to generate displays relating to that table. Normally the display of the table names is done just once and the table names are referred to through out the session. If new AMPS tables were created during a session then it would be necessary to redisplay the table names.

When the user points at a table name the mouse documentation line shows that, the table can be described, have its attributes listed, or be displayed.

> Choosing the description pops up a window with a small bit of english text describing the table.
>
> Choosing the attribute display causes the list of attributes for that table to be displayed in the lower window. (Descriptions of the attributes can be popped up via a mouse gesture where ever they appear.)
>
> Choosing the full display causes the entire contents of that table to be formatted and presented in the lower window of the display. The table is formatted in a tabular

format with the table name preceeding the display and with the attribute list in the headings.

The lower box of the display contains both vertical and horizontal scroll bars. These are necessary since many of the tables take up more space than is available on the screen. The use of these scroll bar is interactively documented in the mouse documentation line. Every presentation made in the lower box is recorded and kept until the user invokes the **clear** command. Previous displays can be reviewed by simply scrolling the display forward or back to where the information was displayed. The mouse sensitivity of the screen display is continually preserved during scrolling operations.

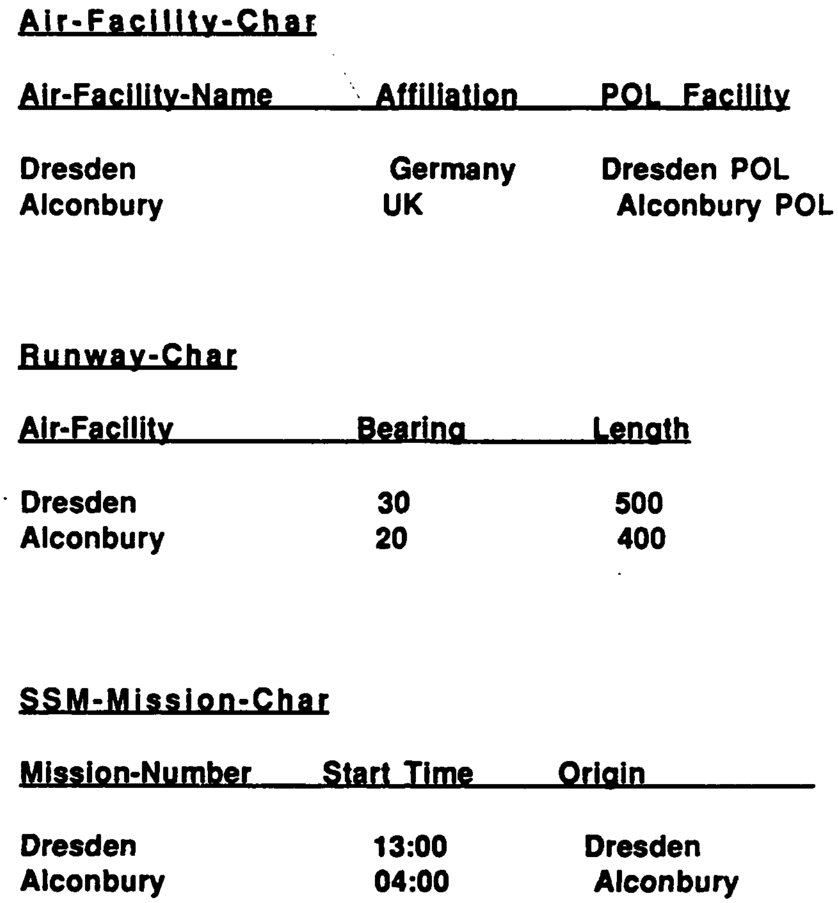
14.3.2 The Generation of Links Between AMPS and SNePS

One of the primary uses of the KB Builder tool is to develop a mapping between the AMPS DB and CUBRICON's SNePS KB which is based on the definition of links (cf. Section 14.1.1). The KB Builder supports the interactive semi-automatic generation of these links based on information in the RDB. The tool performs a heuristic search of the RDB to identify information that might be used to build links. During the search the tool will interactively request information from the user to help it limit its search. When the link information has been collected the tool displays a list of candidate links for the user to select from. Selected links are then automatically added to the KB.

The semi-automatic definition of the links is done in the context of a given class node. The search proceeds in several stages,

- A list of RDB tables is found that may have information in them pertaining to members of the given class.
- These tables together with associated candidate key attributes are displayed and the user selects table/key-attribute combinations to be considered .
- For each table/key-attribute considered, candidate sought attributes are displayed and selected from.
- the user then selects the case frame type to be used to relate the link node with the class node.
- For each collection of link information identified a link is built and placed in the KB.

The tool uses the class node's name as the starting point for its heuristic search. This name is used to identify tables that may contain information relating to members of the given class node. This is done by searching the list of all tables in the RDB for those tables which



Air-Facility-Char

Air-Facility-Name	Affiliation	POL Facility
Dresden	Germany	Dresden POL
Alconbury	UK	Alconbury POL

Runway-Char

Air-Facility	Bearing	Length
Dresden	30	500
Alconbury	20	400

SSM-Mission-Char

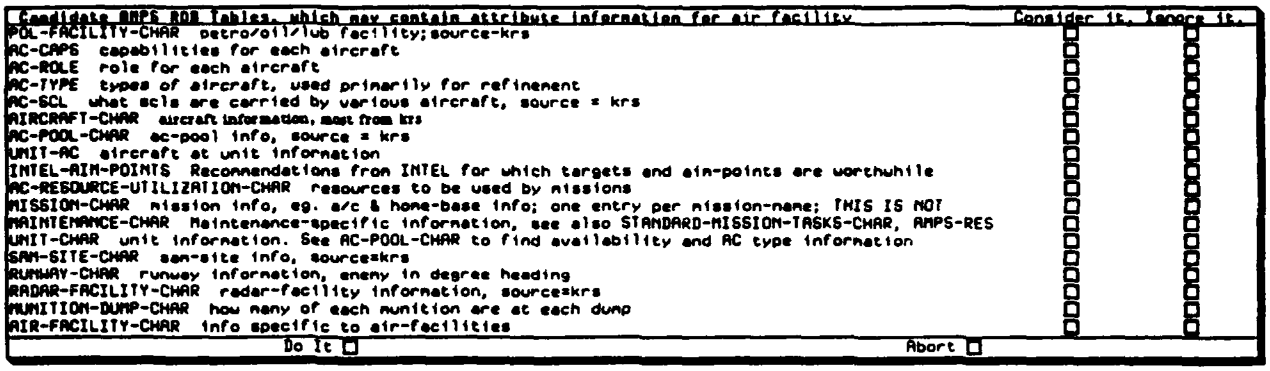
Mission-Number	Start Time	Origin
Dresden	13:00	Dresden
Alconbury	04:00	Alconbury

Figure 14-10: Attribute Based Table Selection

have attributes whose name is "like" the name of the given class. This process identifies candidate table/attribute pairs. The attribute identified is a candidate for the key attribute to be used in the link.

An alternative approach would be to look for RDB tables with names that are "like" the given class name. This approach was tried and found to be inferior for two reasons. It does not identify key attributes which are required and it overlooked tables with pertinent information.

Figure 14-10 illustrates the selected approach. The given class node in this case is air base. An examination of the Figure reveals that the runway-char table contains an attribute called air-facility and the air-facility-char table contains the attribute air-facility-name. The tool would select these two tables because the names air-facility and air-facility-name are "like" the name air-base. The table SSM-mission-char is not selected since none of its attributes are "like" the name air-base. Note that the table runway-char would not have been selected if the alternative approach were used. The tables and key attributes that are selected by the tool are then displayed in a window (cf. Figure 14-11) so that the user can make a further



Candidate AMPS RDB Tables, which may contain attribute information for air facility	Consider it.	Ignore it.
POL-FACILITY-CHAR petro/oil/lub facility;source-krs	□	□
AC-CAPS capabilities for each aircraft	□	□
AC-ROLE role for each aircraft	□	□
AC-TYPE types of aircraft, used primarily for refinement	□	□
AC-SCL what scls are carried by various aircraft, source = krs	□	□
AIRCRAFT-CHAR aircraft information, most from krs	□	□
AC-POOL-CHAR ac-pool info, source = krs	□	□
UNIT-AC aircraft at unit information	□	□
INTEL-AIM-POINTS Recommendations from INTEL for which targets and aim-points are worthwhile	□	□
AC-RESOURCE-UTILIZATION-CHAR resources to be used by missions	□	□
MISSION-CHAR mission info, eg. a/c & home-base info; one entry per mission-name; THIS IS NOT	□	□
MAINTENANCE-CHAR Maintenance-specific information, see also STANDARD-MISSION-TASKS-CHAR, AMPS-RES	□	□
UNIT-CHAR unit information. See AC-POOL-CHAR to find availability and AC type information	□	□
SAM-SITE-CHAR sam-site info, source=krs	□	□
RUNWAY-CHAR runway information, enemy in degree heading	□	□
RADAR-FACILITY-CHAR radar-facility information, source=krs	□	□
MUNITION-DUMP-CHAR how many of each munition are at each dump	□	□
AIR-FACILITY-CHAR info specific to air-facilities	□	□

Figure 14-11: Table and Key Attribute Selection

selection from them.

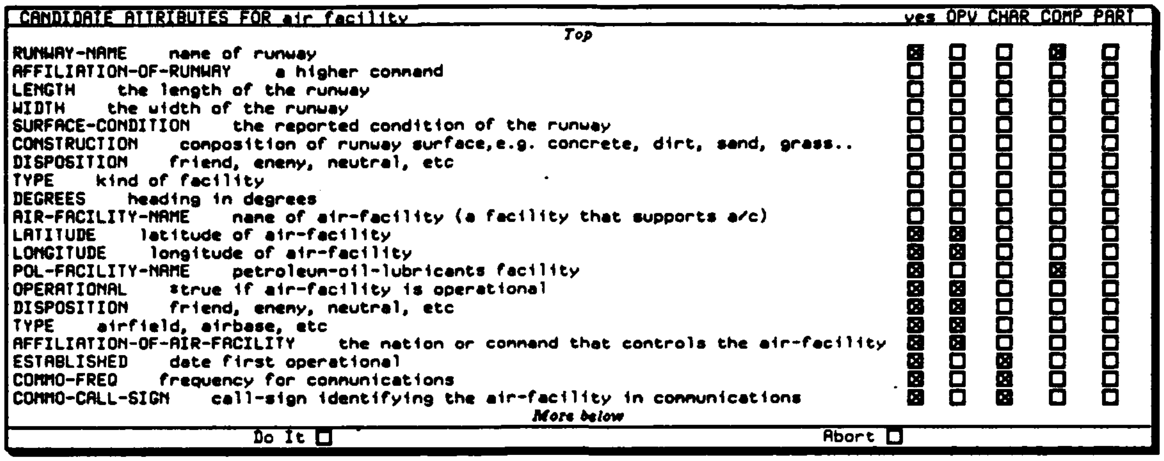
In order for the tool to work it must be able to identify when one name is "like" another. This process is based on a synonym table which the user can interactively edit. This table might express the notion that base and facility are synonymous, or air-base could be associated with air-facility directly. In the process of finding like words the words are de-hyphenated into a list of component words. These component words are augmented by synonyms from the synonym table and all the resulting combinations are compared to see if they are contained in the attribute names of the tables.

Once the table/key-attribute pairs have been selected, the selected tables are used to generate a list of candidate sought attributes. These candidate sought attributes are presented in a display (cf. Figure 14-12) along with the associated table and key attribute. The user selects from this display the collections of table, key attribute, and sought attribute that are to be used to build a link. The type of case frame that will be used to relate the class node and the link is also selected through this display.

For each selected combination of table, key attribute, sought attribute, case frame type, a link is built and related to the given class node. It is assumed that instance names will be used as a key values.

14.3.3 KB Generation

The system enables the selective construction, in a controlled fashion, of large KB's based on information from the AMPS DB. This is an important capability since attempts to map



Top

CANDIDATE ATTRIBUTES FOR air facility	yes	OPV	CHAR	COMP	PART
RUNWAY-NAME name of runway	☒	□	□	☒	□
AFFILIATION-OF-RUNWAY a higher command	□	□	□	□	□
LENGTH the length of the runway	□	□	□	□	□
WIDTH the width of the runway	□	□	□	□	□
SURFACE-CONDITION the reported condition of the runway	□	□	□	□	□
CONSTRUCTION composition of runway surface,e.g. concrete, dirt, sand, grass..	□	□	□	□	□
DISPOSITION friend, enemy, neutral, etc	□	□	□	□	□
TYPE kind of facility	□	□	□	□	□
DEGREES heading in degrees	□	□	□	□	□
AIR-FACILITY-NAME name of air-facility (a facility that supports a/c)	□	□	□	□	□
LATITUDE latitude of air-facility	☒	☒	□	□	□
LONGITUDE longitude of air-facility	☒	☒	□	□	□
POL-FACILITY-NAME petroleum-oil-lubricants facility	☒	□	□	☒	□
OPERATIONAL *true if air-facility is operational	☒	☒	□	□	□
DISPOSITION friend, enemy, neutral, etc	☒	☒	□	□	□
TYPE airfield, airbase, etc	☒	☒	□	□	□
AFFILIATION-OF-AIR-FACILITY the nation or command that controls the air-facility	☒	☒	□	□	□
ESTABLISHED date first operational	☒	□	☒	□	□
COMMO-FREQ frequency for communications	☒	□	☒	□	□
COMMO-CALL-SIGN call-sign identifying the air-facility in communications	☒	□	☒	□	□

More below

Do It □ Abort □

Figure 14-12: Final Selection of Link Related Information

the whole AMPS DB into SNePS resulted in KB's whose size taxed the capabilities of the computer systems used.

The system provides two basic KB generation capabilities, 1) the generation of instance nodes representing members of a given class, and 2) the generation of KB structures (i.e. case frames) which relate information to the class nodes.

14.3.3.1 Knowledge Base Instance Generation The user of the KB Builder tool first uses the tool's KB editing capabilities to build a skeletal class structure in the KB. Ultimately this skeletal structure must be populated with instances of the classes. For example, the node representing the class air base must be related to an instance node representing Dresden air base. The KB builder tool supports the semi-automatic generation of nodes representing such instances from the RDB information.

For a given class node the tool identifies the names of instances of that class using techniques like those us৭d to identify link information. The tool first uses the name of the given class to find tables with attribute names "like" the name of the given class node. The table and attribute pairs are displayed and the user of the tool selects one such pair.

The attribute of the table/attribute pair selected by the user references a column of the selected table. This column contains the names that are to be used as instances for the given class. For example the air-facility-char table may have an attribute called air-facility name. In the column under this attribute are all of the names of airbases contained in the RDB. This set of names are used as the names of the instances of the air base class and are used by the tool to build instance nodes.

14.3.3.2 Case Frame Generation Once the links and instance nodes have been created for a given class node, the links are used to build case frames that relate RDB information to the instances. The procedure for using a link to relate RDB information to an instance node was discussed in Section 2.1.

We note that if an instance has super classes in addition to its class then it may inherit links from the super classes as well as from the class. These links work in the same way as those associated with the class.

14.3.4 WYSIWYG Editing Capabilities

The Knowledge Base Builder provides "what you see is what you get" (WYSIWYG) editing capabilities through the use of the Knowledge Base Builder window shown in Figure 14-9. This display has two windows, the one on the left is the AKO tree display and the one on the right is the case frame information display. This later display presents information related to a given class or instance node through molecular nodes of known case frame types.

14.3.4.1 KB Displays These displays hide many of the SNePS representation details from the viewer so that the information which is pertinent to the user's task is emphasized. As an example consider molecular node M400 in the display in the right window of Figure 14-9. This node is found in the table of OPV type case frames and gives Dresden's location. Figure 14-13. shows a diagram which includes the SnePS network corresponding to this line. The design of the SNePS representations used by the case frame types is a separate concern and would be handled by a different display.

When the KB Builder displays a node it attempts to present that node by printing,

- it's name if the node is a named base node.
- the name of the nodes lexeme if it is a lexed node.
- a standard case frame presentation if the node is a case frame of a known type.

Base nodes are given names in the KB through the use of the namer case frame. If a base node is connected to such a case frame then is is said to be a named node and the named is used as the printed representation of the node. If a base node is not named then the internal name of the node itself is used. An example of this is the node B35 in Figure 14-9. It represents an unnamed SA-6 which is part of Dresden.

Case frames very rarely refer to a name directly. Instead they often point to a molecular node which represents the concept of the name and that node uses a lex arc to point to a base node which contains the name. Such molecular nodes are referred to as lexed nodes

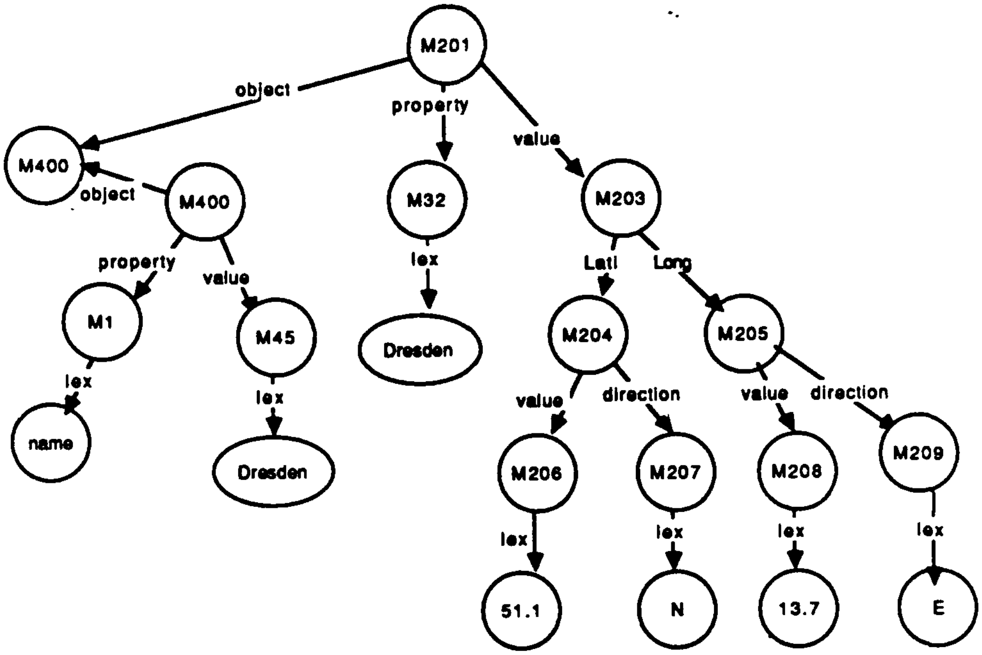


Figure 14-13: SNePS Structures Corresponding to KB Builder Display Items

and the corresponding base node as the lexeme. When such nodes are encountered the name contained in the lexeme is used as the printed representation of the node.

When compound case frames are encountered, such as is the case with node M400 of Figure 14-9, a standard representation is used for the subordinate case frame (the location case frame in this example). A printer function which is stored in the definition of the case frame type is used to print a representation of the subordinate case frame. This function may in turn recursively call other case frame printer functions to display an case frames that are subordinate to it.

The left window (cf. Figure 14-9) displays the skeletal class structure of the KB which is also called it's "A Kind Of" (AKO) hierarchy. This display also includes presentations of the instance node as they relate to their classes. In the figure the air base class node and its instances are shown. This display hides the details of how the relationship between the instances and classes and the relationship between the classes and their super classes are implemented.

The user can controls the extent of the graph in this display by,

- selecting which class or instance node is to be the root of the display tree,
- the depth of the display tree,
- the orientation of the display tree (horizontal or vertical), and
- the size of the window that the AKO trees are displayed in.

For example the user can select a new display root by pointing to the "Find From Name" command at the top of the display. This results in a popup window to which the user can type node names supported by completion and enumeration of acceptable input. When a new node is selected a redisplay is performed with the superclass of the air base node (e.g. the facility node) as the new root of the display.

There are several ways that a new AKO tree root can be specified. The "Find From Name", and "Find Node", commands can be used from the command menu. Also the user can point to any base node on the display including the right hand display and request that that node or one of it's superior nodes be the root of the AKO tree display. This combination of capabilities enables the user to browse the KB and to examine its AKO structure.

The system is currently limited in its ability to find and refer to unnamed nodes since most of the information in the RDB is referred to by named entities. The "Find Node" command provides a limited means to identify such unnamed nodes. If the node is found in the context of some other named node then it can be easily examined (e.g. node B35).

To display tables of case frame information for a base node the user points to the desired node in either the left or right window invokes a function which redraws the right hand display for the selected node. The right hand display of Figure 14-9 was produced by pointing to the Dresden node in the AKO tree display and selecting it to form the basis of the case frame display. Similarly any other base node could be selected to form a new display. For example in Figure 14-9 information relating to the unnamed SA-6, B35, can be displayed simply by pointing to it and selecting it.

The case frame display in the right window provides a list of all of the case frames that relate to the given node or to classes and super classes of the given node. The related case frames are grouped by case frame type and by the node they relate to. In the case frame display of Figure 14-9 three tables of case frame information are fully visible. For the node Dresden there are two tables, a table of PART case frames, and a table of OPV case frames. For Dresden's class node, air base, a table of DRAW case frames is visible.

Each case frame table is preceeded by a comment identifying the type of case frame contained in the table. The tables themselves contain a header line which is underlined. The rows under the header present information regarding the case frames.

The first item in the header is always "case frame node" and the entries in the column under this node identify the exact molecular node used in the case frame. These nodes are presented in the table so that the user has a displayed object that refers to the case frame as a whole. This object is referred to when the user wishes to invoke commands that operate on on the case frame as a whole. An example would be the cut and paste operations which enable the user to move case frames from one base node to another.

The rest of the items in the header display the names of the arcs used in case frames of the type that the table represents. In Figure 14-9 we see that the PART case frame utilizes the arcs super-part, part-name, part and description. From the same figure the OPV case frame can be seen to involve the object, property, and value arcs.

The items in the rows beneath the arc names are representations of the nodes found at the end of that arc for the given case frame. For example the OPV case frame for M400 has a value arc which points to a location case frame. The printed representation of location case frames appears in the table.

14.3.4.2 Editing Capabilities The editing capabilities of the display are all provided in the context of pointing references to items in the display. Each type of display item has a different set of operations that are available for it. These operations are invoked by pushing various combinations of keys and buttons on the pointing device. Such combinations are referred to as "mouse gestures". The operations that are available on various mouse gestures are dynamically documented in a one line display at the bottom of the screen called the mouse documentation line. Pushing the middle mouse button for any type of display item will popup a menu of operations for that item.

The editing capabilities include the ability to,

- Change the value of any item on the screen.
- Edit the displays of case frame objects.
- Cut and past nodes related to molecular nodes.
- Add new case frames, and base nodes.
- Detach case frames and base nodes from related nodes.
- Reorganize the AKO tree.

We note that nodes which become detached and isolated during the editing process are deleted.

When editing case frame displays the system utilizes information stored in the definition of that case frames type to control the editing process. For example when editing a location case frame the user is prompted for the four pieces of information required, the numerical value of the latitude, either N or S, the numerical value of the longitude, and either W or E. When the user enters such values the tool will only accepts values in the correct range and will display the acceptable possibilities if the help button is pressed.

15 Evaluation

15.1 Overview of Approach

A thorough evaluation of CUBRICON was conducted on 3 October through 6 October, 1989. The purpose of the evaluation was to assess CUBRICON with respect to measures of human-computer interface effectiveness and efficiency. Feedback about the performance of CUBRICON from these perspectives is contained in this section. Recommendations for further research are also provided.

The evaluation was performed in two parts. The first part of the evaluation focussed on human engineering issues relevant to CUBRICON. Ms. Mary Lloyd, a human factors specialist from Calspan Corporation, conducted this part of the evaluation. Ms. Lloyd has many years of experience in human factors engineering, including experience in the conduct of human engineering evaluations of prototype systems. Prior to the evaluation, she was unfamiliar with the CUBRICON Project and was therefore able to provide an independent and unbiased evaluation.

The second part of the evaluation was conducted to evaluate CUBRICON from an Air Force applications point of view. This part of the evaluation assessed the applicability of CUBRICON interface concepts to typical and emerging Air Force applications. Mr. Albert Frantz, an engineer from the Rome Air Development Center (RADC/COAD), was employed during this part of the evaluation. Mr. Frantz has experience in the development and management of Air Force C2 system development efforts, and therefore was able to represent the perspective of Air Force users within the present evaluation. He had no prior involvement with the CUBRICON Project, and therefore was also able to provide an independent and unbiased review.

The evaluation addressed the general goals of an intelligent human-computer interface which were outlined in the Statement of Work (SOW). Human factors issues that bear on these goals, as well as other human factors issues that relate to the CUBRICON design and future directions, were evaluated. The items on the human engineering evaluation questionnaire (see Appendix E) are cross-referenced with respect to these goals.

There are a number of issues which constrained the approaches available for the evaluation of CUBRICON. First, CUBRICON is a prototype system. Tasks to be tested had to be confined within current CUBRICON capabilities. For example, the evaluation had to be limited to the vocabulary and grammar supported by CUBRICON, and it had to employ discrete speech input. Also, CUBRICON is implemented on a Symbolics Lisp Machine which provided a very slow response time. This detracted from the "conversational feeling" that one gets from an interface of this nature. Subjects were aware of these limitations and tried to judge the merits of the underlying concepts in spite of implementation limitations.

Second, CUBRICON represents an exploration of new technology. It is a "one of a kind" system. It was not built as an improvement to a pre-existing system. It therefore was not possible to compare performance using CUBRICON to that using traditional technology.

Third, it was not possible to employ "real users" as test subjects. The ultimate CUBRICON applications are yet to be determined and military personnel serving in roles related to the hypothetical problems employed in the CUBRICON evaluation are not available. We did, as noted above, employ an Air Force systems development engineer who was very knowledgeable in tactical military planning tasks and with computer-based systems being developed for these tasks. We feel that the user perspective was well served by the Air Force Evaluator.

Finally, since this is a basic research effort, budgets were limited. We were only able to employ two subjects in the evaluations. While valuable insights have been drawn, there are certainly strengths and weaknesses that were not identified, and we have little information relating to how the range of individual differences will affect the effectiveness of CUBRICON design concepts.

Each of the two participants were able to use the CUBRICON voice recognition system directly, rather than working through an intermediary. While this was not originally planned, our experience in preparing for the evaluation led us to believe that working through an intermediary would have seriously hindered the evaluation process. Only with direct interaction could subjects fully experience CUBRICON and provide meaningful evaluation.

Also based on the test preparations, a *script-based evaluation* was added to the Air Force applications oriented part of the evaluation. This was in addition to the already planned problem solving task. By using the script which was developed for the human engineering part of the evaluation, we were able to ensure that the Air Force Evaluator would: (1) exercise and evaluate all important features of the CUBRICON system; and (2) Experience and evaluate a relatively error-free conversation with CUBRICON.

Other than these deviations, the CUBRICON evaluation was conducted according to the Test Plan delivered to DARPA and RADC in March, 1989 ("Test Plan/Procedures: Intelligent Multi-Media Interface Project", CLIN 002, ELIN A002). The evaluation attempted to identify those aspects of CUBRICON that worked well as well as those that did not work well, and to recommend enhancements and directions to guide future efforts. These evaluations were meant to be constructive. Any criticisms are not in any way intended to impugn the hard work and dedication that went into the CUBRICON development efforts. Much has been accomplished during this effort, and much remains to be done.

15.2 Procedures

As stated above, the CUBRICON evaluation proceeded in two stages. The first stage employed a human factors psychologist and focused on interface engineering issues. The second stage employed an Air Force engineer who was knowledgeable in computer-based tactical planning systems. This part of the evaluation focussed on the applicability of CUBRICON to military planning problems.

Each Evaluator received about five hours of training, including voice training and hands-on interactive practice. Each Evaluator was proficient with the procedures and techniques for using CUBRICON, before conducting their formal evaluations. The evaluations themselves were structured to the perspective of each particular Evaluator. Figures 15-1 and 15-2 show the schedule of the training and evaluations conducted.

15.2.1 Stage 1. Interface Engineering Evaluation

This stage of the CUBRICON evaluation proceeded in two steps. First, the human factors specialist interacted with CUBRICON by following a prepared script (included in Appendix E). This script was developed to exercise all important features of CUBRICON, especially as they relate to the evaluation criteria.

The second step in this stage of the evaluation involved free-form exploration of CUBRICON's capabilities. The Human Factors Evaluator was instructed to interact with CUBRICON in an open-ended fashion to: 1) more fully evaluate CUBRICON's performance vis-a-vis the evaluation criteria and 2) stress the system to find out where weaknesses exist. This part of the evaluation allowed the human factors specialist to tailor the interactions with CUBRICON to tease out data specifically addressing the evaluation criteria.

The Human Factors Evaluator conducted these evaluations with the aid of the engineering evaluation checklist. This checklist guided the evaluations. The evaluator was required to provide judgements about each evaluation item by checking the appropriate selection and noting the reasons behind the selection. The completed checklist is contained in Appendix E. At the conclusion of the evaluation session, the Human Factors Evaluator answered open ended questions that allowed more general impressions to be expressed. These questions included solicitation of suggestions for improving the CUBRICON user interface. This questionnaire with the evaluator's answers is also included in Appendix E. Results of the Human Factors evaluation are part of the evaluation summary of Section 15.3. Recommendations are presented in Section 15.6.

15.2.2 Stage 2. Air Force User Evaluation

This part of the evaluation also proceeded in two steps. First the Air Force Evaluator used the script described above to guide the evaluation. This was not originally planned but was added to allow the Evaluator to experience all important features of CUBRICON in a relatively error-free manner and hopefully gain a better sense of conversational flow.

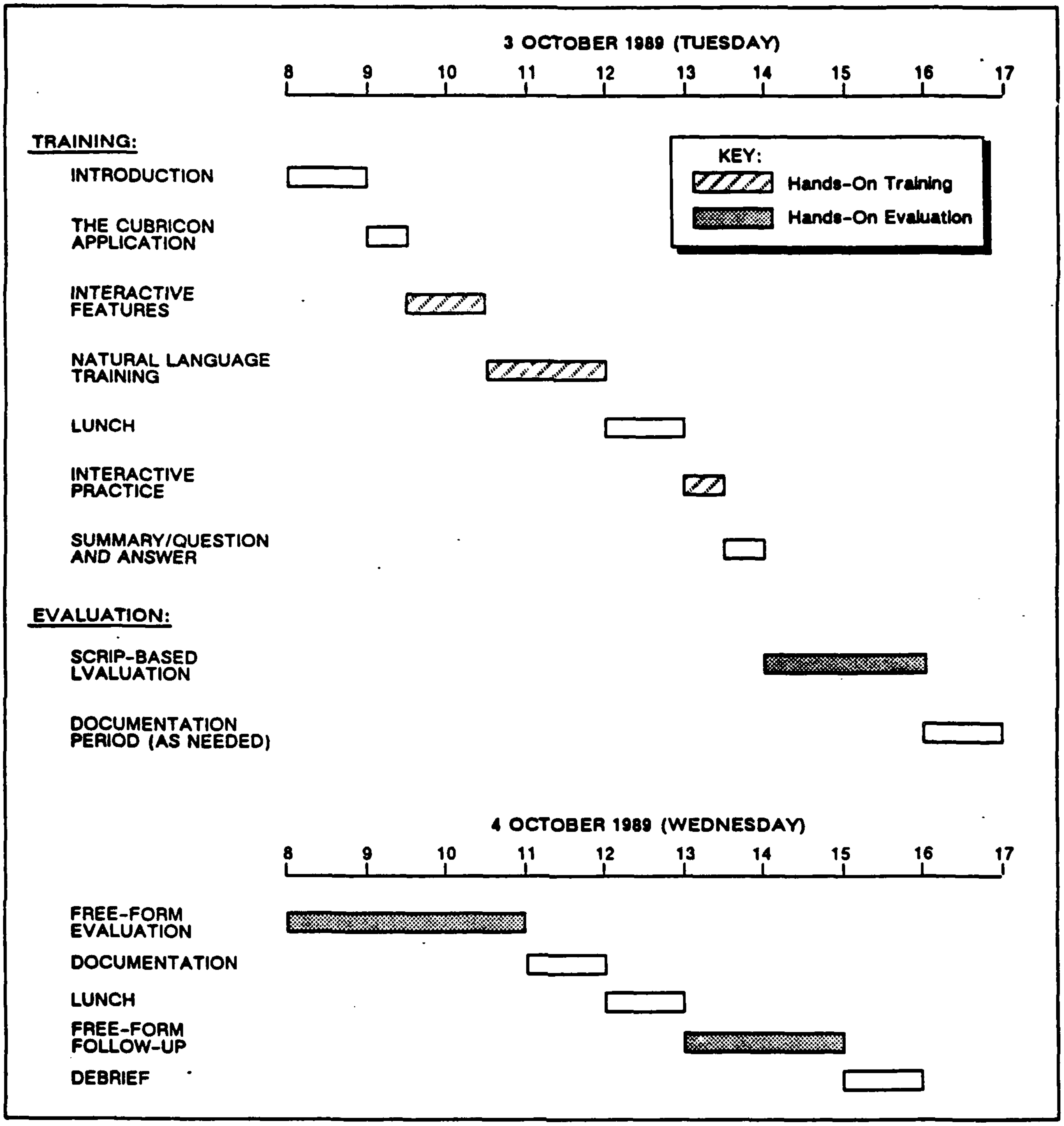


Figure 15-1: CUBRICON Interface Engineering Evaluation Schedule

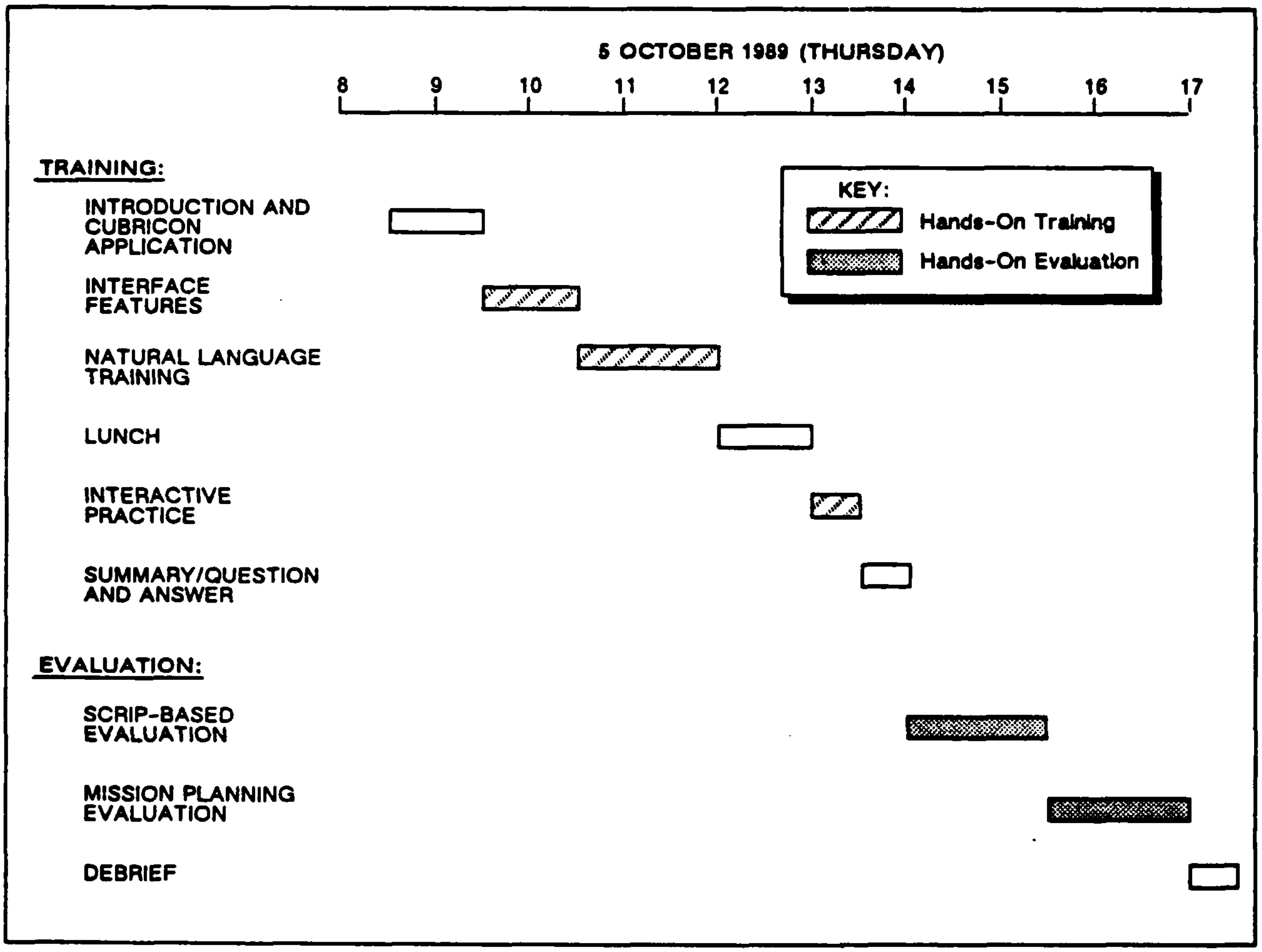


Figure 15-2: CUBRICON Air Force User Evaluation Schedule

This part of the evaluation was followed by a problem-solving task in which the evaluator was asked to use CUBRICON to solve typical military planning tasks. During this part of the evaluation the Evaluator was free to pursue the problem in any manner he thought appropriate. This part of the evaluation afforded more applied problem-solving experience.

The evaluations were accomplished through observation and subsequent debriefing of the Evaluator. These sessions were observed by the Test Conductor who recorded any difficulties experienced in using CUBRICON. The sessions were video taped and the Evaluator was debriefed with a questionnaire following the session. The completed questionnaire is contained in Appendix E.

Finally, the data from this stage of the evaluation was analyzed together with the results of the first stage of the evaluations. Conclusions about the overall system design and functionality were drawn. These are summarized in Section 15.3. Conclusions and recommendations drawn are presented in Section 15.6.

15.3 Summary of Results

The results of this evaluation provide insights into the strengths, weaknesses, and effectiveness of integrated multi-media, human-computer interfaces. Specific recommendations for the improvement of CUBRICON itself were also obtained. A discussion of these results is contained in *this section.*

The concept of an integrated multi-media human-computer interface in which users are able to interact with a computer system via a combination of speech input/output and direct graphic interactions was supported during this evaluation. Both evaluators found that the ability to perform map- and form-based mission planning activities by pointing at objects and describing desired actions verbally, was superior to more traditional typing-based approaches. Further, the ability to interact directly on numerous windows simultaneously was found to be advantageous when information of interest was displayed on more than one window.

The concept of a unified system in which various displays and presentations reflected a single integrated underlying reality was also supported. For example, the ability to manipulate objects on one display and view the effects of that manipulation on other displays, was judged to be an important goal of integrated multi-media interfaces. In fact, the Air Force Evaluator suggested that the CUBRICON system didn't carry this concept far enough. He suggested that CUBRICON should provide tools for real-time sensitivity analyses in which parameters defining a mission plan could be manipulated in one window and the results simultaneously presented in another window.

The Air Force Evaluator found the concept of automatic window management to have merit. The automatic removal of old windows was specifically noted by both Evaluators as having potential. The Human Factors Evaluator stated that windows were usually organized for efficient use. However, she did express a desire for more user control over windowing in general. Both Evaluators liked the concept of iconizing used windows to allow subsequent recall if desired. This window iconization and recall feature is only partially implemented in CUBRICON at this time. Both Evaluators recommended full implementation. The results of the evaluation certainly support further research into automatic window management but indicate a need for making available user control over windows as well.

Several criticisms of the CUBRICON interface were also obtained during the evaluation. These tended to deal with specific interface design issues rather than more general conceptual issues. Several recommendations for the implementation of intelligent integrated multi-media interfaces can be gleaned from these specific criticisms.

The CUBRICON implementation of speech input was criticized by both Evaluators for lack of robustness. The limited vocabulary and grammar available for speech input made the formulation of inputs difficult and unnatural. Difficulties arising from the limited vocabulary and grammar, were compounded by limitations of the inexpensive speech recognition system employed which sometimes had difficulties in recognizing speech inputs. Additionally, the CUBRICON vocabulary was defined very narrowly. Available terms were understood in a restrictive way; for example, it was possible to *display* a map but it was necessary to *present* a flight path. These two terms could not be interchanged even though it would have been natural to do so. A more robust speech and natural language input capability is needed to achieve a truly natural interface.

Another limitation of the *CUBRICON design noted by both Evaluators was that there* was an over-reliance on speech without the availability of non-speech-based shortcuts. For example, the process of selecting an object displayed on a map, for input on the mission planning form, required natural language, either typed or spoken. It was necessary to speak in complete sentences, such as, "put this (mouseclick) SAM, here (mouseclick)." A more efficient approach (i.e., quicker and less prone to mistakes) would be to simply grab the object with a mouseclick and put it where desired with a second mouseclick (perhaps with accompanying words grab and put). The abilit to point and talk is a major strength of CUBRICON. A next step is to add flexibility ana allow for operational shortcuts to improve interface efficiency.

A criticism made by the Human Factors Evaluator was that CUBRICON did not provide sufficient user assistance in the way of menus, prompts, or similar types of guidance. Even the mission planning form, which by its nature provides prompts as to the information to enter on the form, did not distinguish between required and supplemental entries. It is difficult for novice users to know what the system is capable of doing, and how to undertake relatively complex tasks, without guidance. This was especially troublesome for tasks that required well defined and rather rigid procedures. More explicit user guidance should be available on CUBRICON-type interfaces to application systems. Of course, these features are very application dependant and were not the focus of this research. These features would be incorporated in a final implementation.

Error management is another area in which CUBRICON received criticism from the Human Factors Evaluator. Too often an error in entering information or requests led to the message, "sorry, do not understand request, please try again." The user, in this situation, is given no information that would help in reformulating the input. The specific aspect of the request that was not understood should be identified for the user, and provisions for correcting the misunderstood part of the input should be available, rather than requiring the user to repeat the entire input[1].

The evaluation also identified a need for more user control. This was reflected by comments from both Evaluators, but was primarily a concern of the Human Factors Evaluator. While CUBRICON attempted (with some sucess) to provide outputs that clearly provided the information desired and needed by the user, there were situations in which CUBRICON displays deviated from that which was actually desired by the user. This is inevitable. In these situations, there was little the user could do in directly affecting changes to the display format or content. For example, it is possible to zoom-in on any point on any map, but it was not possible to tell the system how much area should be included in the zoomed-in area and how much resolution to provide.

Sometimes CUBRICON provided too much information. This was noted by both Evaluators and is particularly evident in the generation of a table to supplement each map window displayed. When the monochrome display contains the mission planning form, all map windows and corresponding tables are placed on the color graphics display. Since only four windows will fit on the color graphics display, the creation of a table for each and every map quickly exceeds the display capability. Deletion of old windows and overlapping among displayed windows, was frequently necessary. There was often no obvious way to tell which table corresponded to which map. Suggestions were made by the Air Force Evaluator about how to present information in support of maps without requiring numerous tables.

15.4 Evaluation with Respect to SOW Goals

The Human Engineering Evaluation was developed to provide assessment of CUBRICON performance with respect to human-computer interface efficiency and especially with respect to the human-interface goals specified in the Statement of Work (SOW). As stated in the Evaluation Test Plan/Procedures, these goals are:

1. "... minimize the requirement for translation and reformulation of information on the part of the the human. The computer should accept information from the human in a

[1] An update to CUBRICON since the interface evaluation has incorporated provision for more specific feedback when inputs are not understood. This improved part of CUBRICON was not subjected to human factors evaluation, but is expected to make the process of error correction much easier.

form that is natural for the human" (SOW, p. 4).

2. "Formats should be flexible to conform to individual styles yet need to be unambiguous and usable by more than one human user" (SOW, p.4).
3. "The system should assist the user in accessing an appropriate amount of information that is relevant to his needs" (SOW, p. 4 and p. 7).
4. "Machine outputs should be organized in a way that the human can easily assimilate the information within the context of the task(s) being performed" (SOW, p.4).
5. "The context of all communication must be kept clear" (SOW, p.4).
6. Speech, natural language, and graphics must be integrated for both computer input and output (SOW, p. 2).
7. "...dynamically define how information will be presented and how human/computer dialogue can be adapted based on the context of the dialogue or the decisions being made" (SOW, p.4).
8. "...track the focus space of the human/computer discourse ..." and determine "the appropriate referent of definite references, particularly those definite references involving multi-media expressions" (SOW, p. 7).

The items on the human engineering evaluation questionnaire (see Appendix E) are cross-referenced with respect to these goals. The results of the evaluation organized according to SOW goals is available upon request from the authors.

15.5 Completed Evaluation Questionnaires and Checklists

The evaluation Questionnaires and Checklists that were completed by the Human Factors Evaluator and the Air Force User Evaluator are included in Appendix E. This appendix contains

- the entire Interface Engineering Evaluation Checklist and Questionnaire as completed by the Human Factors Evaluator as part of the Human Engineering Evaluation. The Checklist and Questionnaire were intended to elicit detailed information from the Evaluator about how well the Evaluator believed CUBRICON performed with respect to human factors considerations and to solicit suggestions for improvement.
- the entire Application Task-Oriented Evaluation Questionnaire as completed by the Air Force User Evaluator. This Questionnaire was intended to elicit the Evaluator's judgements about how well CUBRICON performed with respect to Air Force user considerations and to solicit suggestions for improvements.

15.6 Conclusions and Recommendations

Briefly, the following conclusions and recommendations can be drawn form this evaluation:

- Continue research in intelligent, integrated multi-media interfaces (great potential).
- Speech/vocabulary must be robust. It must capture the fluid and variable form and style in which language is organized and used to express ideas and information.
- Provide for user control. Automatic interface management offers good potential but users must be able to step in and exert control and authority over it when needed.
- Incorporate demonstrated human-computer interface (HCI) technology to supplement new integrated multi-media technology (don't throw out the baby with the bathwater). For example, human-like natural language I/O should be combined with proven techniques to best harness the full potential and power of the computer.
- Continue the development of the CUBRICON system. CUBRICON offers potential as a research testbed and should lead to an interface system that can serve as an interface to complex operational application systems.
- Perform research to better understand how to apply intelligent integrated human-computer interface technology for improved system effectiveness.
- Need faster computer (than the computer used for CUBRICON) to realize ultimate potential of human-like human-computer interfaces combining sophisticated graphics and natural language/speech I/O. Recognition of speech inputs and generation of system outputs must be as fast, or nearly as fast, as human-to-human dialogue.

16 Future Directions

The previous section on Evaluation discussed many of the recommendations and suggestions that were made by the Evaluators. In addition to these recommendations and suggestions, there are numerous significant areas and ideas to be pursued and developed to advance this research. In this section, we focus on just a few of them.

The concept of a knowledge-based multi-modal system, for which intelligent multi-modal I/O processing is part of its task domain, should be investigated and developed. Such a multi-modal system has great potential in two respects: (1) such a system could be used to develop other multi-modal human-computer interface systems (just as some User Interface Management Systems are used to develop user interfaces) with the benefit of using the natural and efficient intelligent multi-modal interaction capability to accomplish the development task quickly and efficiently and (2) when such a system is used to interface to an application system, it should have the ability to be modified or tailored to meet the needs of individual users via dialogue during normal interactive sessions. That is, a user should be able to express his preferences or dislikes with regard to the behavior of the multi-modal interface and have the system understand his interface-relevant inputs and modify its behavior accordingly. For example, a user might want to tell the system to always put up a table of information about any SAM systems that can affect the flight path of an aircraft penetrating enemy territory whenever it puts up a map showing the flight path. Or a user might want to conditionally change the colors that the system uses to present geographic information. Or a user might want to restrict the system's use of speech output (he may find it annoying or he may be hard-of-hearing) and be able to specify the limited conditions under which speech output should be used. The specifications and rules that any individual user might input to the system to tailor its behavior could be saved by the system in a model of the specific user and recalled whenever he logs on to the system for its use. Thus, a knowledge-based multi-modal system that includes multi-modal I/O processing as its task domain could be used with great benefit in either capacity: used to develop new multi-modal human-computer interface systems or used to provide users with the ability to tailor their specific interface system to meet their needs.

The issue of integrating continuous speech into a multi-modal human-computer interface system such as CUBRICON should be investigated and such a capability developed. CUBRICON currently accepts discrete speech input, but we anticipate that continuous speech will provide a much more natural form of input for the user in combination with deictic gestures and graphic expressions. Since users must speak deliberately with a discrete speech recognition system, they tend to point carefully also. We anticipate that with more natural continuous speech, users may tend to be more careless with pointing and other graphic expressions. Users may not synchronize their pointing gestures or drawing expressions with their corresponding natural language phrases. Characteristics of the problem that need to be investigated are: To what degree is there a lack of synchronization between gestures of different types and their corresponding natural language phrases? How frequently does the phenomenon occur? Is there a correlation between characteristics of the phenomenon and characteristics of the corresponding natural language? In addition to this research, methods need to be developed that would enable a human-computer interface system to decide which phrase of the accompanying natural language input is the co-referring phrase for any pointing gesture or drawing expression that is not synchronized with its co-referring phrase.

CUBRICON and its underlying technology should be extended so that the system has the ability to handle a greater variety of gesture types and drawing expressions as input from the user, both in isolation as well as in combination with natural language. Such an extended verbal/graphic input language can be used for both referencing objects that the system already knows about as well as explaining and defining new concepts to the system. This form of input would be especially useful for the definition and explanation of geographical and spatial concepts to a system that would then use the concepts for knowledge-based decision-making applications. Military domains abound in applications of this form of extended multi-modal input capability. Aircraft flight paths, regions, planned troop movements, and resource placements are just a few of the areas in which this form of input capability would be useful.

17 Summary

This report has discussed the results of the Intelligent Multi-Media Interfaces (IMMI) project. This research effort was motivated by the need for more effective human-computer interface technology. The IMMI project has been devoted to the application of artificial intelligence technology to the development of human-computer interface technology that integrates speech input, speech output, natural language text, maps, tables, forms, graphics, and deictic gestures for interactive dialogues between human and computer. As part of this project, a proof-of-concept human-computer interface system, called CUBRICON, has been developed. The application domain used to drive the research and development of CUBRICON has been that of Air Force tactical air control and mission planning.

The IMMI project has focused on the development of an interface system that is capable of intelligent and highly integrated use of multiple modalities for human-computer dialogues. The CUBRICON input modalities include speech, pointing gestures, form-based input, and typed text. Output modalities include speech, maps, graphics, pointing/highlighting, forms, tables, and typed text.

CUBRICON has several unique features. CUBRICON temporally coordinates input from different devices, such as the keyboard, speech recognition system, and pointing device. CUBRICON accepts natural language combined with coordinated simultaneous pointing gestures. CUBRICON allows a wide variety of object types to be the targets of point gestures, accepts varying numbers of point gestures within a phrase, and accepts varying numbers of multi-modal phrases within sentences. CUBRICON handles certain types of ill-formed multi-modal inputs: (1) CUBRICON can use natural language input to disambiguate accompanying pointing gestures and vice versa and (2) CUBRICON can infer the intended referent of certain types of multi-modal phrases in which the natural language is inconsistent with the accompanying pointing gestures.

CUBRICON provides for intelligent and automatic management of windows. This includes: (1) A method for determining window importance (used for deciding which windows to remove when display space is needed for other windows); and (2) a procedure for automatically managing windows in a dual monitor environment. This procedure considers window importance and type.

CUBRICON generates multi-modal outputs in a manner that enhances understandability. Specific features are: (1) Relevant information is selected by CUBRICON for presentation to the user. Relevant information is that which is specifically requested by the user, relevant to the dialogue, relevant to the user's task, and helps maintain consistency between related displays. (2) Modality selection is based on the characteristics of the information to be expressed vis-a-vis human sensing and understanding capabilities, as well as task and dialog context. (3) Multiple modalities are combined to: i) take best advantage of the relative strengths of each; ii) add emphasis or orientation to accompanying modalities; and iii) provide redundancy to ensure understanding and notice of important information. (4) All multi-media outputs are temporally synchronized (e.g., highlighting of graphics is temporally coordinated with related speech). (5) Spoken and written natural language outputs are designed for short-term and long-term reference, respectively. For example, written outputs include specific object referencing while a spoken reference can consist of one or more point gestures and a simple demonstative pronoun. (6) System outputs maintain format consistency within and across displays, and also provide for contextual orientation across all displays throughout the user-computer dialogue.

CUBRICON is a knowledge-based system. Input understanding and output composition considers dialog context (i.e., what is currently being displayed and has recently been expressed), task context (i.e., the importance of information relative to the ongoing task), and information context (i.e., the nature of the information vis-a-vis human sensing and understanding capabilities).

An evaluation of the CUBRICON system was conducted at the end of the project. This evaluation was conducted according to the Test Plan/Procedures submitted to the project sponsor and monitor eight months prior to the end of the contract. Many recommendations and suggestions were made for the future advancement this research area and the CUBRICON system. The following are just a few of the conclusions of the evaluators: (1) The concept of an integrated multi-media human-computer interface, such as CUBRICON, in which users are able to interact with a computer system via a combination of speech input/output and direct graphic interactions was supported by the evaluators. (2) The evaluators supported the concept of a unified system, such as CUBRICON, in which various displays and presentations reflected a single integrated underlying reality. (3) The concept of automatic window management, a prototype of which was included in the CUBRICON system, was found to have merit and be worthy of continued research and development.

The IMMI project has produced significant new knowledge-based multi-modal human-computer interface technology. This technology has significant potential for solving some of the problems posed by the need for military decision-makers to deal with increasingly large amounts of information and the increasing sophistication of military information processing systems.

18 References

[**Aceves84**] Aceves, J.J.G., Poggio, A. & Eliott, D. 1984. Research into multimedia message system architecture, Final Report. *Proj. No. 5363*, SRL International, Menlo Park, CA.

[**Andriole86**] Andriole, S.J. 1986. Graphic equivalence, graphic explanations, and embedded process modeling for enhanced process modeling for enhanced user-system interaction. *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. 16, No. 6.

[**Arens88**] Arens, Y., Miller, L., & Sondheimer, N. 1988. Presentation Planning Using an Integrated Knowledge Base, *Architectures for Intelligent Interfaces: Elements and Prototypes*, J.W. Sullivan & S.W. Tyler (eds.), Addison-Wesley Pub. Co., pp. 93-108.

[**Bates78**] Bates, M. 1978. The theory and practice of augmented transition network frammars, in *Lecture Notes in Computer Science, vol 63, Natural Language Communication with Computers*, G. Goos, J. Jartmanis, & L. Bolc (eds.), Springer-Verlag, Berlin etc.

[**Bly86**] Bly, S.A., & Rosenberg, J.K. 1986. A comparison of tiled and overlapping windows. in *CHI'86 Proceedings*, ACM 0-89791-180-6/86/04000101, pages 101-106.

[**Carberry87**] Carberry, S. 1987. First international workshop on user modeling, *AI Magazine*, Vol.8, No.3, pages 71-74.

[**Carbonell85**] Carbonell, J.G., Boggs, W.M., Mauldin, M.L., & Anick, P.G. 1985. An integrated natural language interface, in *Applications in Artificial Intelligence*, S. Andriole (ed.), Petrocelli Books, Inc., pages 227-245.

[**CACM88**] *Communications of the ACM: Special Issue on HyperText* 1988. Vol. 31, No. 7.

[**Cheikes88**] Cheikes, B.A. & Webber, B.L. 1988. The Design of a Cooperative Respondent, *Architectures for Intelligent Interfaces: Elements and Prototypes*, J.W. Sullivan & S.W. Tyler (eds.), Addison-Wesley Pub. Co., pp. 3-18.

[**Conklin87**] Conklin, J. 1987. Hypertext: An introduction and survey, *Computer*, Vol. 20, No. 9, pages 17-41.

[**Davies85**] Davies, S.E., Bury, K.F., & Darnell, M.J. 1985. An Experimental Comparison of a Windowed vs a Non-Windowed Operating System Environment, *Proceedings of the 29th Annual Meeting of the Human Factors Society*, pp. 250-254.

[**Feiner82**] Feiner, S., Nagy, S. & van Dam, A. 1982. An experimental system for creating & presenting interactive graphical documents, *Trans. Graphics*, Vol. 1, No. 1, pages 59-72.

[Grice75] Grice, H.P. 1975. Logic and conversation, in P. Cole & J.L. Morgan (eds.), *Syntax and Semantics, Vol. 3: Speech Acts*, Academic Press, pages 41-48.

[Grosz78] Grosz, B. J. 1978. Discourse analysis, in D. Walker (ed.), *Understanding Spoken Language*, Elsevier North-Holland, New York, pages 229-345.

[Grosz81] Grosz, B.J. 1981. Focusing and description in natural language dialogues, *Elements of Discourse Understanding*, A.Joshi, B. Webber, & I.Sag (eds.), Cambridge Univ. Press, pages 84-105.

[Grosz85] Grosz, B.J. & Sidner, C.L.. 1985. Discourse structure and the proper treatment of interruptions, *Proc. of IJCAI*, pages 832-839.

[Grosz86] Grosz, B.J. 1986. The representation and use of focus in a system for understanding dialogs," in *Readings in Natural Language Processing*, B.J. Grosz, K.S. Jones, B.L. Webber (eds.), Morgan Kaufmann Pulishers, pages 353-362.

[Haller] Haller, S.M. 1989. Technical report: spatial relations and locative phrase generation in a map context, Computer Science Department, State University of New York at Buffalo.

[Herkovits85] Herkovits, A. 1985. Semantics and pragmatics of locative expressions, in *Cognitive Science, vol 9*, pages 241-378.

[Hilton87] Hilton, M.L. 1987. Design and implementation of the MAP DISPLAY SYSTEM, RADC Report.

[Hollan88] Hollan, J., Miller, J.R., Rich, E., & Wilner, W. 1988. Knowledge Bases and Tools for Building Integrated Multimedia Intelligent Interfaces, *Architectures for Intelligent Interfaces: Elements and Prototypes*, J.W. Sullivan & S.W. Tyler (eds.), Addison-Wesley Pub. Co., pp. 19-38.

[Kass88] Kass, R. & Finin, T. 1988. General User Modelling: A Facility to Support Intelligent Interaction, *Architectures for Intelligent Interfaces: Elements and Prototypes*, J.W. Sullivan & S.W. Tyler (eds.), Addison-Wesley Pub. Co., pp. 169-184.

[Kaplan82] Kaplan, S.J. 1982. Cooperative Responses from a Portable Natural Language Database Query System, *Computational Models of Discourse*, M. Brady (ed.), The MIT Press.

[Katz84] Katz, A. 1984. An experimental internetwork multimedia mail system, *Proc. IFIP 6.5 Working Conf. Computer Message Services*, Nottingham, England.

[Kobsa86] Kobsa, A., Allgayer, J., Reddig, C., Reithinger, N., Schmauks, D., Harbusch, K., & Wahlster, W. 1986. Combining Deictic Gestures and Natural Language for

Referent Identification, *Proc. of the 11th International Conference on Computational Linguistics*, Bonn, FR Germany.

[Kobsa88] Kobsa, A. & Wahlster, W. (Editors) 1988. *Computational Linguistics: Special Issue on User Modeling*, Vol. 14, No. 3.

[Li87] Li, N. 1987. Pronoun Resolution in SNePS. SNeRG Technical Note No. 18. Department of Computer Science, SUNY at Buffalo.

[Maida85] Maida, A.S. & Shapiro, S.C. 1985. Intensional concepts in propositional semantic networks," in *Readings in Knowledge Representation*, R.J. Brachman & H.J. Levesque (eds.), Morgan Kaufmann Pub., page 169-190.

[Mason88] Mason, J.A. & Edwards, J.L. 1988. Explicit Models in Intelligent Interface Design, *Architectures for Intelligent Interfaces: Elements and Prototypes*, J.W. Sullivan & S.W. Tyler (eds.), Addison-Wesley Pub. Co., pp. 151-168.

[Martin73] Martin, J. *Design of man-computer dialogues.* Prentice Hall

[McKay80] McKay, D.P. & Shapiro, S.C. 1980. MULTI - A LISP based multiprocessing system, *Conference Record of the 1980 LISP Conference*, Stanford Univ., pages 29-37.

[Miller56] Miller, G.A. 1956. The magical number seven plus or minus two, *Psychological Review*, 63, pages 81-97.

[Neal86] Neal, J.G. & Shapiro, S.C. 1986. Knowledge representation for reasoning about language, in *The Role of Language in Problem Solving*, J.C. Boudreauz, B.W. Hamill, & R. Jernigan (eds.), Springer-Verlag Pub., pages 27-47.

[Neal87] Neal, J.G. & Shapiro, S.C. 1987. Knowledge based parsing, in *Natural Language Parsing Systems*, L. Bolc (ed.), Springer-Verlag Pub., pages 49-92.

[Neal88a] Neal, J.G., Bettinger, K.E., Byoun, J.S., Dobes, Z., & Thielman, C.Y. 1988. An Intelligent Multi-Media Human-Computer Dialogue System, *Proceedings of the Workshop on Space Operations, Automation, and Robotics (SOAR-88)*, Wright State University, Dayton, OH.

[Neal88b] Neal, J.G., Dobes, Z., Bettinger, K.E., & Byoun, J.S. 1988. Multi-Modal References in Human-Computer Dialogue, *Proc. AAAI-88*, pages 819-823.

[Neal88c] Neal, J.G. & Shapiro, S.C. 1988. Intelligent Multi-Media Interface Technology, *Architectures for Intelligent Interfaces: Elements and Prototypes*, J.W. Sullivan & S.W. Tyler (eds.), Addison-Wesley Pub. Co., pages 69-91.

[Neal89a] Neal, J.G., Thielman, C.Y., Funke, D.J., & Byoun, J.S. 1989. Multi-Modal Output Composition for Human-Computer Dialogues, *Proc. of the 1989 Artificial Intelligence Systems in Government Conference*, pages 250-257.

[Neal89b] Neal, J.G., Thielman, C.Y., Dobes, Z., Haller, S.M., & Shapiro, S.C. 1989. Natural Language with Integrated Deictic and Graphic Gestures. *Proc. of the 1989 DARPA Workshop on Speech and Natural Language*, Harwich Port, MA, Morgan Kaufmann Publishers.

[Neal89c] Neal, J.G., Thielman, C.Y., Dobes, Z., Haller, S.M., Glanowski, S., & Shapiro, S.C. 1989. CUBRICON: A Multi-Modal User Interface. *Proc. of the GIS/LIS '89 Conference*, Orlando, FL.

[Neches86] Neches, R. & Kaczmarek, T. 1986. *AAAI-86 Workshop on Intelligence in Interfaces*, USC/Information Sciences Institute, August, 1986.

[Potash77] Potash, L.M. 1977. Design of maps and map-related research, *Human Factors 19(2)*, pages 139-150.

[Press86] Press, B. 1986. The U.S. Air Force TEMPLAR project status and outlook, *Western Conf, on Knowledge-Based Engineering and Expert Systems*, Anaheim, CA, pages 42-48.

[Reithinger87] Reithinger, N. 1987. Generating Referring Expressions and Pointing Gestures, *Natural Language Generation*, G. Kempen (ed.), Dordrecht: Nijhoff, pp. 71-81.

[Roth88] Roth, S., Matthis, J., & Mesnard, X. 1988. Graphics and Natural Language as Components of Automated Explanation, *Architectures for Intelligent Interfaces: Elements and Prototypes*, J.W. Sullivan & S.W. Tyler (eds.), Addison-Wesley Pub. Co., pp. 109-128.

[Shapiro75] Shapiro, S.C. 1975. Generation as parsing from a network into a linear string, *AJCL*, Microfiche 33, pages 45-62.

[Shapiro79a] Shapiro, S.C. 1979a. The SNePS semantic network processing system, in Findler, ed., *Associative Networks - The Representation and Use of Knowledge by Computers*, Academic Press, New York, pages 179-203.

[Shapiro79b] Shapiro, S.C. 1979b. Using non-standard connectives & quantifiers for representing deduction rules in a semantic network, invited paper presented at *Current Aspects of AI Research*, a seminar held at the Electrotechnical Laboratory, Tokyo.

[Shapiro79c] Shapiro, S.C. 1979c. Numerical quantifiers and their use in reasoning with negative information, *Proc. IJCAI*, pages 791-796.

[Shapiro89] Shapiro, S.C. & the SNePS Implementation Group 1989. *SNePS-2.1 User's Manual*, C.S. Dept., SUNY at Buffalo, N.Y.

[Shapiro82a] Shapiro, S.C. 1982. Generalized augmented transition network grammars for generation from semantic networks, *AJCL*, Vol. 8, No. 1, pages 12-25.

[Shapiro82b] Shapiro, S.C., Martens, J., & McKay, D. 1982. Bi-directional inference, *Proc. of the Cognitive Science Society*, pages 90-93.

[Shapiro82c] Shapiro, S.C. & Neal, J.G. 1982. A knowledge engineering approach to natural language understanding, *Proc. ACL*, pages 136-144.

[Shapiro86] Shapiro, S.C. & Rapaport, W. 1986. SNePS considered as a fully intensional propositional semantic network, *Proc. AAAI-86*, pages 278-283; in G. McCalla & N. Cercone (eds.) *Knowledge Representation*, Springer-Verlag Pub.

[Shapiro87] Shapiro, S.C. & Rapaport, W. 1987. SNePS considered as a fully intensional propositional semantic network, in *The Knowledge Frontier, Essays in the Representation of Knowledge*, N. Cerone & G. McCalla (eds.), Springer-Verlag, pages 263-315.

[Sidner83] Sidner, C.L. 1983. Focusing in the comprehension of definite anaphora, in M. Brady & R.C. Berwick (eds.), *Computational Models of Discourse*, The MIT Press, pages 267-330.

[Simmons72] Simmons, R. & Slocum, J. 1972. Generating English discourse from semantic networks, *CACM*, 15:10, pages 891-905.

[Smith86] Smith, S.L. & Mosier, J.N. 1986. Guidelines for Designing User Interface Software, ESD-TR-86-278, August, 1986.

[Sullivan88] Sullivan, J.W. & Sherman, W.T. (eds.) 1988. *Architectures for Intelligent Interfaces: Elements and Prototypes*, Addison-Wesley Pub. Co.

[Thomas85] Thomas, R.H., Forsdick, H.C., Crowley, T.R., Schaaf, R.W., Tomlinson, R.S., Travers, V.M., & Robertson, G.G. 1985. Diamond: a multimedia message system built on a distributed architecture, *IEEE Computer*, December, pages 65-78.

[Wahlster88] Wahlster, W. 1988. User and Discourse Models for Multimodal Communication, *Architectures for Intelligent Interfaces: Elements and Prototypes*, J.W. Sullivan & S.W. Tyler (eds.), Addison-Wesley Pub. Co.

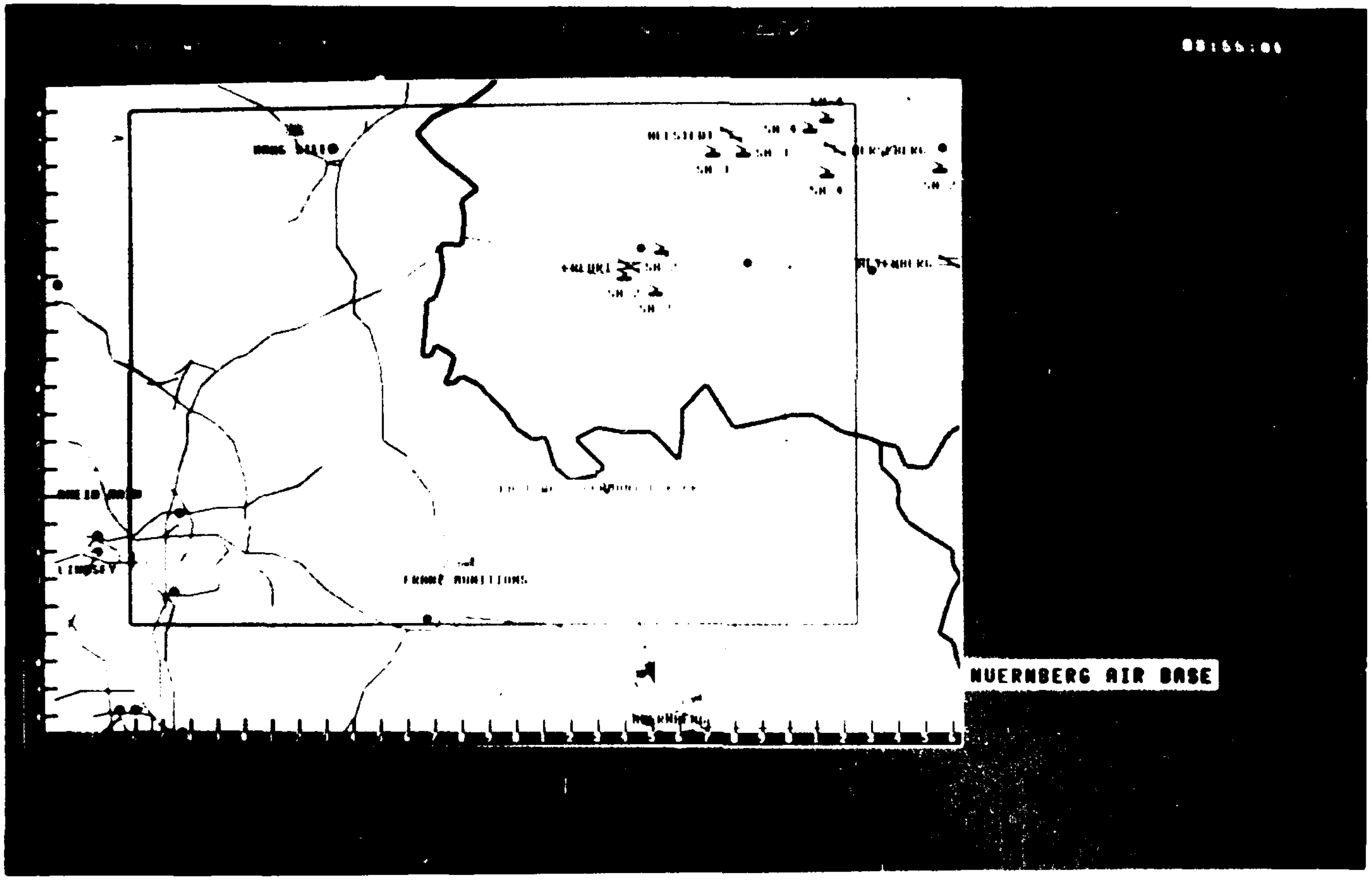
[Young88] Young, R.L. 1988. A Dialogue Model for User Interfaces, *Architectures for Intelligent Interfaces: Elements and Prototypes*, J.W. Sullivan & S.W. Tyler (eds.), Addison-Wesley Pub. Co., pp. 39-54.

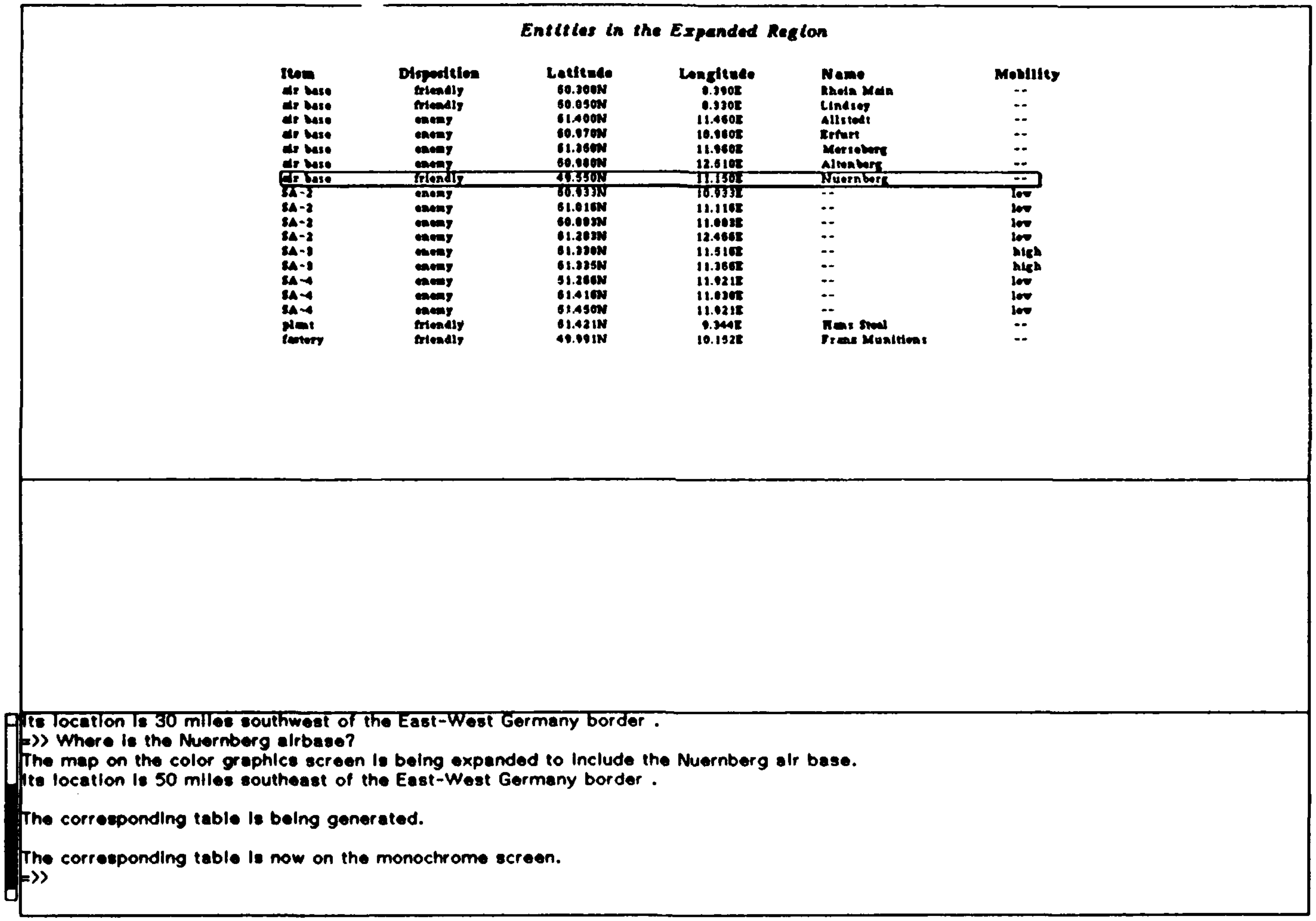
APPENDIX A

Example CUBRICON Dialogue

This appendix contains an example dialogue between a human user and CUBRICON. These examples were selected to illustrate key CUBRICON capabilities and include both verbal dialogue and associated graphics.

USER: **"Display the Fulda Gap region."**
USER: **"Where is the Franz Munitions factory?"**
USER: **"Where is the Nuernberg airbase?"**
NOTE: The system performs a zoom out since the user's task has not changed.



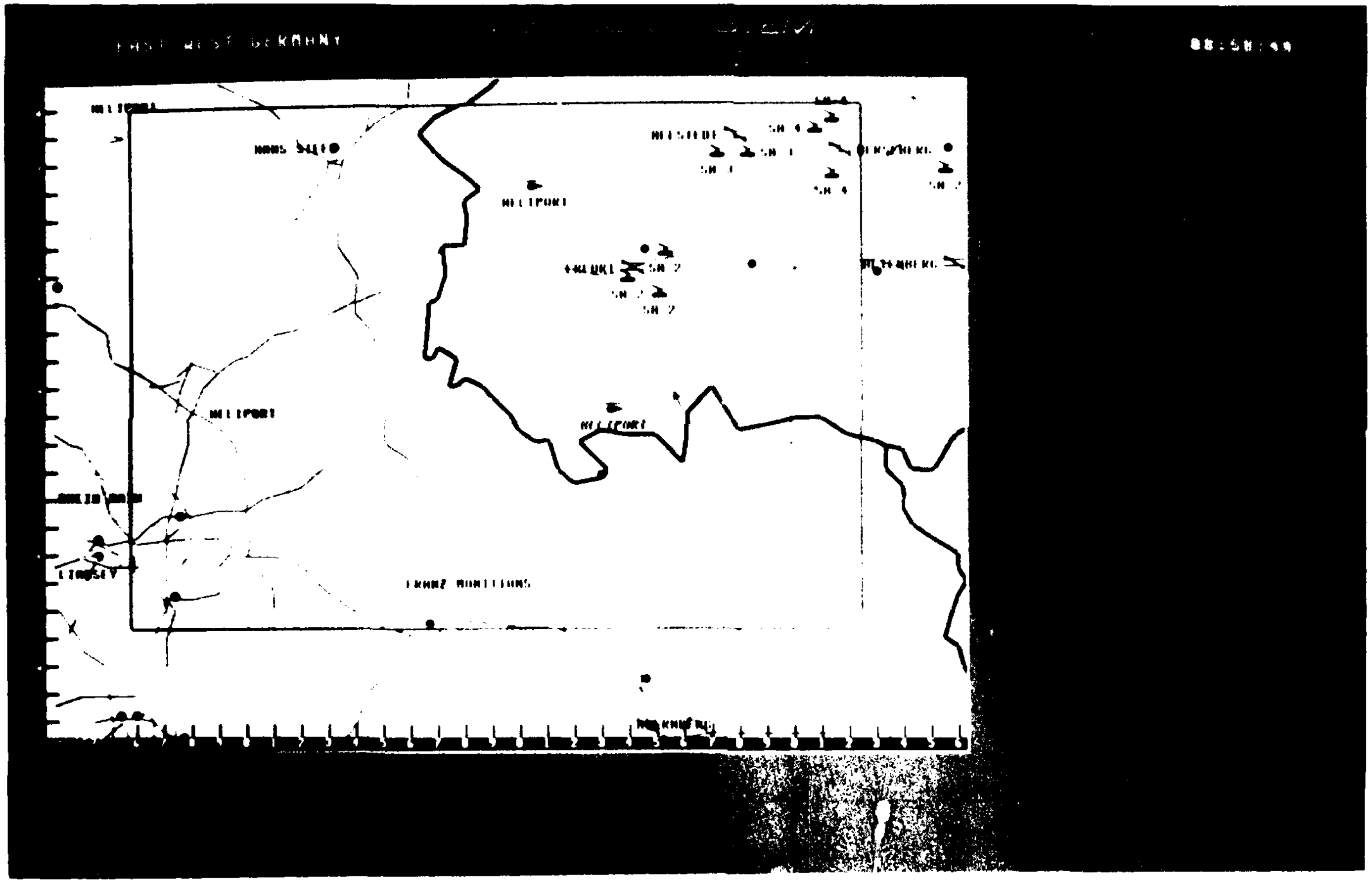


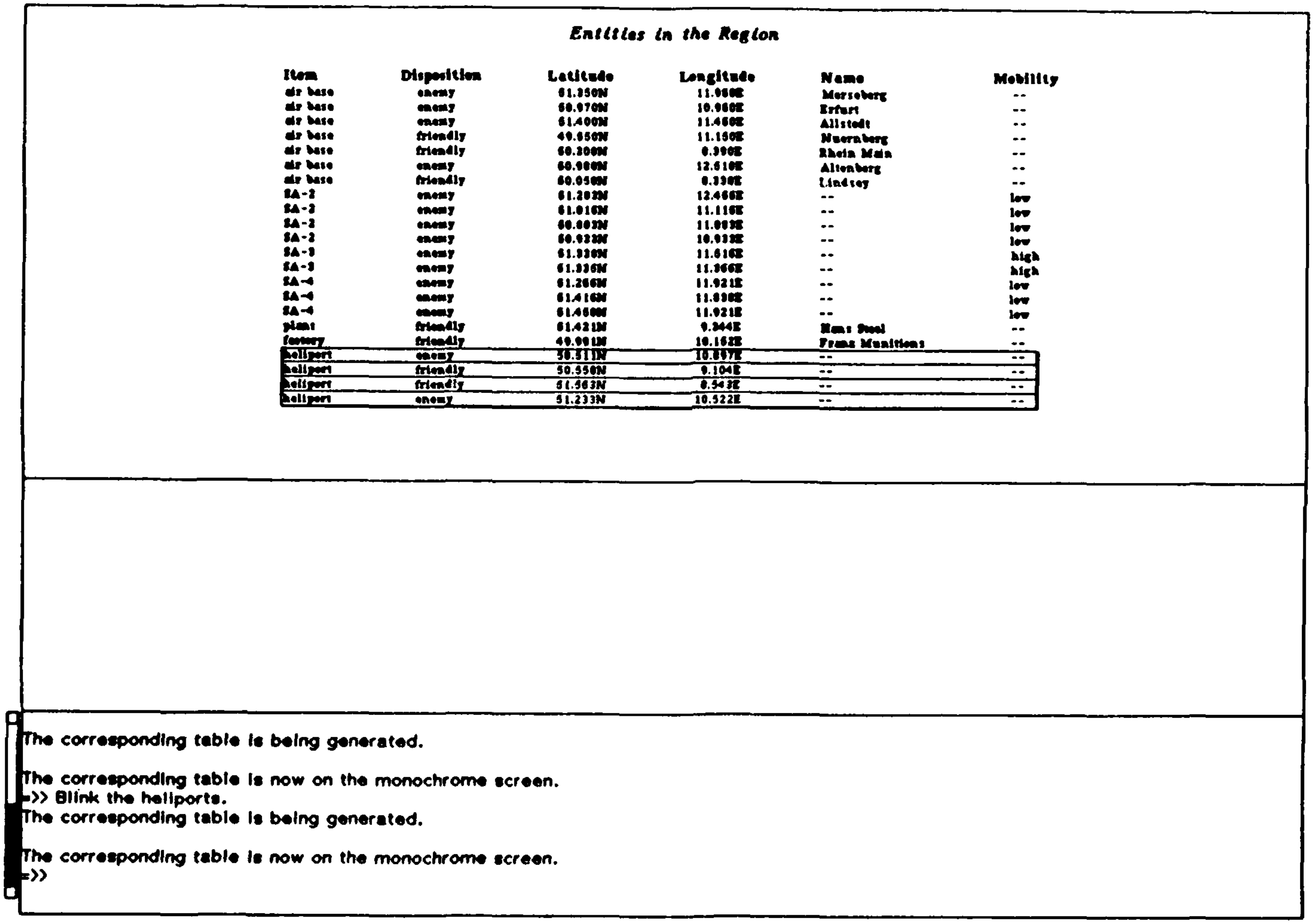
Entities in the Expanded Region

Item	Disposition	Latitude	Longitude	Name	Mobility
air base	friendly	50.300N	8.390E	Rhein Main	--
air base	friendly	50.050N	8.330E	Lindsey	--
air base	enemy	51.400N	11.460E	Allstedt	--
air base	enemy	50.970N	10.960E	Erfurt	--
air base	enemy	51.360N	11.960E	Merseberg	--
air base	enemy	50.980N	12.510E	Altenberg	--
air base	friendly	49.550N	11.150E	Nuernberg	--
SA-2	enemy	50.933N	10.933E	--	low
SA-2	enemy	51.016N	11.116E	--	low
SA-2	enemy	50.883N	11.883E	--	low
SA-2	enemy	51.283N	12.466E	--	low
SA-3	enemy	51.330N	11.516E	--	high
SA-3	enemy	51.335N	11.366E	--	high
SA-4	enemy	51.266N	11.921E	--	low
SA-4	enemy	51.416N	11.830E	--	low
SA-4	enemy	51.450N	11.921E	--	low
plant	friendly	51.421N	9.344E	Hans Steel	--
factory	friendly	49.991N	10.152E	Franz Munitions	--

```
Its location is 30 miles southwest of the East-West Germany border .
=>> Where is the Nuernberg airbase?
The map on the color graphics screen is being expanded to include the Nuernberg air base.
Its location is 50 miles southeast of the East-West Germany border .

The corresponding table is being generated.

The corresponding table is now on the monochrome screen.
=>>
```

USER: **"Blink the heliports."**





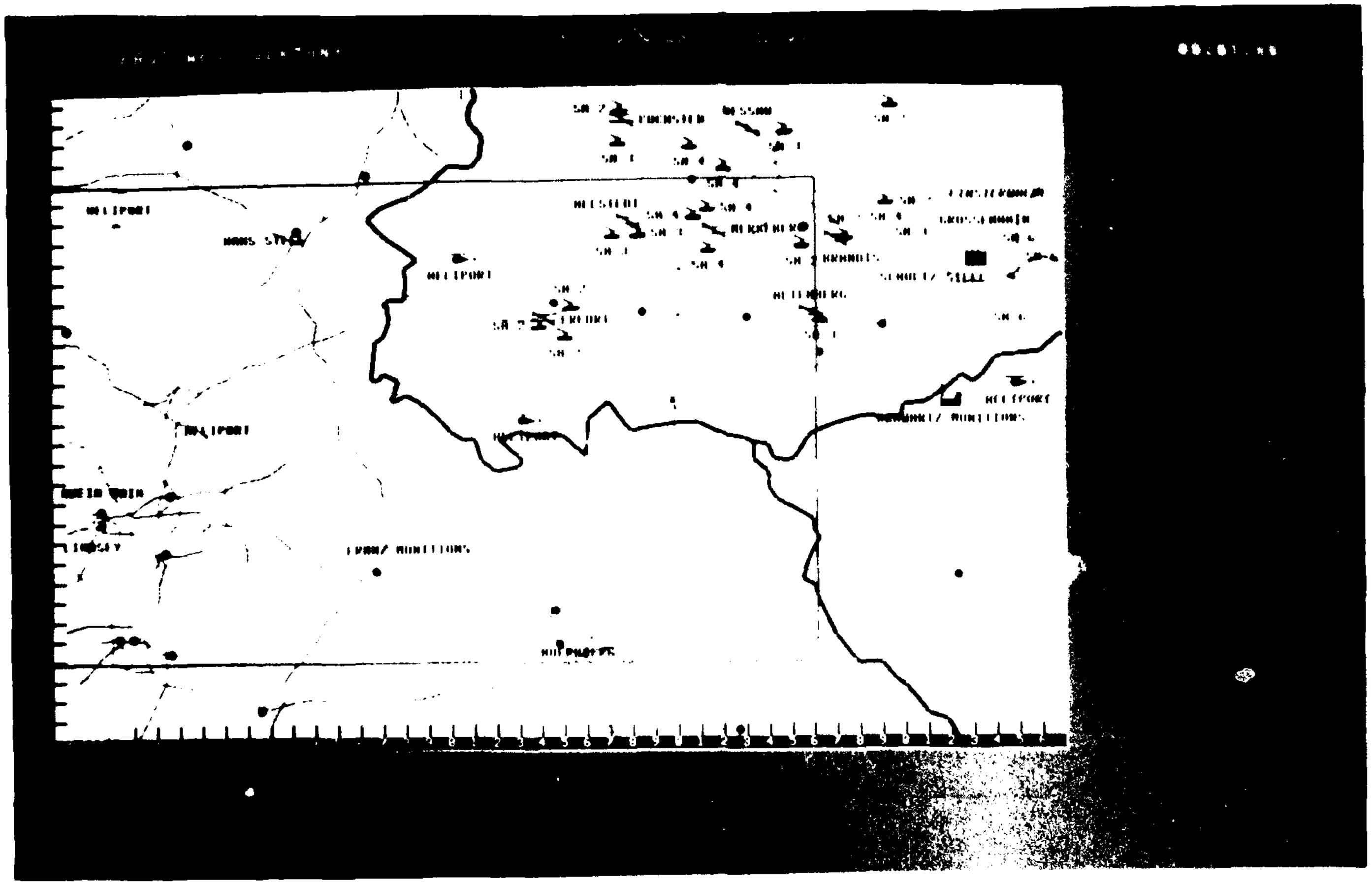
Entities in the Region

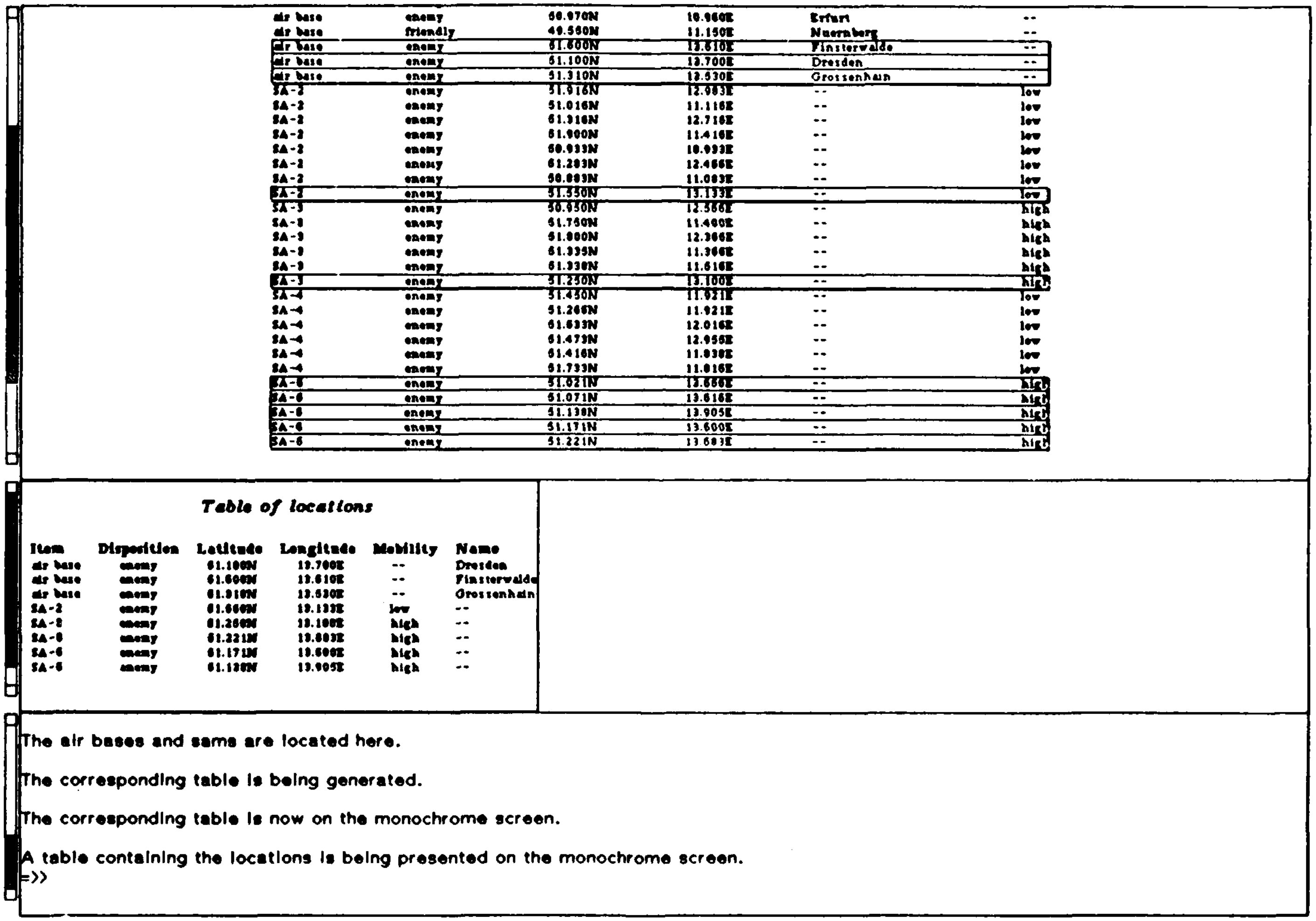
Item	Disposition	Latitude	Longitude	Name	Mobility
air base	enemy	51.350N	11.950E	Merseberg	--
air base	enemy	50.970N	10.960E	Erfurt	--
air base	enemy	51.400N	11.460E	Allstedt	--
air base	friendly	49.550N	11.150E	Nuernberg	--
air base	friendly	50.200N	8.390E	Rhein Main	--
air base	enemy	50.900N	12.510E	Altenberg	--
air base	friendly	50.050N	8.330E	Lindsey	--
SA-2	enemy	51.202N	12.460E	--	low
SA-2	enemy	51.016N	11.116E	--	low
SA-2	enemy	50.803N	11.083E	--	low
SA-2	enemy	50.932N	10.933E	--	low
SA-3	enemy	51.330N	11.616E	--	high
SA-3	enemy	51.336N	11.366E	--	high
SA-4	enemy	51.266N	11.921E	--	low
SA-4	enemy	51.416N	11.830E	--	low
SA-4	enemy	51.460N	11.921E	--	low
plant	friendly	51.421N	9.344E	Hans Steel	--
factory	friendly	49.991N	10.162E	Franz Munitions	--
heliport	enemy	50.511N	10.897E	--	--
heliport	friendly	50.550N	9.104E	--	--
heliport	friendly	51.563N	8.543E	--	--
heliport	enemy	51.233N	10.522E	--	--

```
The corresponding table is being generated.

The corresponding table is now on the monochrome screen.
=>> Blink the heliports.
The corresponding table is being generated.

The corresponding table is now on the monochrome screen.
=>>
```

USER: **"What are the threats around the Dresden airbase?"**

NOTE: The system performs a zoom out since the user's task has not changed.





air base	enemy	50.970N	10.960E	Erfurt	--
air base	friendly	49.500N	11.150E	Nuernberg	--
air base	enemy	51.600N	13.610E	Finsterwalde	--
air base	enemy	51.100N	13.700E	Dresden	--
air base	enemy	51.310N	13.530E	Grossenhain	--
SA-2	enemy	51.916N	12.983E	--	low
SA-2	enemy	51.016N	11.116E	--	low
SA-2	enemy	51.316N	12.716E	--	low
SA-2	enemy	51.900N	11.416E	--	low
SA-2	enemy	50.933N	10.933E	--	low
SA-2	enemy	51.283N	12.466E	--	low
SA-2	enemy	50.883N	11.083E	--	low
SA-2	enemy	51.550N	13.133E	--	low
SA-3	enemy	50.950N	12.566E	--	high
SA-3	enemy	51.750N	11.400E	--	high
SA-3	enemy	51.800N	12.366E	--	high
SA-3	enemy	51.335N	11.366E	--	high
SA-3	enemy	51.330N	11.616E	--	high
SA-3	enemy	51.250N	13.100E	--	high
SA-4	enemy	51.450N	11.921E	--	low
SA-4	enemy	51.266N	11.921E	--	low
SA-4	enemy	51.533N	12.016E	--	low
SA-4	enemy	51.473N	12.950E	--	low
SA-4	enemy	51.416N	11.830E	--	low
SA-4	enemy	51.733N	11.816E	--	low
SA-6	enemy	51.021N	13.650E	--	high
SA-6	enemy	51.071N	13.616E	--	high
SA-6	enemy	51.138N	13.905E	--	high
SA-6	enemy	51.171N	13.600E	--	high
SA-6	enemy	51.221N	13.683E	--	high

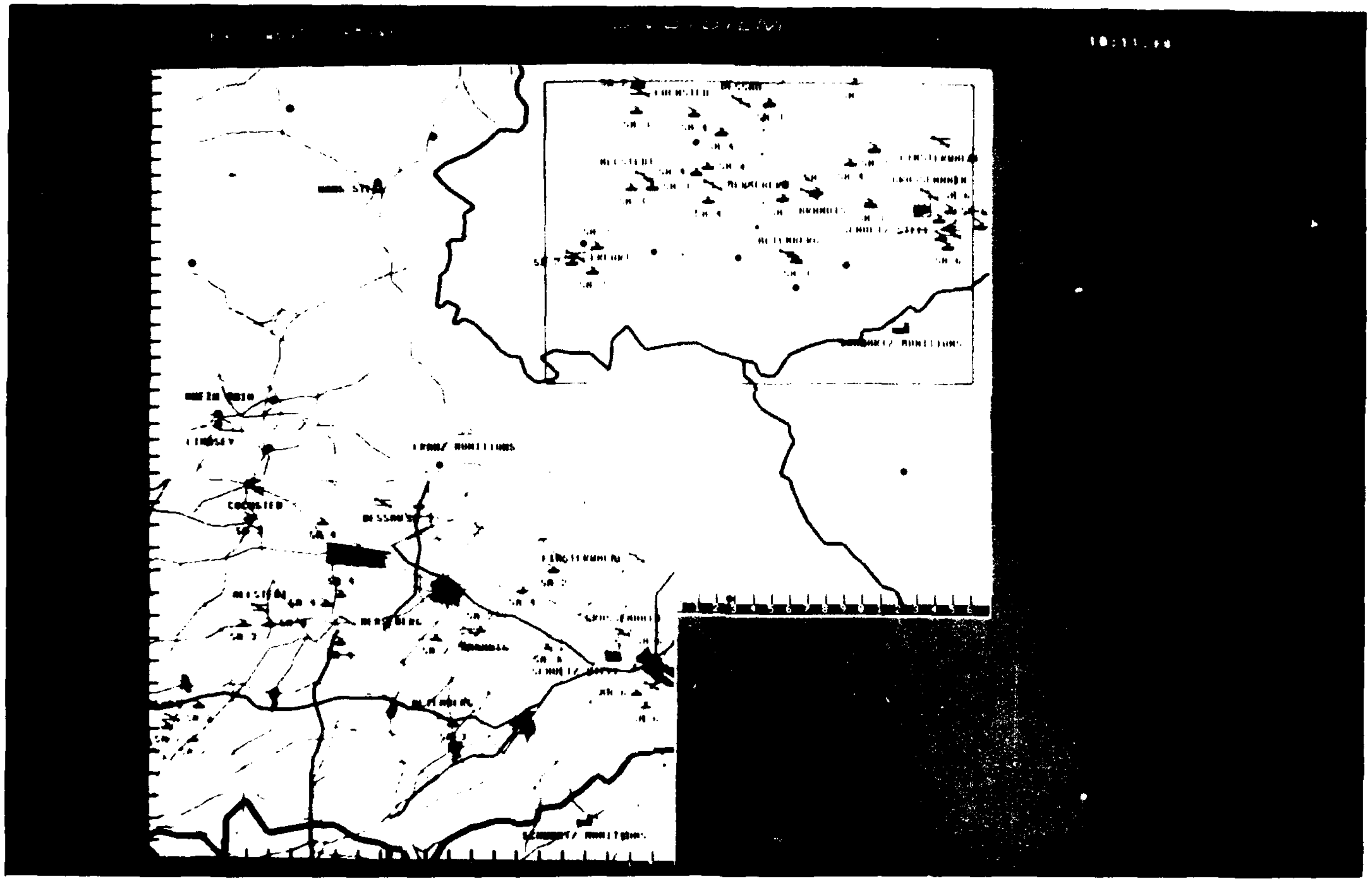
Table of locations

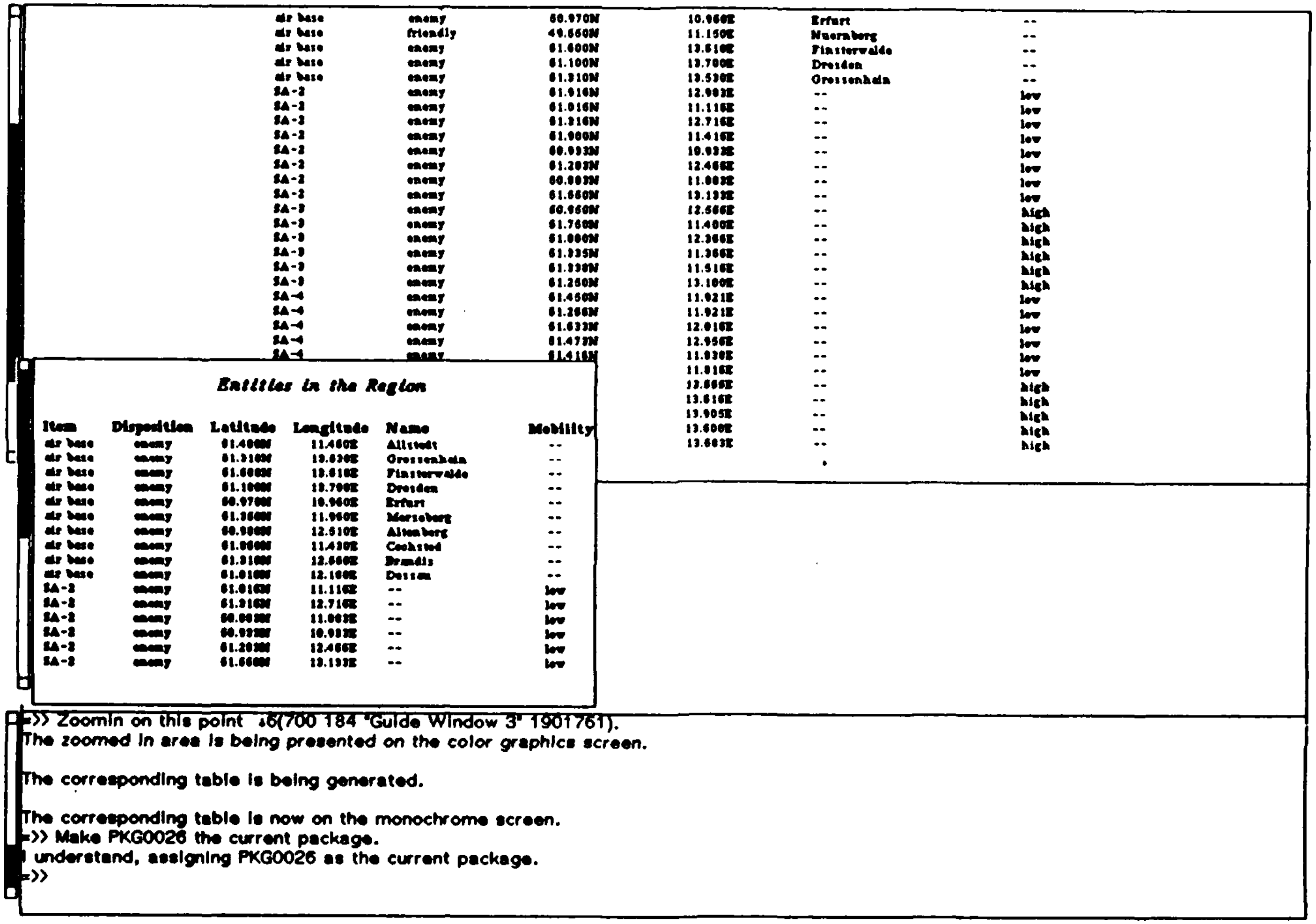
Item	Disposition	Latitude	Longitude	Mobility	Name
air base	enemy	51.100N	13.700E	--	Dresden
air base	enemy	51.600N	13.610E	--	Finsterwalde
air base	enemy	51.310N	13.530E	--	Grossenhain
SA-2	enemy	51.550N	13.133E	low	--
SA-3	enemy	51.250N	13.100E	high	--
SA-6	enemy	51.221N	13.683E	high	--
SA-6	enemy	51.171N	13.600E	high	--
SA-6	enemy	51.138N	13.905E	high	--

```
The air bases and sams are located here.

The corresponding table is being generated.

The corresponding table is now on the monochrome screen.

A table containing the locations is being presented on the monochrome screen.
=>>
```

USER: **"Zoom in on this point <pt-near-Dresden>."**

NOTE: The system performs a minor zoom in since the user's task has not changed.





air base	enemy	50.970N	10.960E	Erfurt	--
air base	friendly	49.650N	11.150E	Nuernberg	--
air base	enemy	51.600N	13.610E	Finsterwalde	--
air base	enemy	51.100N	13.700E	Dresden	--
air base	enemy	51.310N	13.530E	Grossenhain	--
SA-2	enemy	51.916N	12.903E	--	low
SA-2	enemy	51.016N	11.116E	--	low
SA-2	enemy	51.316N	12.716E	--	low
SA-2	enemy	51.900N	11.416E	--	low
SA-2	enemy	50.933N	10.933E	--	low
SA-2	enemy	51.203N	12.466E	--	low
SA-2	enemy	50.803N	11.083E	--	low
SA-2	enemy	51.660N	13.133E	--	low
SA-3	enemy	50.950N	12.566E	--	high
SA-3	enemy	51.760N	11.400E	--	high
SA-3	enemy	51.000N	12.366E	--	high
SA-3	enemy	51.335N	11.366E	--	high
SA-3	enemy	51.330N	11.516E	--	high
SA-3	enemy	51.250N	13.100E	--	high
SA-4	enemy	51.450N	11.921E	--	low
SA-4	enemy	51.266N	11.921E	--	low
SA-4	enemy	51.633N	12.016E	--	low
SA-4	enemy	51.473N	12.956E	--	low
SA-4	enemy	51.416N	11.030E	--	low
			11.816E	--	low
			13.866E	--	high
			13.616E	--	high
			13.905E	--	high
			13.600E	--	high
			13.683E	--	high

Entities in the Region

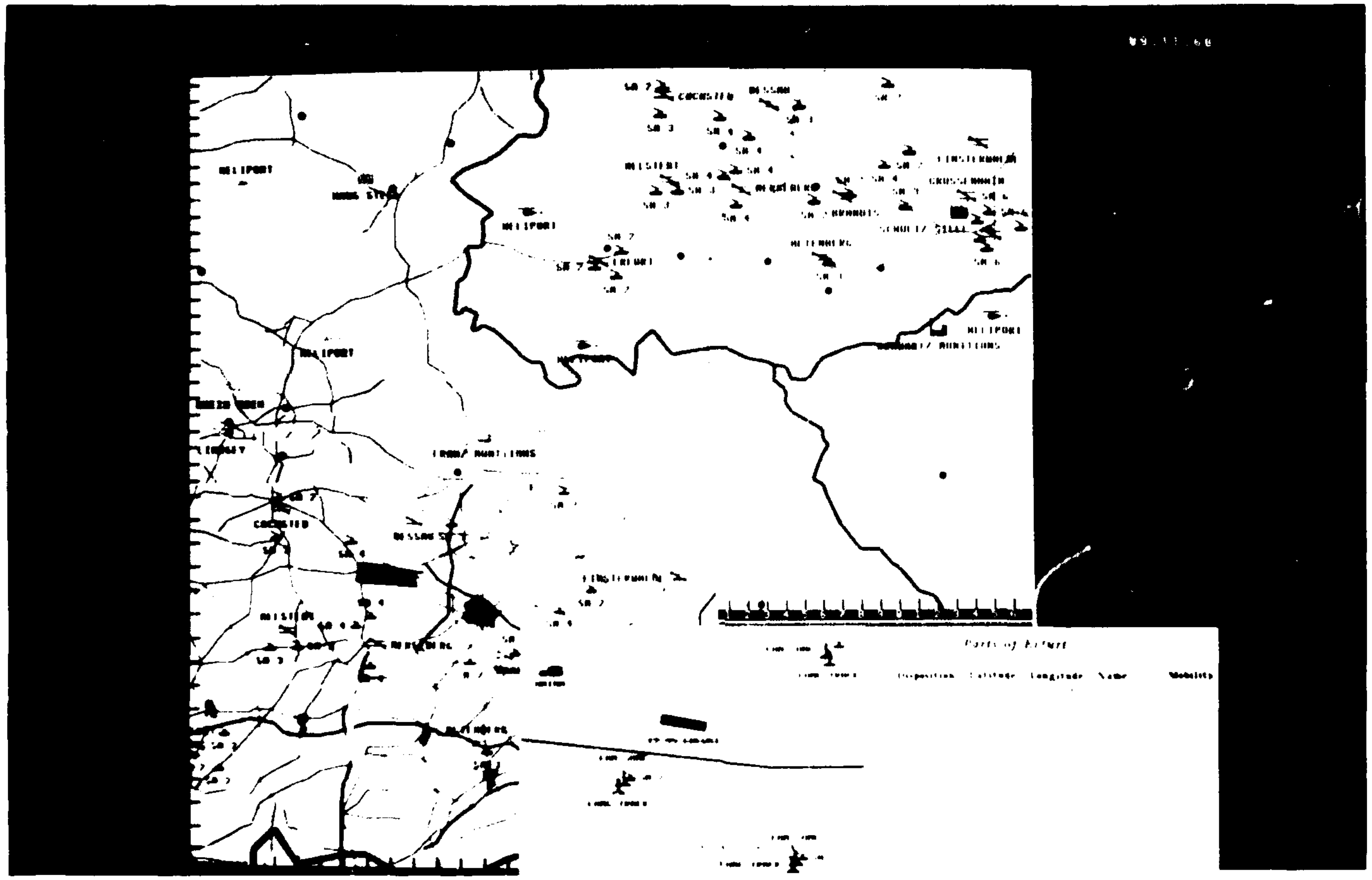
Item	Disposition	Latitude	Longitude	Name	Mobility
air base	enemy	51.400N	11.460E	Allstedt	--
air base	enemy	51.310N	13.530E	Grossenhain	--
air base	enemy	51.600N	13.610E	Finsterwalde	--
air base	enemy	51.100N	13.700E	Dresden	--
air base	enemy	50.970N	10.960E	Erfurt	--
air base	enemy	51.350N	11.960E	Merseberg	--
air base	enemy	50.980N	12.510E	Altenberg	--
air base	enemy	51.860N	11.430E	Cochsted	--
air base	enemy	51.310N	12.660E	Brandis	--
air base	enemy	51.810N	12.160E	Dessau	--
SA-2	enemy	51.016N	11.116E	--	low
SA-2	enemy	51.316N	12.716E	--	low
SA-2	enemy	50.803N	11.083E	--	low
SA-2	enemy	50.933N	10.933E	--	low
SA-2	enemy	51.203N	12.466E	--	low
SA-2	enemy	51.660N	13.133E	--	low

```
=>> Zoomin on this point  ↓6(700 184 "Guide Window 3" 1901761).
The zoomed in area is being presented on the color graphics screen.

The corresponding table is being generated.

The corresponding table is now on the monochrome screen.
=>> Make PKG0026 the current package.
I understand, assigning PKG0026 as the current package.
=>>
```

USER: **"Make PKG0026 the current package."**

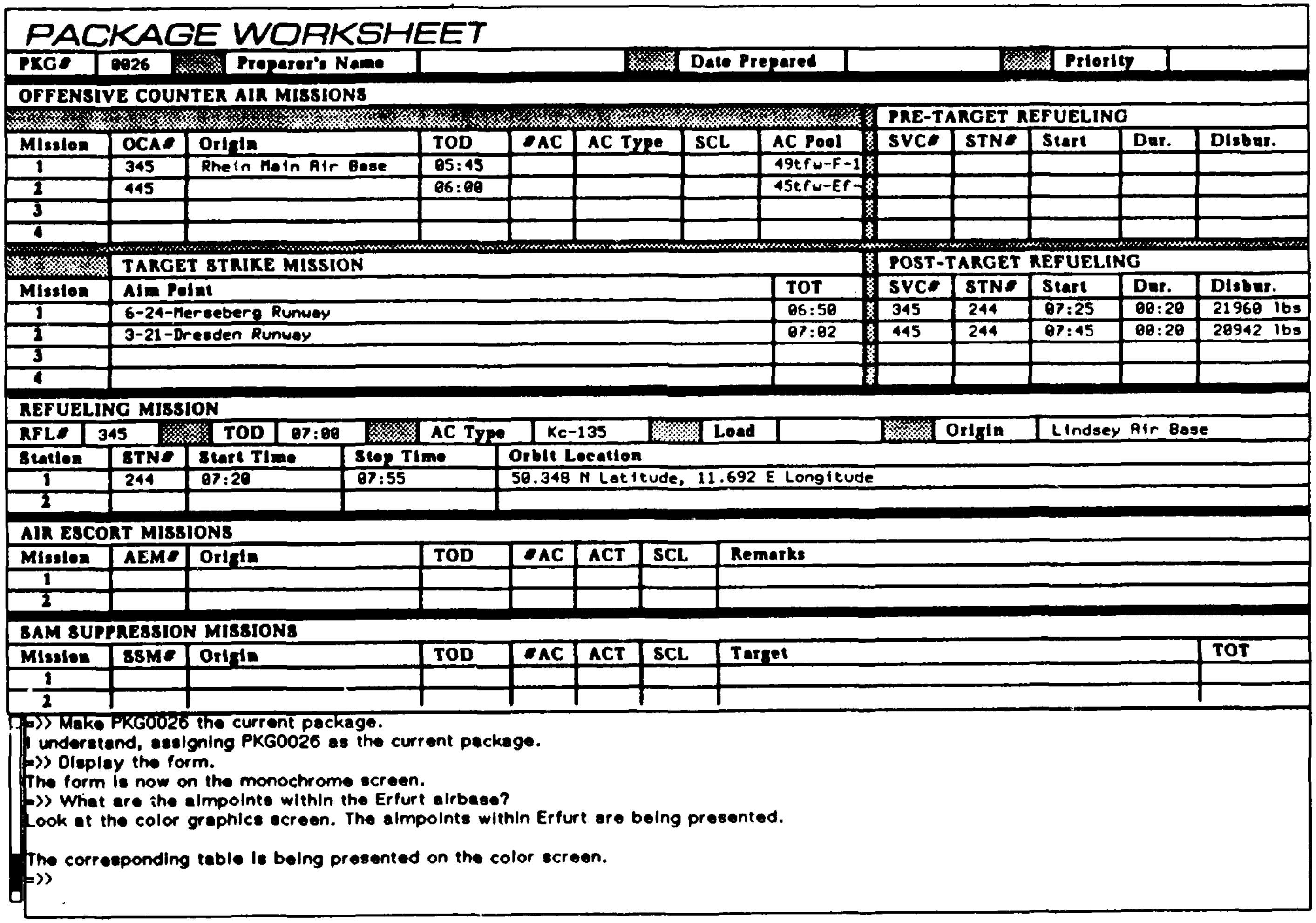
USER: **"Display the form."**

USER: **"What are the aimpoints within the Erfurt airbase?"**



Parts of Erfurt

Disposition Latitude Longitude Name Mobility



PACKAGE WORKSHEET

PKG#	0026	Preparer's Name		Date Prepared		Priority	

OFFENSIVE COUNTER AIR MISSIONS

								PRE-TARGET REFUELING				
Mission	OCA#	Origin	TOD	#AC	AC Type	SCL	AC Pool	SVC#	STN#	Start	Dur.	Disbur.
1	345	Rhein Main Air Base	05:45				49tfu-F-1					
2	445		06:00				45tfu-Ef-					
3												
4												

	TARGET STRIKE MISSION		POST-TARGET REFUELING				
Mission	Aim Point	TOT	SVC#	STN#	Start	Dur.	Disbur.
1	6-24-Merseberg Runway	06:50	345	244	07:25	00:20	21960 lbs
2	3-21-Dresden Runway	07:02	445	244	07:45	00:20	20942 lbs
3							
4							

REFUELING MISSION

RFL#	345	TOD	07:00	AC Type	Kc-135	Load		Origin	Lindsey Air Base

Station	STN#	Start Time	Stop Time	Orbit Location
1	244	07:20	07:55	50.348 N Latitude, 11.692 E Longitude
2				

AIR ESCORT MISSIONS

Mission	AEM#	Origin	TOD	#AC	ACT	SCL	Remarks
1							
2							

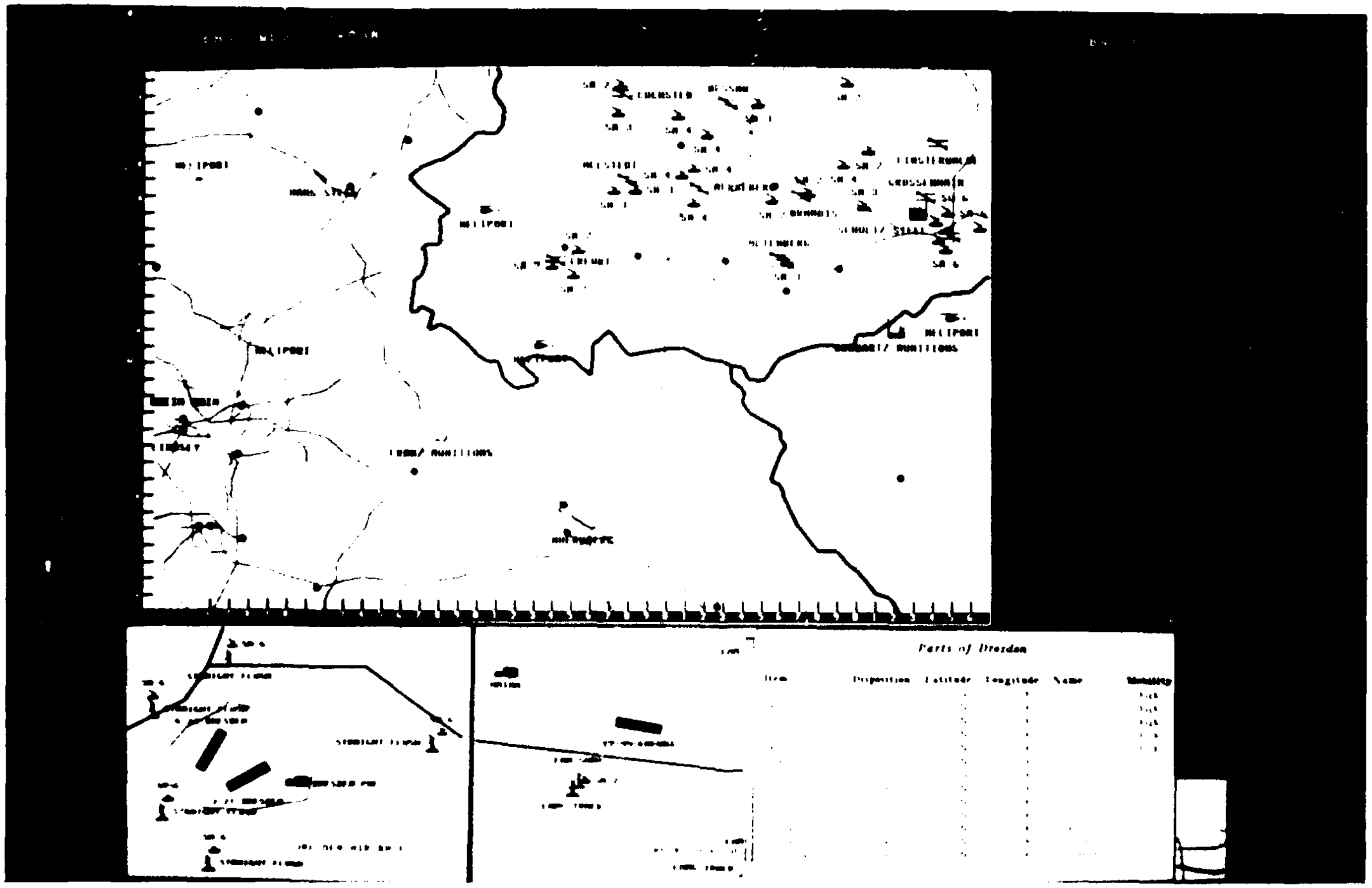
SAM SUPPRESSION MISSIONS

Mission	SSM#	Origin	TOD	#AC	ACT	SCL	Target	TOT
1								
2								

=>> Make PKG0026 the current package.
I understand, assigning PKG0026 as the current package.
=>> Display the form.
The form is now on the monochrome screen.
=>> What are the aimpoints within the Erfurt airbase?
Look at the color graphics screen. The aimpoints within Erfurt are being presented.

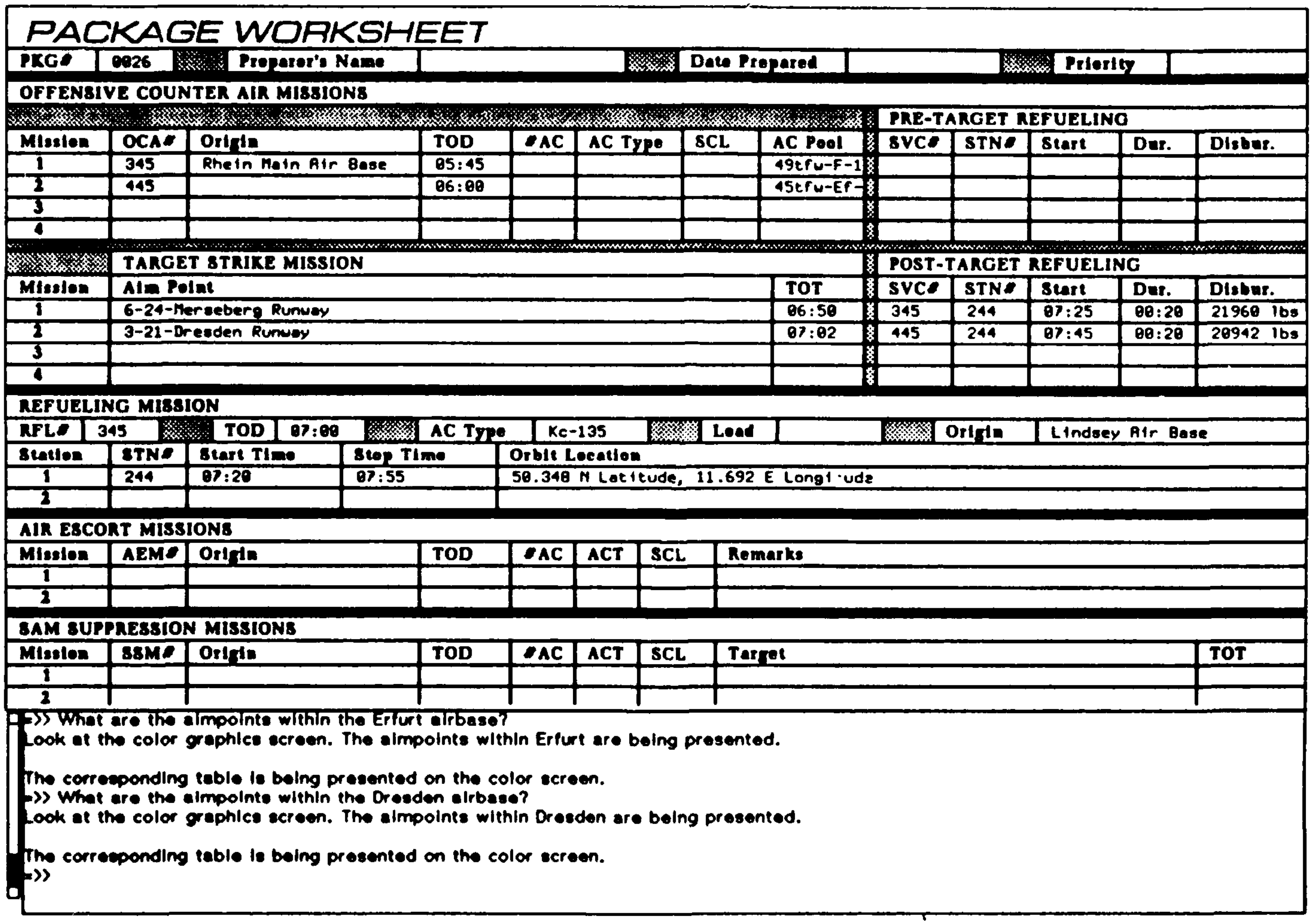
The corresponding table is being presented on the color screen.
=>>

USER: "What are the aimpoints within the Dresden airbase?"



Parts of Dresden

Item	Disposition	Latitude	Longitude	Name	Mobility



PACKAGE WORKSHEET

PKG#	0026	▨	Preparer's Name		▨	Date Prepared		▨	Priority	

OFFENSIVE COUNTER AIR MISSIONS

Mission	OCA#	Origin	TOD	#AC	AC Type	SCL	AC Pool	SVC#	STN#	Start	Dur.	Disbur.
								PRE-TARGET REFUELING				
1	345	Rhein Main Air Base	05:45				49tfu-F-1					
2	445		06:00				45tfu-Ef-					
3												
4												

TARGET STRIKE MISSION

Mission	Aim Point	TOT	SVC#	STN#	Start	Dur.	Disbur.
			POST-TARGET REFUELING				
1	6-24-Merseberg Runway	06:50	345	244	07:25	00:20	21960 lbs
2	3-21-Dresden Runway	07:02	445	244	07:45	00:20	20942 lbs
3							
4							

REFUELING MISSION

RFL#	345	▨	TOD	07:00	▨	AC Type	Kc-135	▨	Lead		▨	Origin	Lindsey Air Base

Station	STN#	Start Time	Stop Time	Orbit Location
1	244	07:20	07:55	50.348 N Latitude, 11.692 E Longi·ude
2				

AIR ESCORT MISSIONS

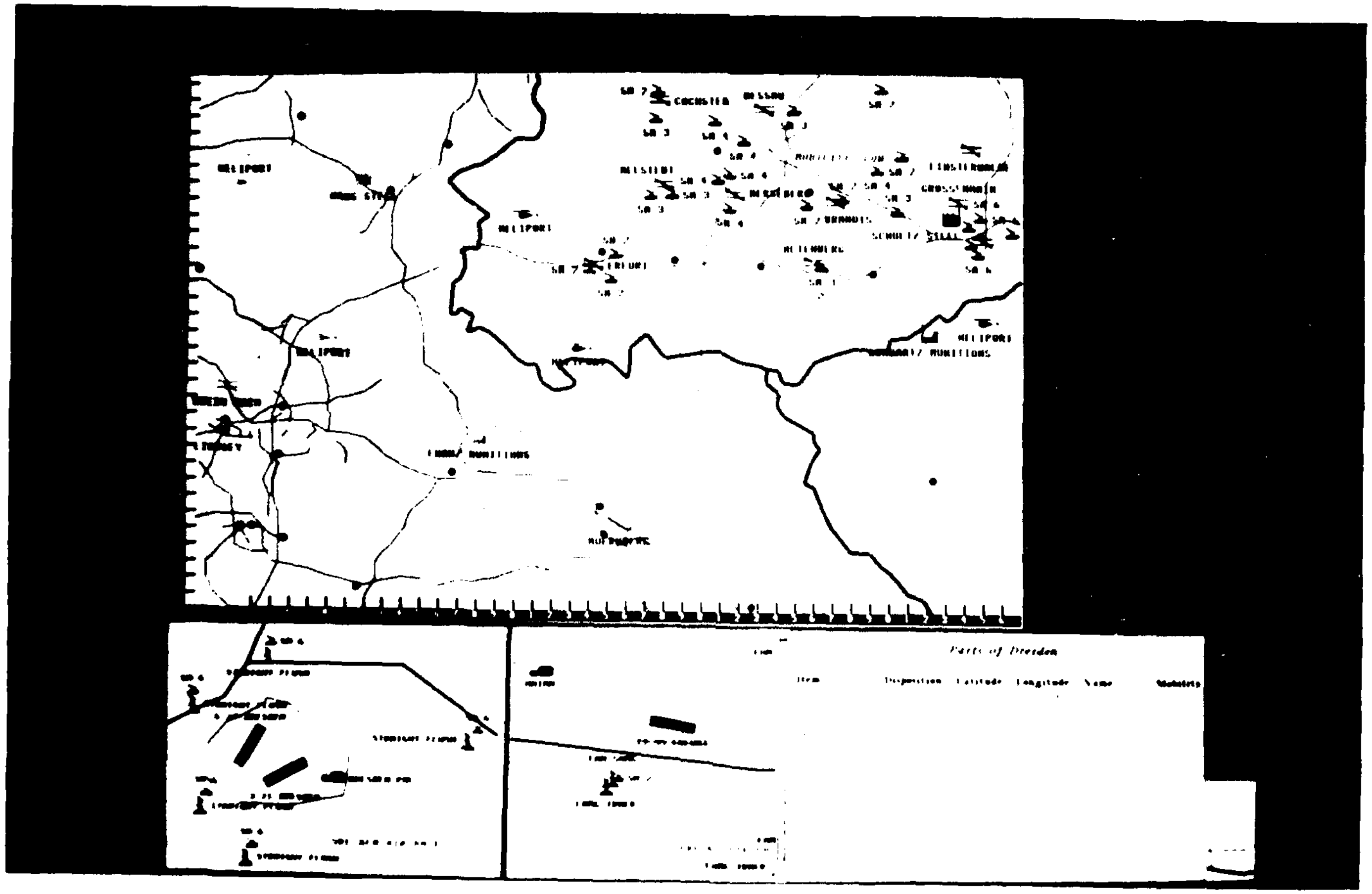
Mission	AEM#	Origin	TOD	#AC	ACT	SCL	Remarks
1							
2							

SAM SUPPRESSION MISSIONS

Mission	SSM#	Origin	TOD	#AC	ACT	SCL	Target	TOT
1								
2								

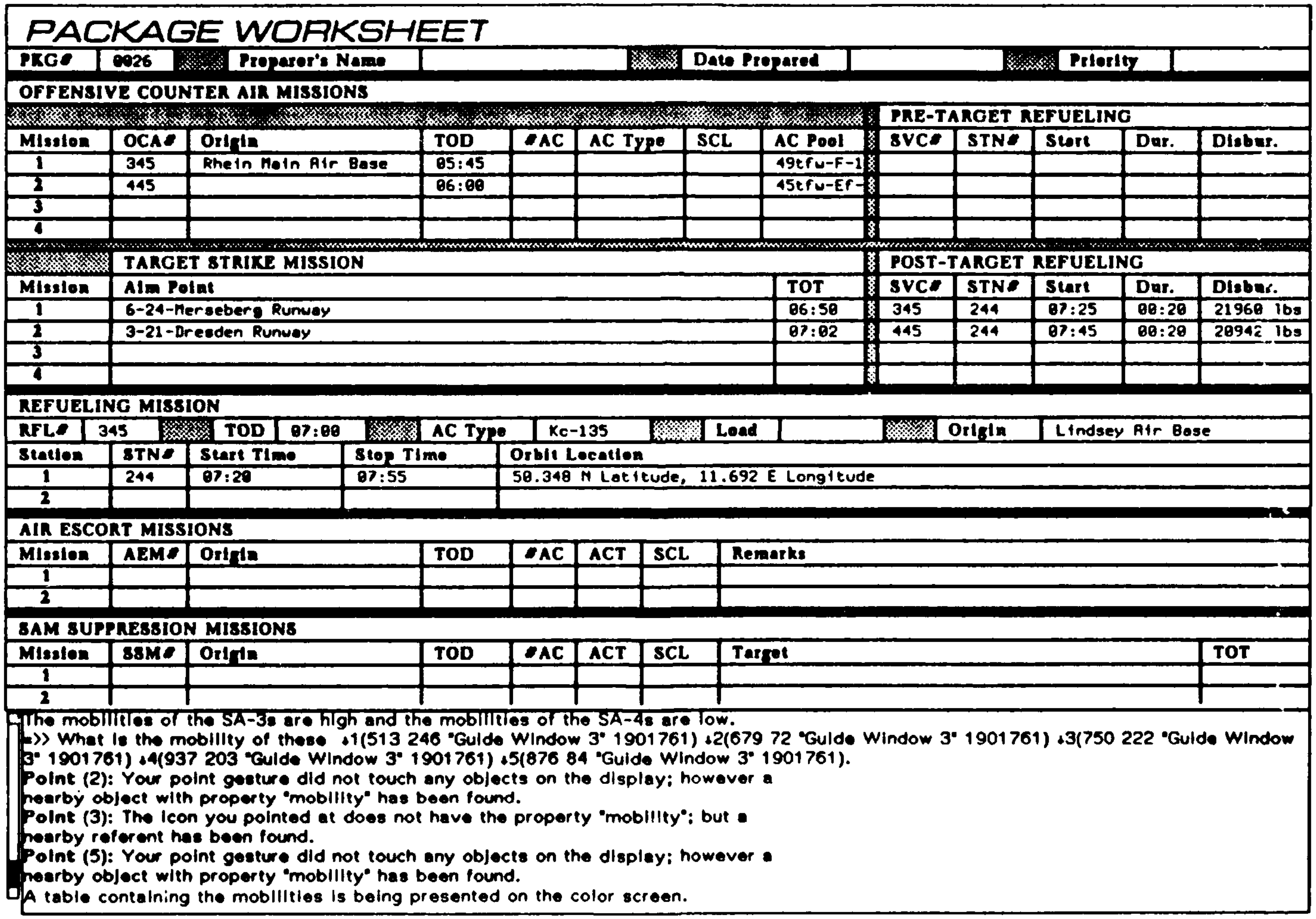
```
>> What are the aimpoints within the Erfurt airbase?
Look at the color graphics screen. The aimpoints within Erfurt are being presented.

The corresponding table is being presented on the color screen.
>> What are the aimpoints within the Dresden airbase?
Look at the color graphics screen. The aimpoints within Dresden are being presented.

The corresponding table is being presented on the color screen.
>>
```

USER: “What is the mobility of these <point [illegible] <point [illegible] <point [illegible] <point [illegible] <point [illegible] ?”



Parts of Dresden

Item	Disposition	Latitude	Longitude	Name	Mobility



PACKAGE WORKSHEET

PKG#	0026	Preparer's Name		Date Prepared		Priority	

OFFENSIVE COUNTER AIR MISSIONS

								PRE-TARGET REFUELING				
Mission	OCA#	Origin	TOD	#AC	AC Type	SCL	AC Pool	SVC#	STN#	Start	Dur.	Disbur.
1	345	Rhein Main Air Base	05:45				49tfw-F-1					
2	445		06:00				45tfw-Ef-					
3												
4												

	TARGET STRIKE MISSION		POST-TARGET REFUELING				
Mission	Aim Point	TOT	SVC#	STN#	Start	Dur.	Disbur.
1	6-24-Merseberg Runway	06:58	345	244	07:25	00:20	21960 lbs
2	3-21-Dresden Runway	07:02	445	244	07:45	00:20	20942 lbs
3							
4							

REFUELING MISSION

RFL#	345	TOD	07:00	AC Type	Kc-135	Load		Origin	Lindsey Air Base

Station	STN#	Start Time	Stop Time	Orbit Location
1	244	07:20	07:55	50.348 N Latitude, 11.692 E Longitude
2				

AIR ESCORT MISSIONS

Mission	AEM#	Origin	TOD	#AC	ACT	SCL	Remarks
1							
2							

SAM SUPPRESSION MISSIONS

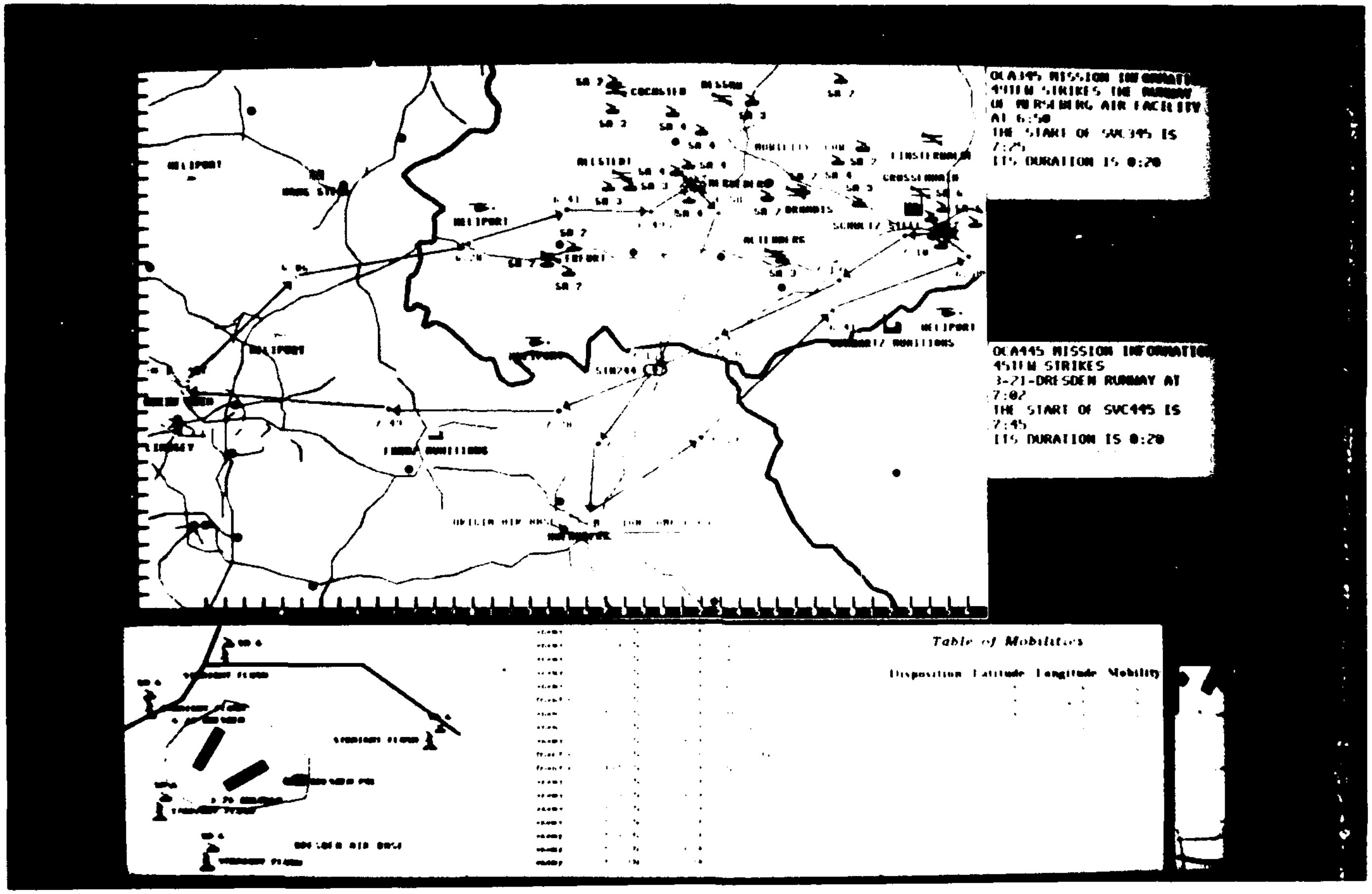
Mission	SSM#	Origin	TOD	#AC	ACT	SCL	Target	TOT
1								
2								

The mobilities of the SA-3s are high and the mobilities of the SA-4s are low.
>> What is the mobility of these ↓1(513 246 "Guide Window 3" 1901761) ↓2(679 72 "Guide Window 3" 1901761) ↓3(750 222 "Guide Window 3" 1901761) ↓4(937 203 "Guide Window 3" 1901761) ↓5(876 84 "Guide Window 3" 1901761).
Point (2): Your point gesture did not touch any objects on the display; however a nearby object with property "mobility" has been found.
Point (3): The icon you pointed at does not have the property "mobility"; but a nearby referent has been found.
Point (5): Your point gesture did not touch any objects on the display; however a nearby object with property "mobility" has been found.
A table containing the mobilities is being presented on the color screen.

USER: **"Enter the Nuernberg airbase here - point-form-445-origin-slot. . ."**

NOTE: Flight paths for the OCA345 and OCS445 missions were entered interactively, but not shown in these figures.

USER: **"Present the OCA345 and OCA445 mission plans."**

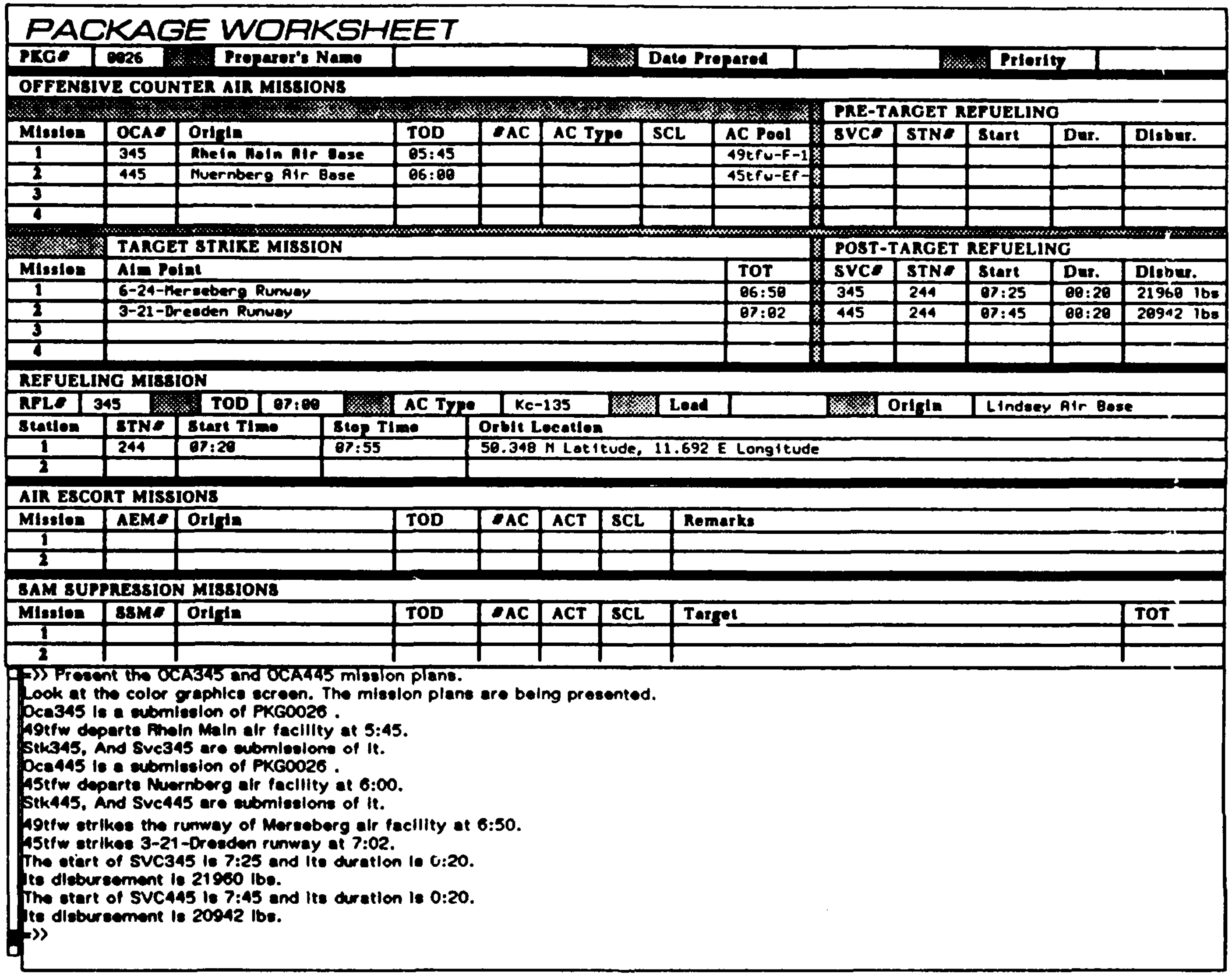


OCA345 MISSION INFORMATI
49TH W STRIKES THE RUNWAY
OF MERSEBURG AIR FACILITY
AT 6:50
THE START OF SVC345 IS
7:25
ITS DURATION IS 0:20

OCA445 MISSION INFORMATIO
45TH W STRIKES
3-21-DRESDEN RUNWAY AT
7:02
THE START OF SVC445 IS
7:45
ITS DURATION IS 0:20

Table of Mobilities

Disposition	Latitude	Longitude	Mobility
[illegible]	[illegible]	[illegible]	[illegible]



PACKAGE WORKSHEET

PKG#	0026	Preparer's Name		Date Prepared		Priority

OFFENSIVE COUNTER AIR MISSIONS

Mission	OCA#	Origin	TOD	#AC	AC Type	SCL	AC Pool
1	345	Rhein Main Air Base	05:45				49tfu-F-1
2	445	Nuernberg Air Base	06:00				45tfu-Ef-
3							
4							

PRE-TARGET REFUELING

SVC#	STN#	Start	Dur.	Disbur.

TARGET STRIKE MISSION

Mission	Aim Point	TOT
1	6-24-Merseberg Runway	06:50
2	3-21-Dresden Runway	07:02
3		
4		

POST-TARGET REFUELING

SVC#	STN#	Start	Dur.	Disbur.
345	244	07:25	00:20	21960 lbs
445	244	07:45	00:20	20942 lbs

REFUELING MISSION

RFL#	345	TOD	07:00	AC Type	Kc-135	Lead		Origin	Lindsey Air Base

Station	STN#	Start Time	Stop Time	Orbit Location
1	244	07:20	07:55	50.348 N Latitude, 11.692 E Longitude
2				

AIR ESCORT MISSIONS

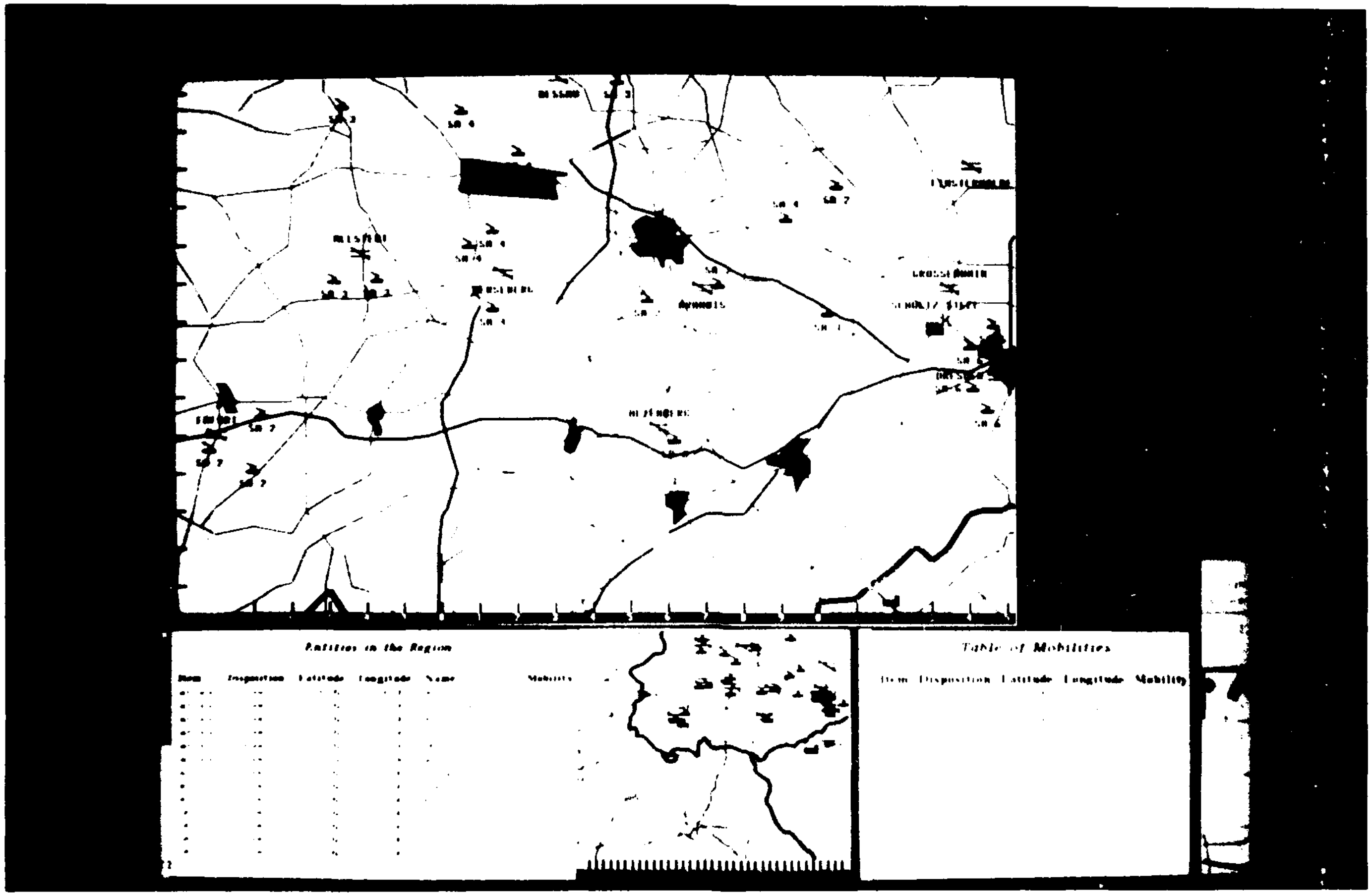
Mission	AEM#	Origin	TOD	#AC	ACT	SCL	Remarks
1							
2							

SAM SUPPRESSION MISSIONS

Mission	SSM#	Origin	TOD	#AC	ACT	SCL	Target	TOT
1								
2								

```
=>> Present the OCA345 and OCA445 mission plans.
Look at the color graphics screen. The mission plans are being presented.
Oca345 is a submission of PKG0026 .
49tfw departs Rhein Main air facility at 5:45.
Stk345, And Svc345 are submissions of it.
Oca445 is a submission of PKG0026 .
45tfw departs Nuernberg air facility at 6:00.
Stk445, And Svc445 are submissions of it.
49tfw strikes the runway of Merseberg air facility at 6:50.
45tfw strikes 3-21-Dresden runway at 7:02.
The start of SVC345 is 7:25 and its duration is 0:20.
Its disbursement is 21960 lbs.
The start of SVC445 is 7:45 and its duration is 0:20.
Its disbursement is 20942 lbs.
=>>
```

USER: **"Make PKG0066 the current package."**
USER: **"Remove the flight paths."**
USER: **"Zoom in on this point < point-on-map >."**
NOTE: The system performs a major zoom in since the user's task changed.

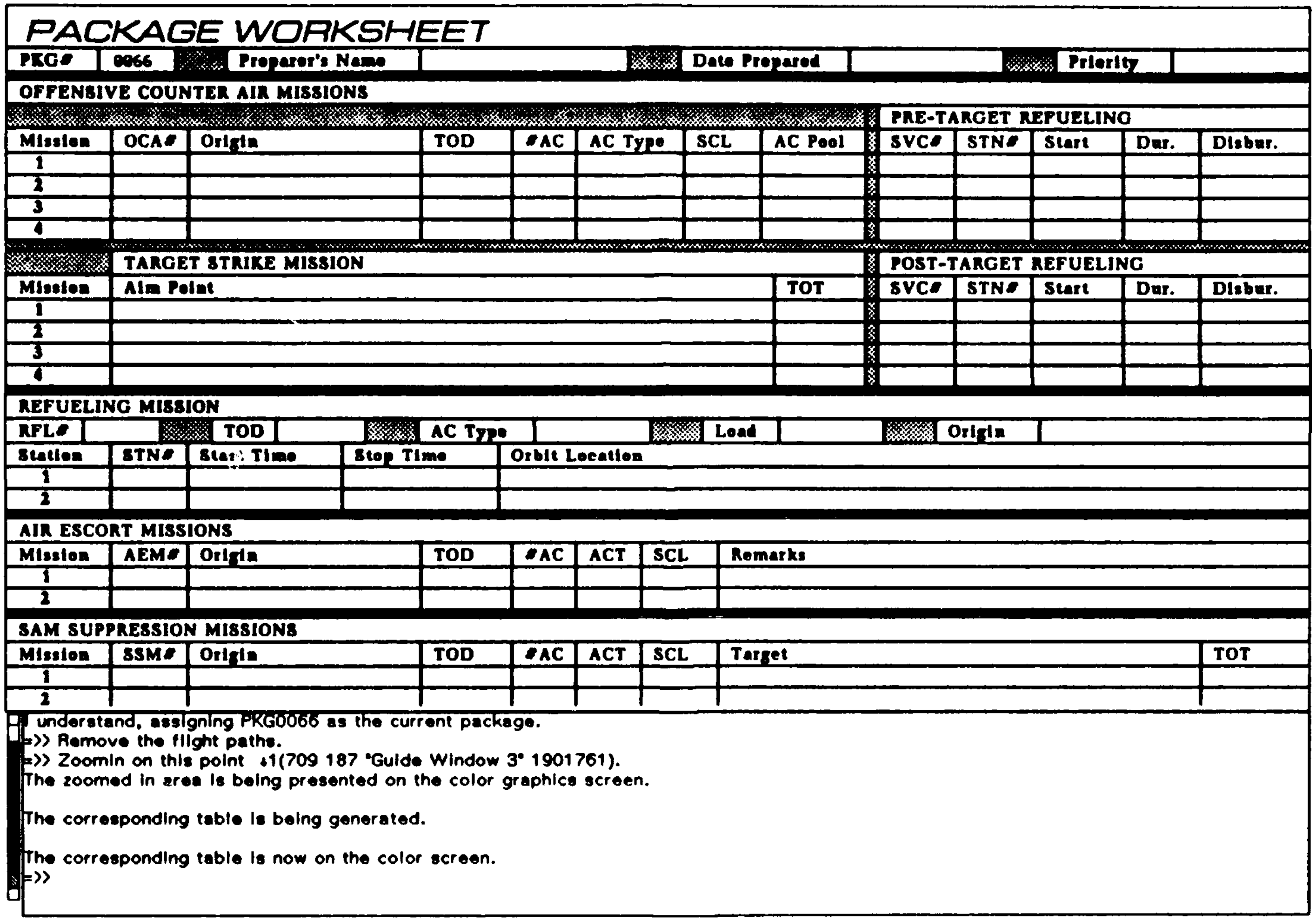


Entities in the Region

Item	Disposition	Latitude	Longitude	Name	Mobility

Table of Mobilities

Item	Disposition	Latitude	Longitude	Mobility



PACKAGE WORKSHEET

PKG#	0066	Preparer's Name		Date Prepared		Priority	

OFFENSIVE COUNTER AIR MISSIONS

								PRE-TARGET REFUELING				
Mission	OCA#	Origin	TOD	#AC	AC Type	SCL	AC Pool	SVC#	STN#	Start	Dur.	Disbur.
1												
2												
3												
4												

	TARGET STRIKE MISSION		POST-TARGET REFUELING				
Mission	Aim Point	TOT	SVC#	STN#	Start	Dur.	Disbur.
1							
2							
3							
4							

REFUELING MISSION

RFL#		TOD		AC Type		Load		Origin	

Station	STN#	Start Time	Stop Time	Orbit Location
1				
2				

AIR ESCORT MISSIONS

Mission	AEM#	Origin	TOD	#AC	ACT	SCL	Remarks
1							
2							

SAM SUPPRESSION MISSIONS

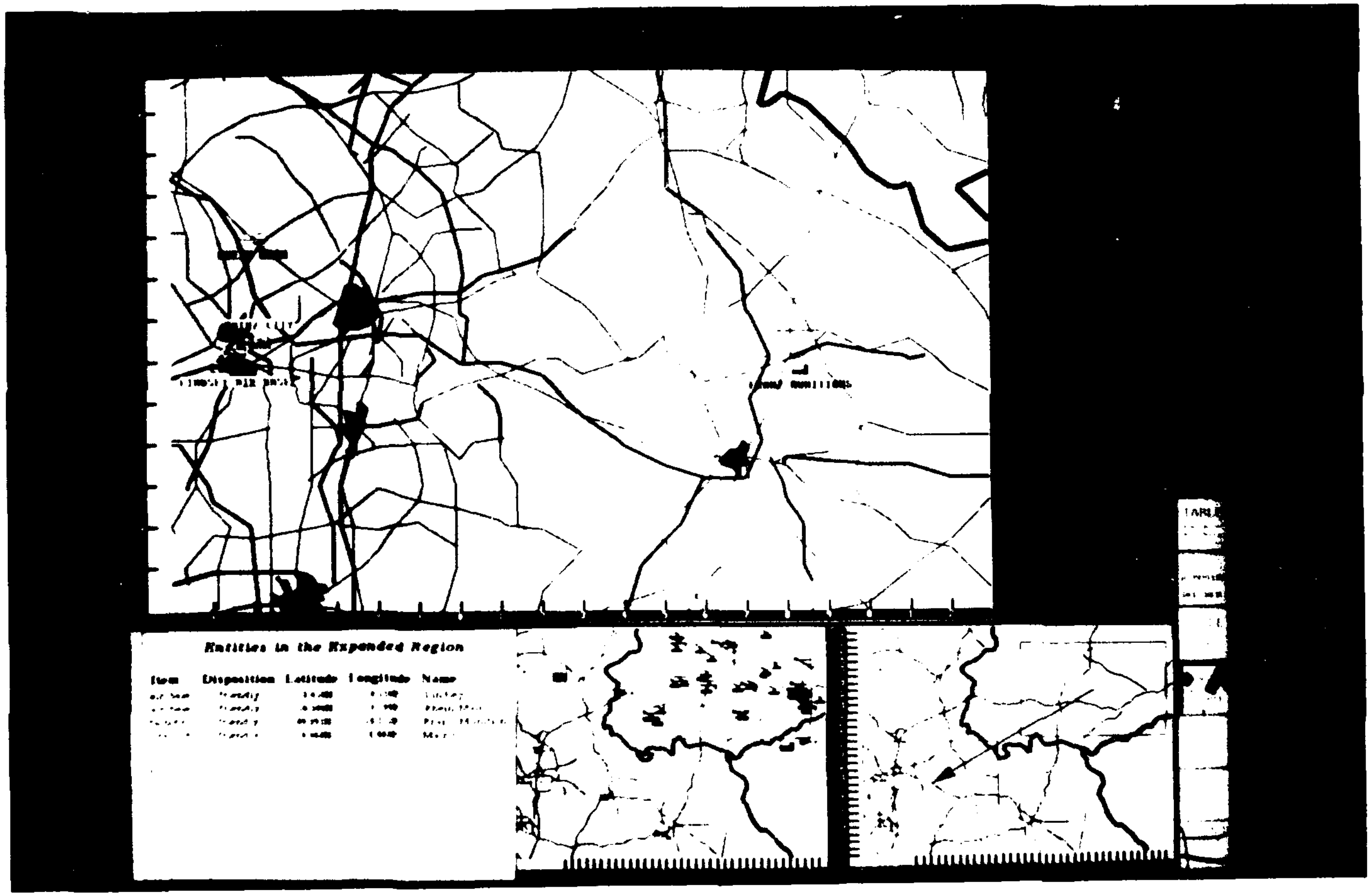
Mission	SSM#	Origin	TOD	#AC	ACT	SCL	Target	TOT
1								
2								

```
I understand, assigning PKG0066 as the current package.
=>> Remove the flight paths.
=>> Zoomin on this point ↓1(709 187 "Guide Window 3" 1901761).
The zoomed in area is being presented on the color graphics screen.

The corresponding table is being generated.

The corresponding table is now on the color screen.
=>>
```

USER: **"Make PKG0023 the current package."**

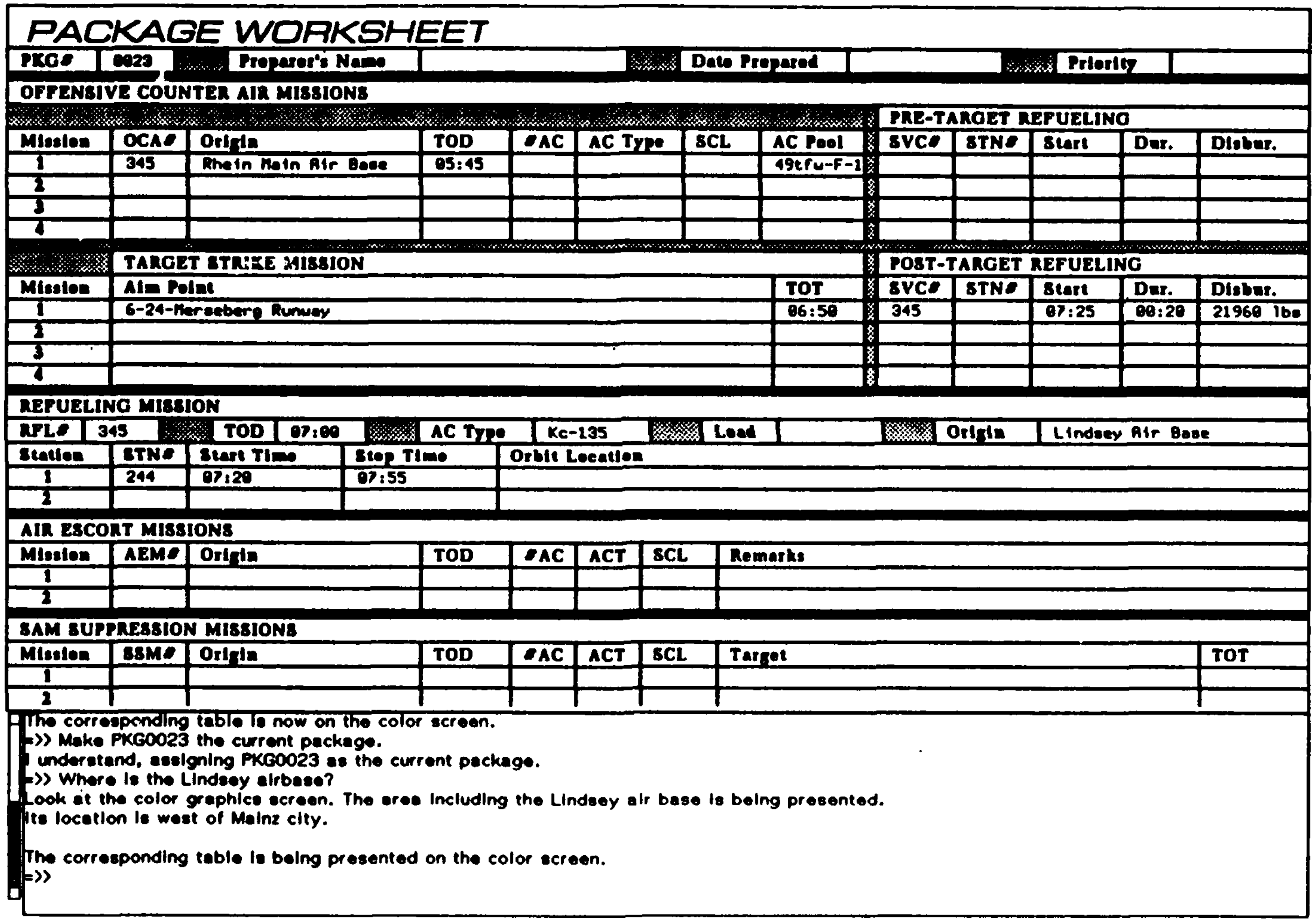
USER: **"Where is the Lindsey airbase?"**

NOTE: The system performs a pan since the user's task changed.



Entities in the Expanded Region

Item	Disposition	Latitude	Longitude	Name
air base	friendly	[illegible]	[illegible]	Lindsey
air base	friendly	[illegible]	[illegible]	[illegible]
[illegible]	friendly	[illegible]	[illegible]	[illegible]
[illegible]	friendly	[illegible]	[illegible]	[illegible]



PACKAGE WORKSHEET

PKG#	0023	Preparer's Name		Date Prepared		Priority	

OFFENSIVE COUNTER AIR MISSIONS

								PRE-TARGET REFUELING				
Mission	OCA#	Origin	TOD	#AC	AC Type	SCL	AC Pool	SVC#	STN#	Start	Dur.	Disbur.
1	345	Rhein Main Air Base	05:45				49tfu-F-1					
2												
3												
4												

	TARGET STRIKE MISSION		POST-TARGET REFUELING				
Mission	Aim Point	TOT	SVC#	STN#	Start	Dur.	Disbur.
1	6-24-Merseberg Runway	06:50	345		07:25	00:20	21960 lbs
2							
3							
4							

REFUELING MISSION

RFL#	345	TOD	07:00	AC Type	Kc-135	Lead		Origin	Lindsey Air Base

Station	STN#	Start Time	Stop Time	Orbit Location
1	244	07:20	07:55	
2				

AIR ESCORT MISSIONS

Mission	AEM#	Origin	TOD	#AC	ACT	SCL	Remarks
1							
2							

SAM SUPPRESSION MISSIONS

Mission	SSM#	Origin	TOD	#AC	ACT	SCL	Target	TOT
1								
2								

```
The corresponding table is now on the color screen.
=>> Make PKG0023 the current package.
I understand, assigning PKG0023 as the current package.
=>> Where is the Lindsey airbase?
Look at the color graphics screen. The area including the Lindsey air base is being presented.
Its location is west of Mainz city.

The corresponding table is being presented on the color screen.
=>>
```

APPENDIX B

CUBRICON Grammar and Lexicon

This section contains a description of the CUBRICON lexicon and a simplified version of the grammar. The grammar defines the subset of natural language which must be used when making inputs to CUBRICON. The lexicon includes a listing of the lexicon that is understood by CUBRICON. This lexicon is in two parts: (1) a complete alphabetical list of the lexicon and (2) a listing of CUBRICON vocabulary by word type.

A version of this lexicon and grammar were used to support the CUBRICON evaluation.

B Input Grammar

This appendix contains a description of the CUBRICON input grammar for multi-modal sentences. The first part contains a surface description in BNF notation, the second part lists the actual ATN used by the multi-modal parser.

BNF Input Grammar Description

This section contains a surface level description of the multi-modal language accepted by the CUBRICON system. It is a partly syntactic and partly semantic description in BNF-like notation, provided mainly for ease of reading and understanding, and as a reference document. The motivation for this description is pragmatic rather than linguistic. It describes the most important types of sentences and phrases accepted, without pretending to be exhaustive. Example sentences and phrases are provided. Note that this essentially context-free description allows meaningless sentences to be derived from it, which will most likely not be accepted by the system. For a more precise and complete description of the input grammar, refer to the following section.

The notational conventions used in this section are as follows:

```
<symbol-1>  →  <symbol-2><symbol-3> | <symbol-2><symbol-4>
<symbol-2>  →  an
<symbol-3>  →  example
<symbol-4>  →  instance
```

meaning that <symbol-1> can be rewritten as the string consisting of <symbol-2> and <symbol-3>, or the string consisting of <symbol-2> and <symbol-4> in that order, which translates to "an example" or "an instance" after substituting these symbols as well. Blanks are not represented in the description to keep it readable. Any non-terminal symbol, i.e. a symbol that cannot occur as such in the input stream, appears between pointed brackets <>; terminal symbols, i.e. words as they occur in the input stream, appear without brackets. Alternatives are separated by the vertical bar |. Examples are given in between lines of BNF forms, with parentheses () mimicking the BNF structure, as in

```
<noun-phrase>  →  <determiner><noun>
                  (an)(example)
```

Mouse clicks are represented by four special nonterminal symbols: ↓, ⊙, ⊕ and ⊗. Where ↓ appears in the grammar, it means that *exactly one* mouse click *must* occur *at that place* in the phrase it appears in, for instance:

```
<locative>  →  here ↓
```

which expands to "here <point>".

Where ⊙ appears in the grammar, it means that *exactly one* mouse click *must* occur *anywhere within* the phrase it appears in, for instance:

<demonstrative> → this ⊙

which expands to either "this <point>" or "<point> this".

Where ⊕ occurs in the grammar, it means that *one or more* mouse clicks *must* occur *anywhere within* the phrase it appears in, including before the first or after the last word, for instance:

<noun-phrase> → ⊕

which may expand to one or more mouse clicks.

Where ⊗ occurs in the grammar, it means that *zero or more* mouse clicks *may* occur *anywhere within* the phrase it appears in, including before the first or after the last word, for instance:

<noun-phrase> → ⊗<determiner><noun>

which may expand to phrases like "the airbase", "<point> the airbase", "the <point> airbase", "the airbase <point>", or the same with more than one <point> (mouse click) in succession, or even a phrase like "the <point> sams <point> <point>", where all three mouse clicks are taken to refer together to a plural referent for "sams".

For readability we will put the ⊗ symbol at the beginning of phrases, and the ↓, ⊙ and ⊕ symbols at the place of the noun they are substituted for, keeping in mind that only the ↓ symbol implies restrictions to that particular position.

Since it is impossible to describe which combinations of natural language and mouse clicks are acceptable (interpretable) for the system at this level of description, we will make no attempt to do so.

The nonterminal symbol # is used to represent a digit 0...9.

Additional comments are provided in the grammar between square brackets []. The actual grammar starts below.

<sentence> → <question> ?
(where is the Dresden airbase) (?)
| <imperative> .
(display the FG region) (.)

<question> → where <be-verb> <noun-phrase>
(where) (is) (the Dresden airbase)
| what <be-verb> <noun-phrase>
(what) (is) (this)
| what <be-verb> <noun-phrase> <locative>
(what) (are) (the aimpoints) (within the Dresden airbase)
| what <noun> <be-verb> <locative>
(what) (ac-pools) (are) (at the Nuernberg airbase)
| what <be-verb> <noun-phrase> <prepositional-complement>
(what) (is) (the mobility) (of this ⊙)

```
                         | <be-verb> <demonstrative> <noun-phrase>
                         (is) (this ⊙) (a sam)
<imperative>       →     <command-verb> <demonstrative>
                         (remove) (this window ⊙)
                         | <command-verb> <demonstrative> <locative>
                         (enter) (this ⊙) (here ↓)
                         | <command-verb> <demonstrative> <noun-phrase>
                         (make) (this ⊙) (the current package)
                         | <command-verb> <noun-phrase>
                         (display) (the FG region)
                         | <command-verb> <noun-phrase> <locative>
                         (enter) (20 minutes) (here ↓)
                         | <command-verb> <noun-phrase> <noun-phrase>
                         (make) (PKG0026) (the current package)
                         | <command-verb> <noun-phrase> <prepositional-complement>
                         (plan) (a flight path) (for OCA345)
                         | <command-verb> <prepositional-complement>
                         (zoom in) (on this point ⊙)

<noun-phrase>      →      ⊗ <det> <noun>
                          ⊗ (the) (sam)
                          | ⊗ <det> <proper-name-group> <noun>
                          ⊗ (the) (Nuernberg) (airbase)
                          | ⊗ <det> <noun> <proper-name>
                          ⊗ (the) (ac-pool) (45TFW-EF-111E)
                          | ⊗ <proper-name>
                          ⊗ (STN244)
                          | ⊕
                          | the value <proper-name>
                          (the) (value) (KC-135)
                          | the value <number> <unit>
                          (the) (value) (21960) (lbs)
                          | <time>
                          (7:45)
                          | <duration>
                          (20 minutes)

<proper-name-group>   →   <proper-name>
                          (OCA123)
                          | <proper-name> <conjunction> <proper-name-group>
                          (OCA123) (and) (OCA345)
```

```
<demonstrative>  →  this ⊙
                    | these ⊕
                    | this <noun> ⊙
                    (this) (sam) ⊙
                    | these <noun> ⊕
                    (these) (sams) ⊕

<locative>  →  here ↓
               | <locative-preposition> <noun-phrase>
               (around) (⊗ the Dresden airbase)

<prepositional-complement>  →  <complement-preposition> <noun-phrase>
                               (for) (OCA123)
                               | <complement-preposition> <demonstrative>
                               (on) (this point ⊙)

<time>  →  <hours>:<minutes>
           (6) (:) (45)

<duration>  →  <number> minutes
               (20) (minutes)

<be-verb>  →  is | are

<command-verb>  →  assign | blink | call | display | enter | expose
                   | highlight | list | make | plan | present | remove
                   | zoom in
```

[nouns can occur in singular or pural form where appropriate]

```
<noun>  →  ac pool | ac type | ac-pool | aimpoint | air base
           | air escort mission | airbase | aircraft | aircraft unit
           | area | artillery | base | battalion | color
           | color graphics display | current | current mission plan
           | current package | disbursement | disposition | duration
           | factory | fighter | fighter base | flight | flight path
           | form | forms window | fuel | fuel storage tank | fuel tank
           | heliport | hour | location | map | minute | missile | mission
           | mission plan | mission plan form | mission-plan | mobility
           | munition | munition factory | name | nationality | number
           | OCA mission plan | OCA plan | orbit location | origin
           | package | package number | plant | point | population center
           | post-target refueling mission | pre-target refueling mission
           | preparer | priority | radar | refueling mission | region
           | sam | sam suppression mission | slot | start | steel plant
           | STN mission plan | stn number | STN plan | strike date
           | SVC mission plan | table | tank | tank battalions | target
           | target strike mission | task | threat | time | TOD | TOT
           | type | unit | value | window
```

<proper-name> → AEM### | Allstedt | Altenberg | Brandis | Cochsted
| Dessau | Dresden | East-West Germany | EF-111E
| Erfurt | EW | EW Germany | F-111F | F-15C | F-16C
| F-16D | F-4G | FG | Findsterwalde | Franz Munitions
| Fulda Gap | Grossenhain | Hans steel | KC-135
| Leipzig | Lindsey | Merseberg | Nuernberg | OCA
| OCA### | PKG### | PKG#### | RFL###
| Rhein Main | Schultz steel | Schwartz Munitions | SSM###
| Stargard | STN | STN### | SVC | SVC###
| SVC#### | TSM### | 34TFS | 34TFS-F-15C | 435TAW
| 45TFW | 45TFW-EF-111E | 45TFW-F-16C | 45TFW-F-16D
| 45TFW-F-4G | 49TFW | 49TFW-F-111F | 49TFW-F-16C

<det> → a | an | the

<conjunction> → and

<locative-preposition> → around | at | in | near | on | within

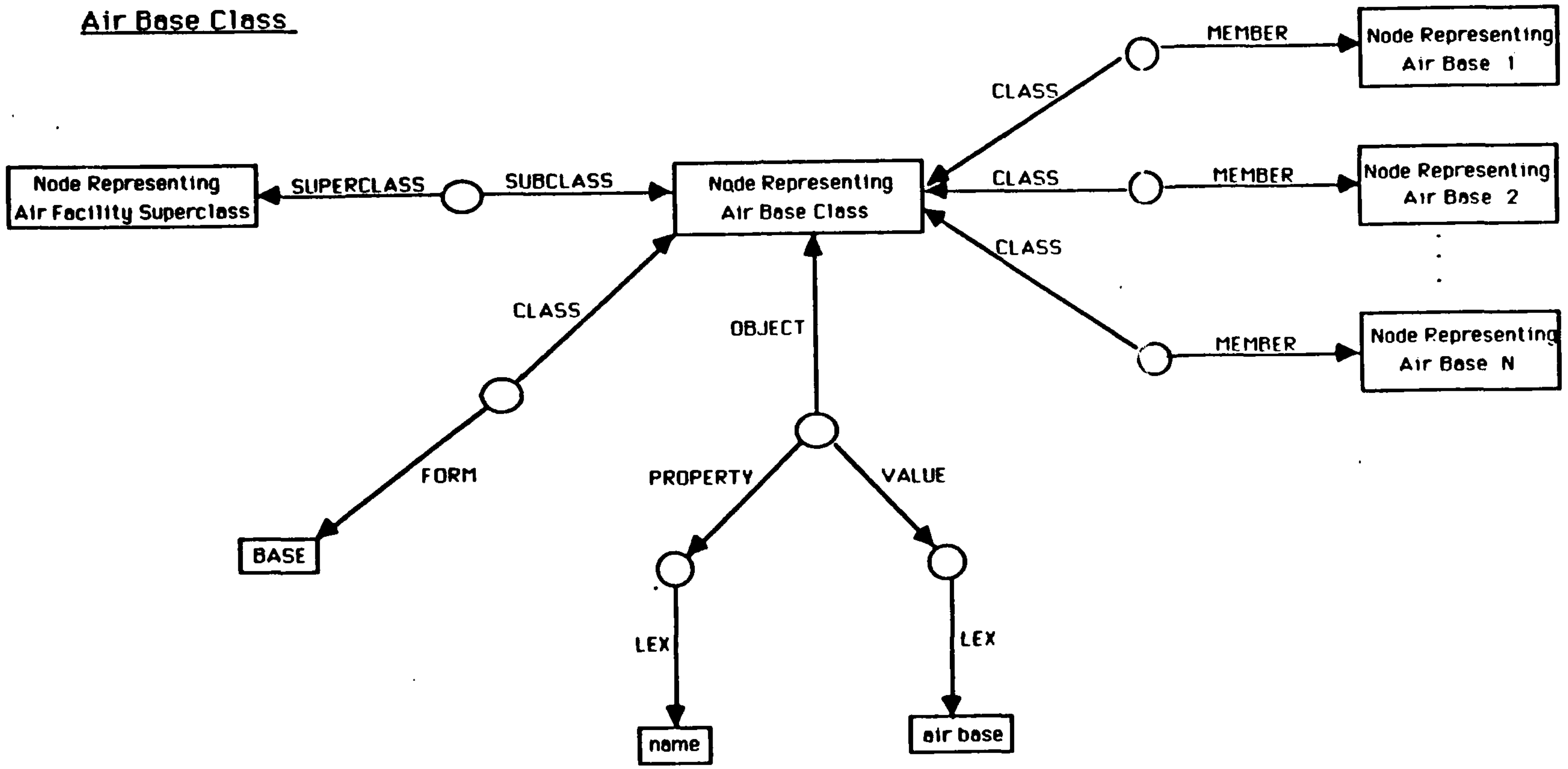
<complement-preposition> → as | by | for | from | of | to

APPENDIX C

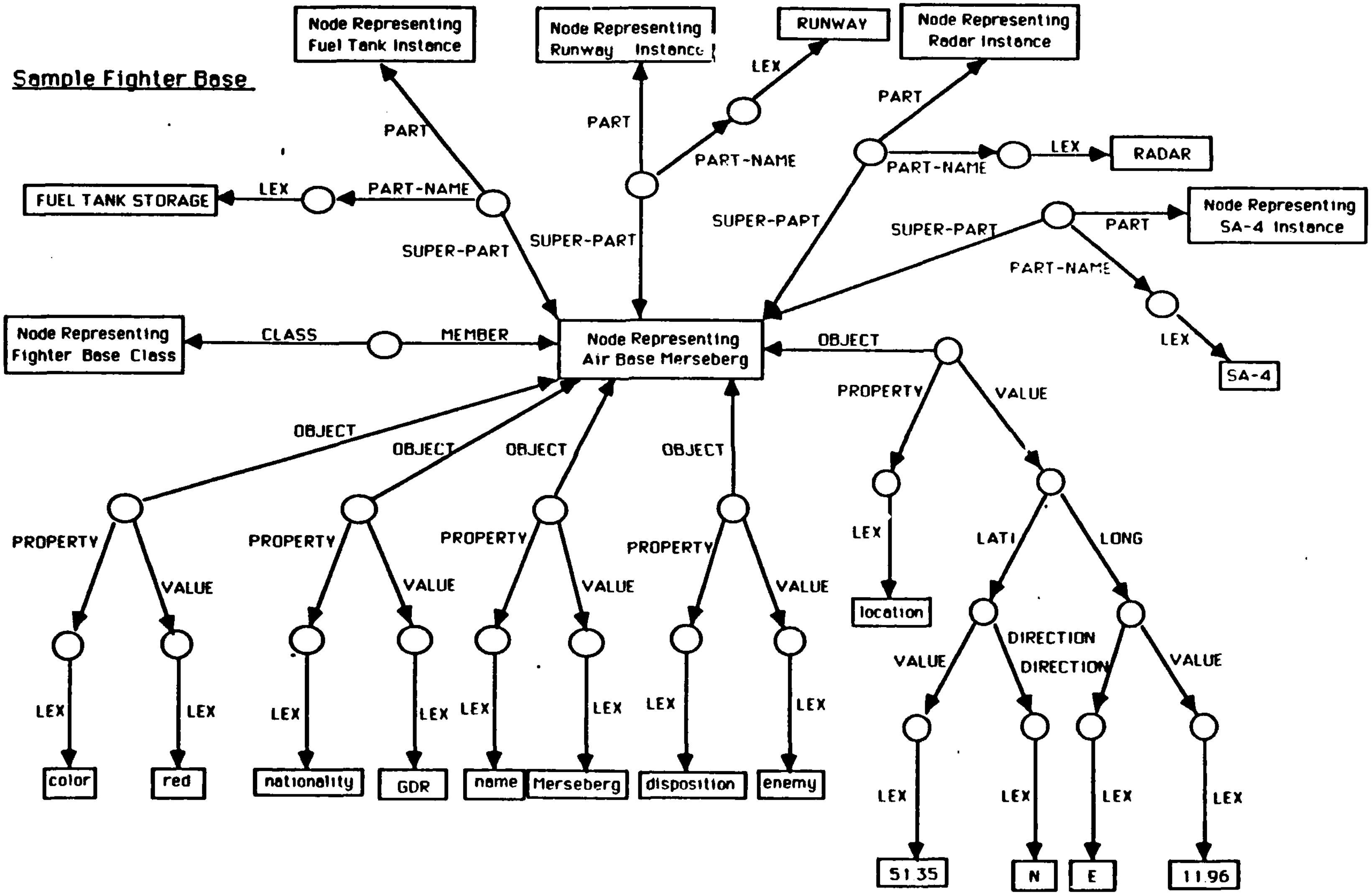
Semantic Network Knowledge Base Illustrations

This appendix contains illustrations of the SNePS representations of selected concepts from the CUBRICON knowledge base of interface and domain-specific knowledge. The CUBRICON knowledge base was discussed in Section 4.

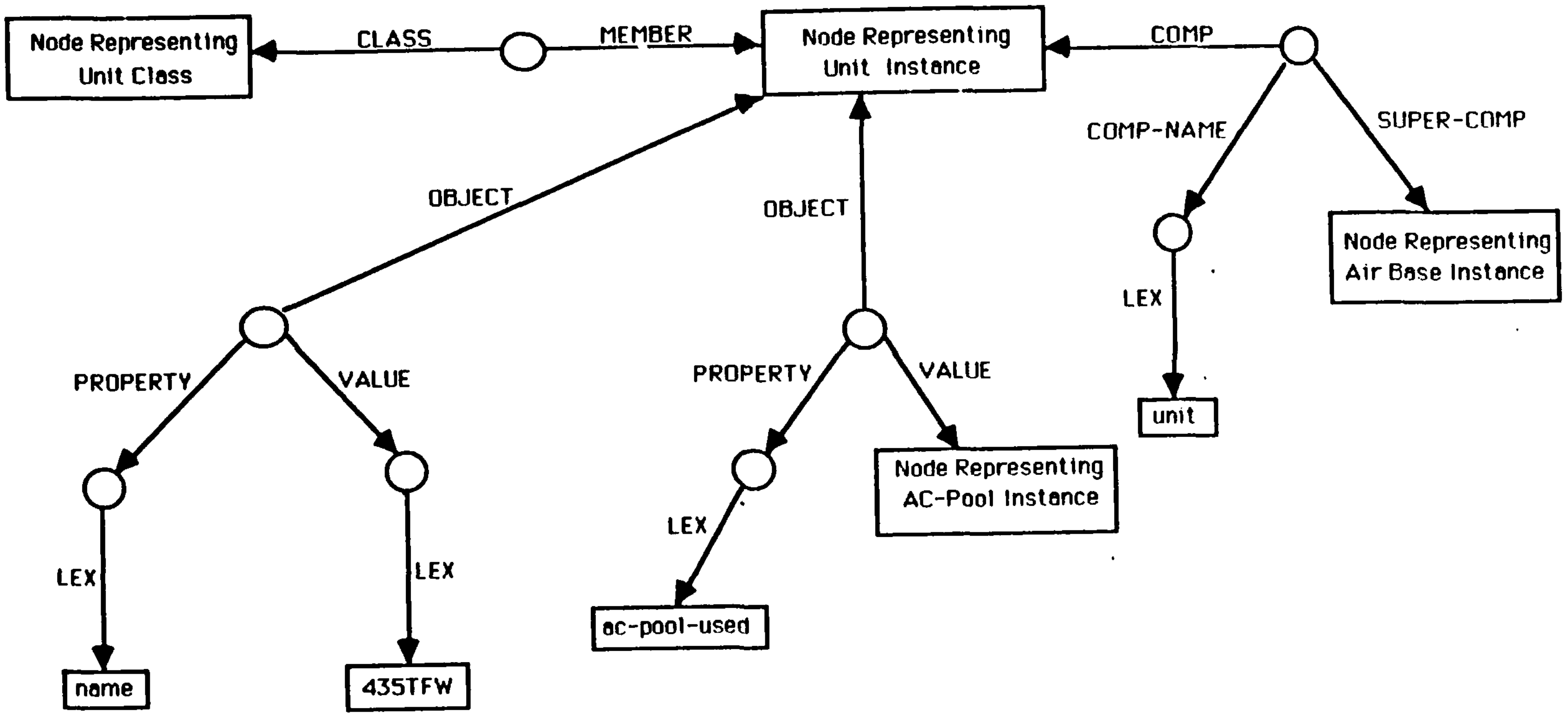
Air Base Class


MEMBER
Node Representing Air Base 1
CLASS
Node Representing Air Facility Superclass
SUPERCLASS
SUBCLASS
Node Representing Air Base Class
CLASS
MEMBER
Node Representing Air Base 2
CLASS
CLASS
OBJECT
MEMBER
Node Representing Air Base N
FORM
PROPERTY
VALUE
BASE
LEX
LEX
name
air base

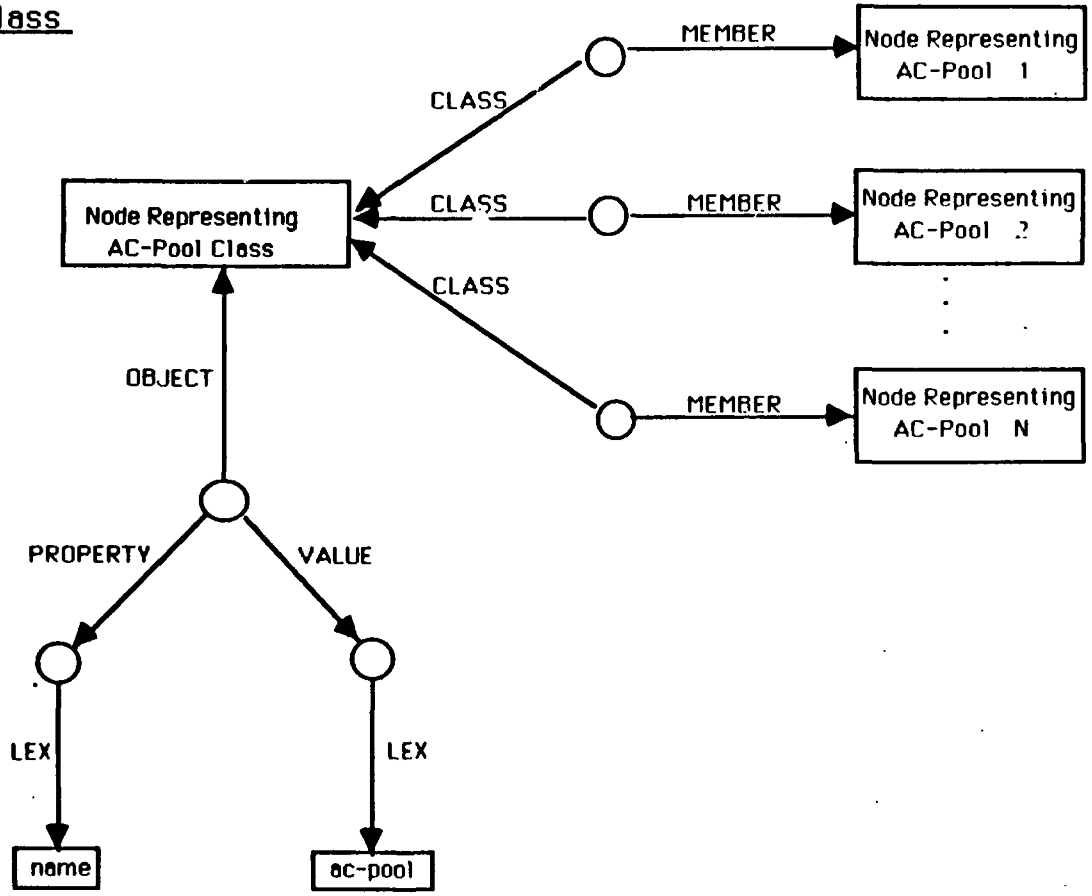
Sample Fighter Base


Node Representing Fuel Tank Instance
Node Representing Runway Instance
RUNWAY
Node Representing Radar Instance
RADAR
Node Representing SA-4 Instance
SA-4
FUEL TANK STORAGE
Node Representing Fighter Base Class
Node Representing Air Base Merseberg
PART
PART-NAME
LEX
SUPER-PART
SUPER-PAPT
CLASS
MEMBER
OBJECT
PROPERTY
VALUE
LATI
LONG
DIRECTION
location
color
red
nationality
GDR
name
Merseberg
disposition
enemy
51.35
N
E
11.96

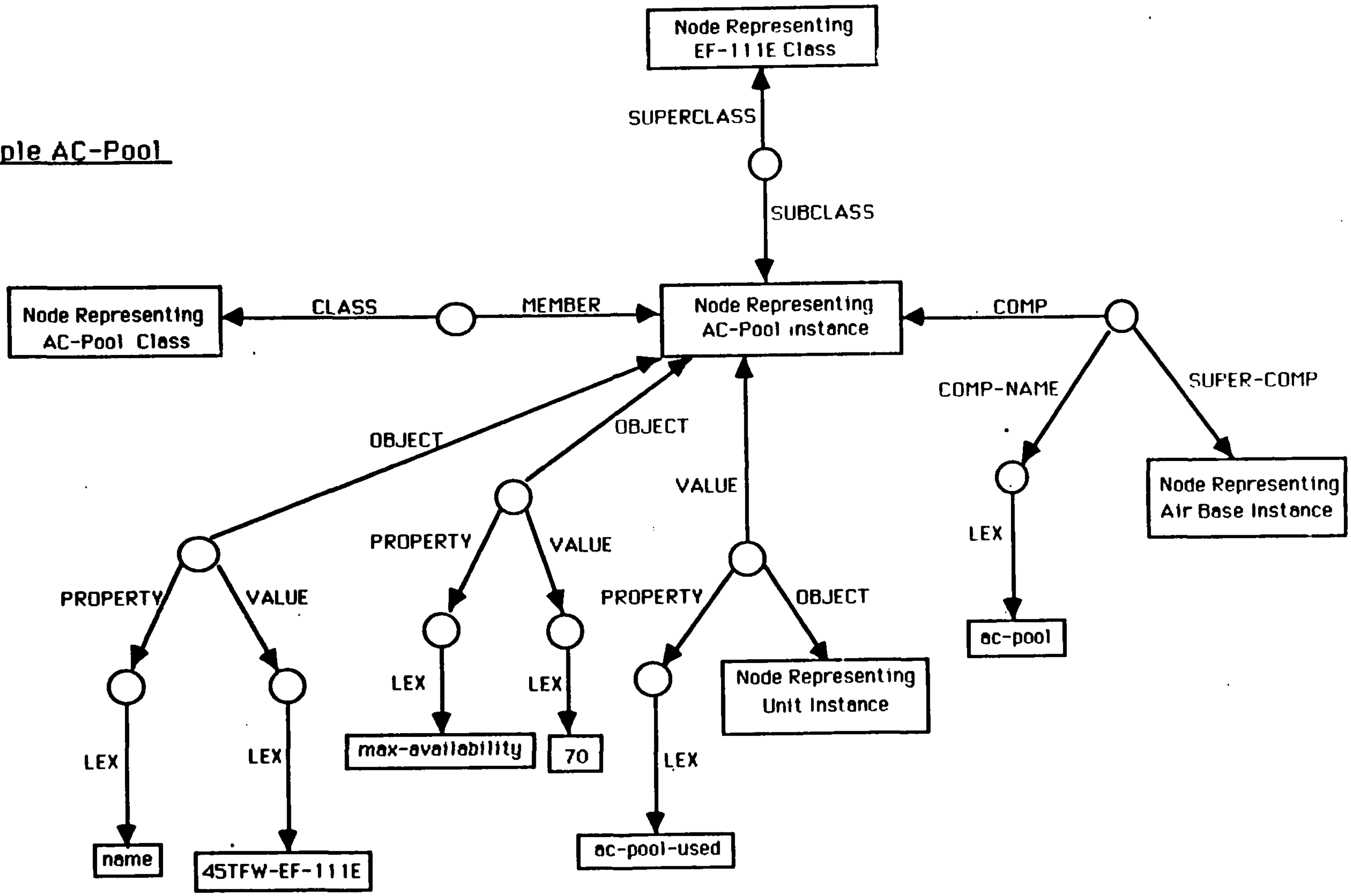
Sample Unit


Node Representing Unit Class
CLASS
MEMBER
Node Representing Unit Instance
COMP
COMP-NAME
SUPER-COMP
LEX
unit
Node Representing Air Base Instance
OBJECT
OBJECT
PROPERTY
VALUE
PROPERTY
VALUE
LEX
LEX
LEX
name
435TFW
ac-pool-used
Node Representing AC-Pool Instance

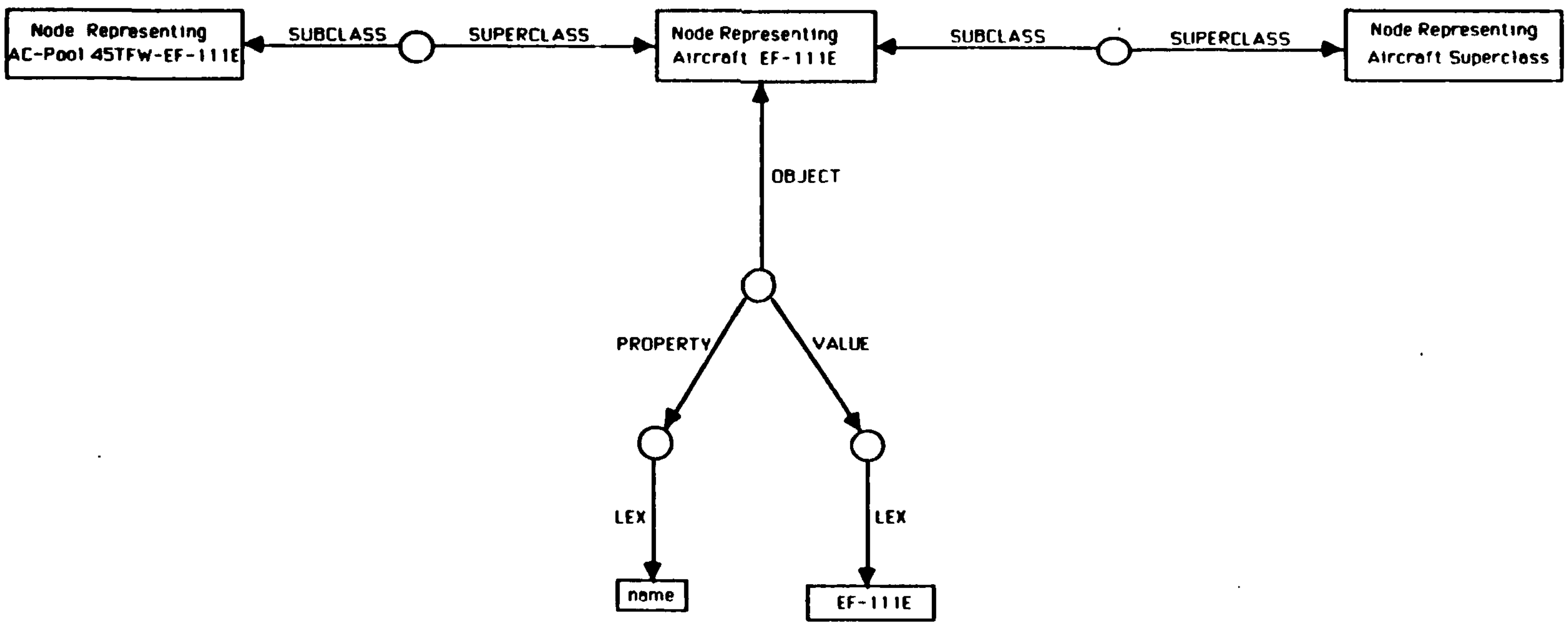
AC-Pool Class


Node Representing AC-Pool 1
Node Representing AC-Pool 2
Node Representing AC-Pool N
MEMBER
MEMBER
MEMBER
CLASS
CLASS
CLASS
Node Representing AC-Pool Class
OBJECT
PROPERTY
VALUE
LEX
LEX
name
ac-pool

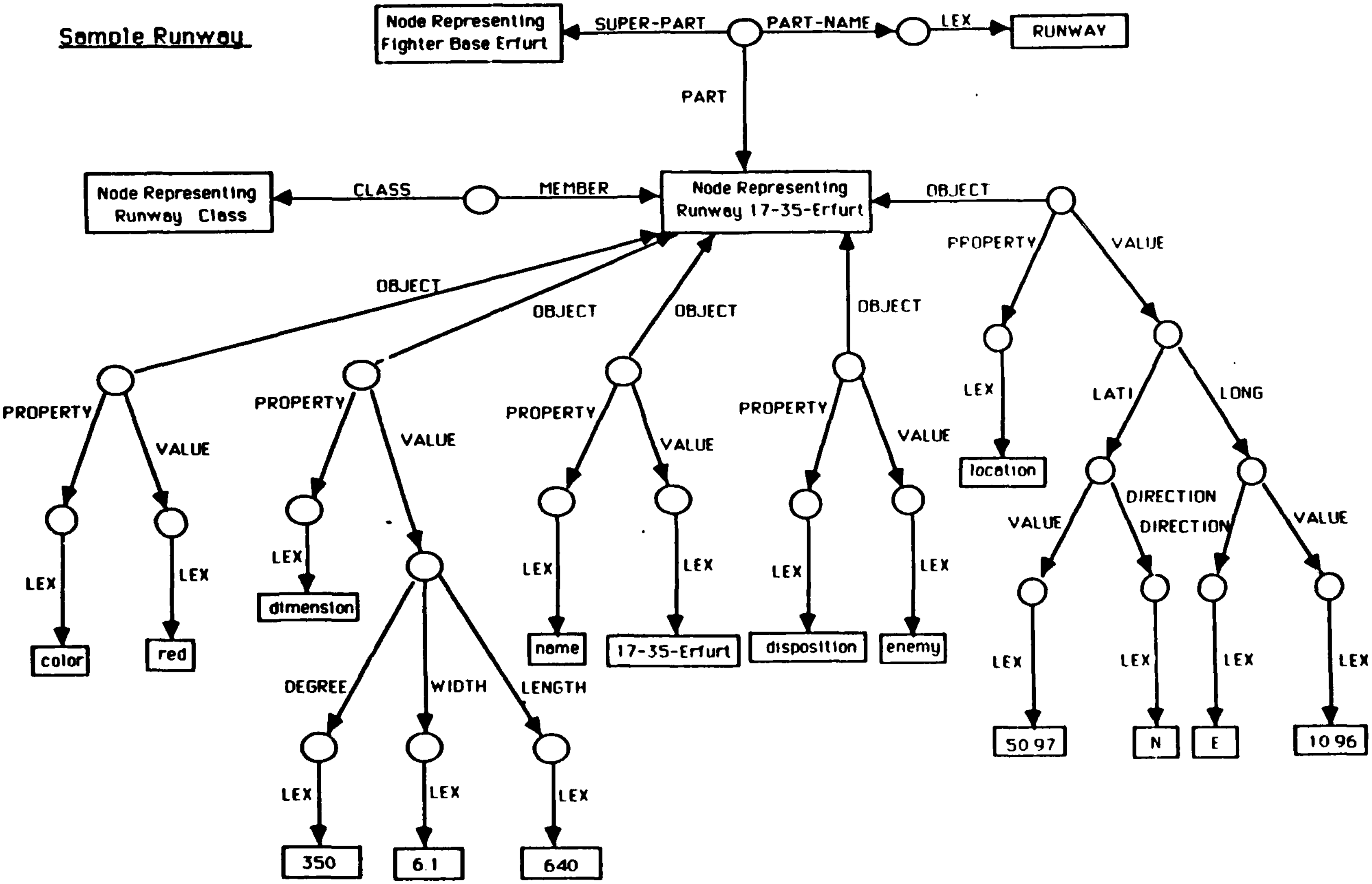
Sample AC-Pool



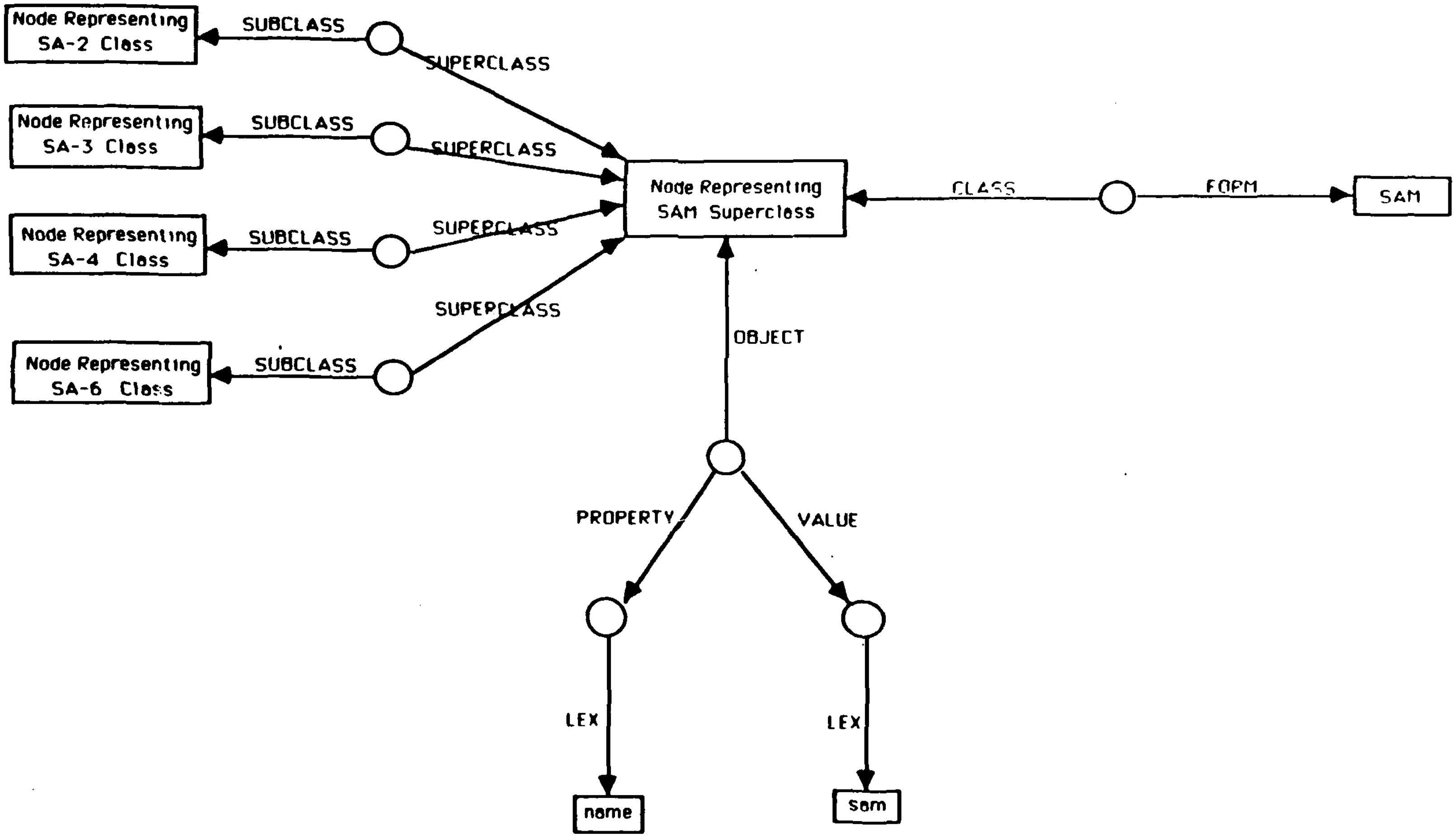
Sample Aircraft


Node Representing AC-Pool 45TFW-EF-111E
SUBCLASS
SUPERCLASS
Node Representing Aircraft EF-111E
SUBCLASS
SUPERCLASS
Node Representing Aircraft Superclass
OBJECT
PROPERTY
VALUE
LEX
LEX
name
EF-111E

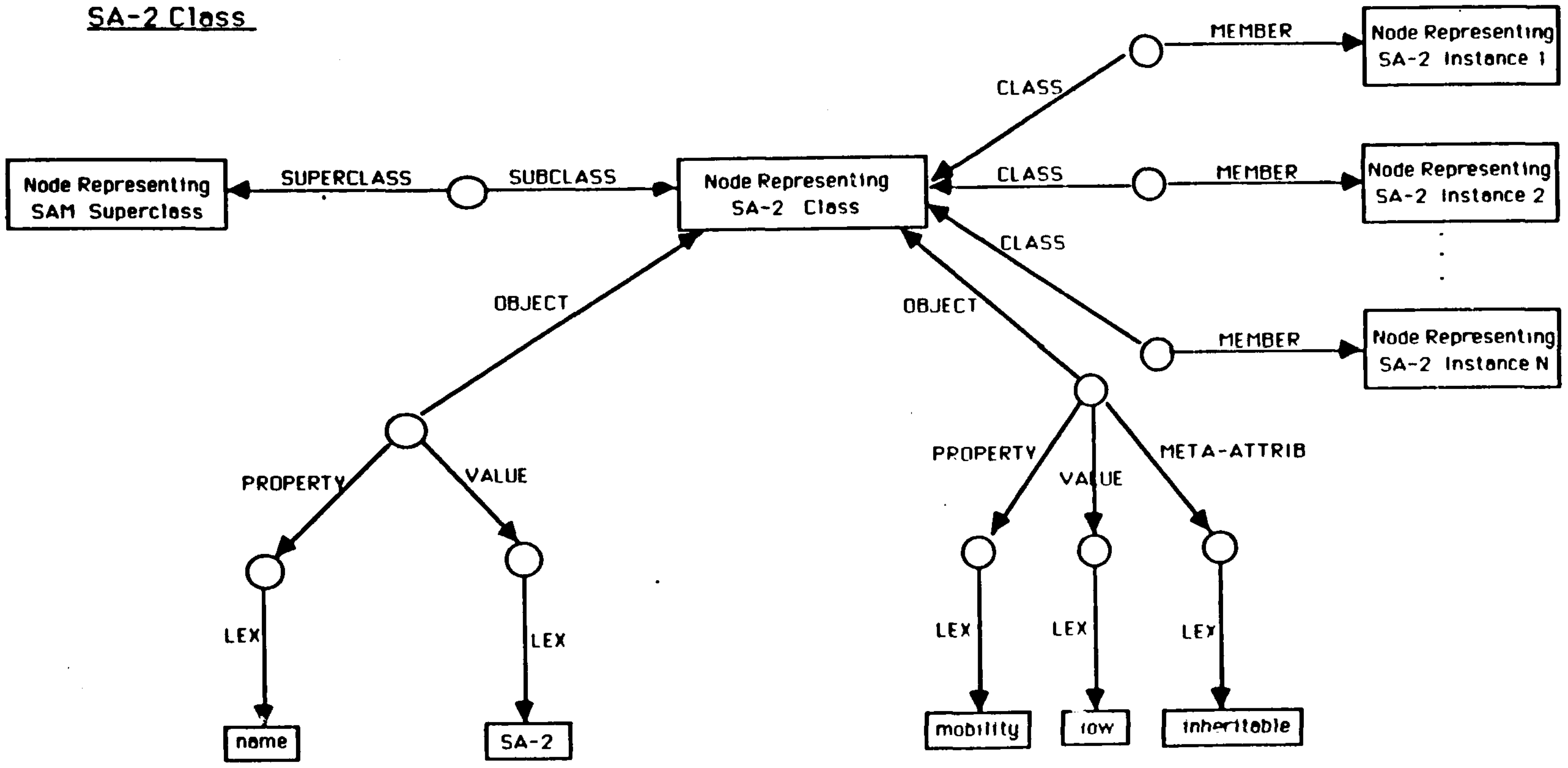
Sample Runway



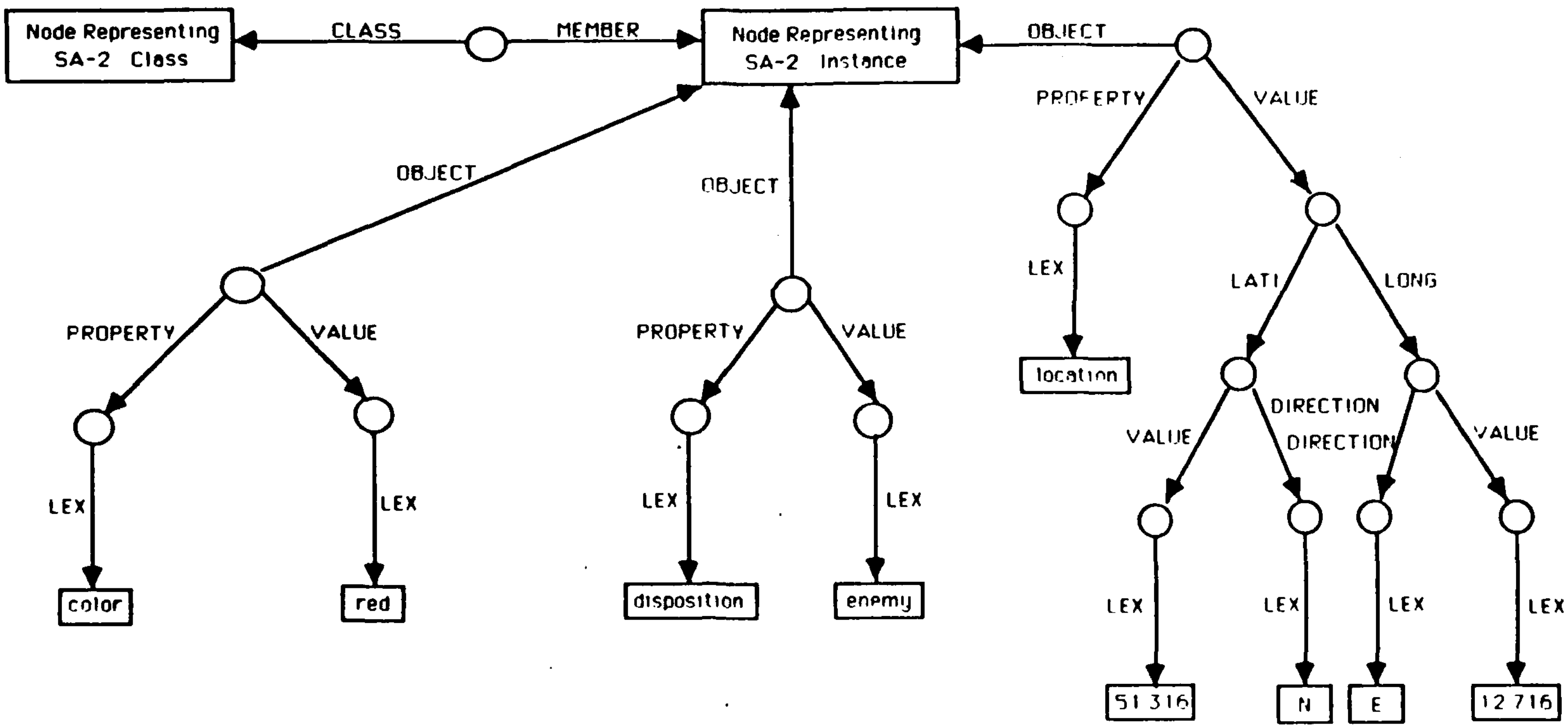
SAM Superclass


Node Representing SA-2 Class
Node Representing SA-3 Class
Node Representing SA-4 Class
Node Representing SA-6 Class
SUBCLASS
SUBCLASS
SUBCLASS
SUBCLASS
SUPERCLASS
SUPERCLASS
SUPERCLASS
SUPERCLASS
Node Representing SAM Superclass
CLASS
FORM
SAM
OBJECT
PROPERTY
VALUE
LEX
LEX
name
sam

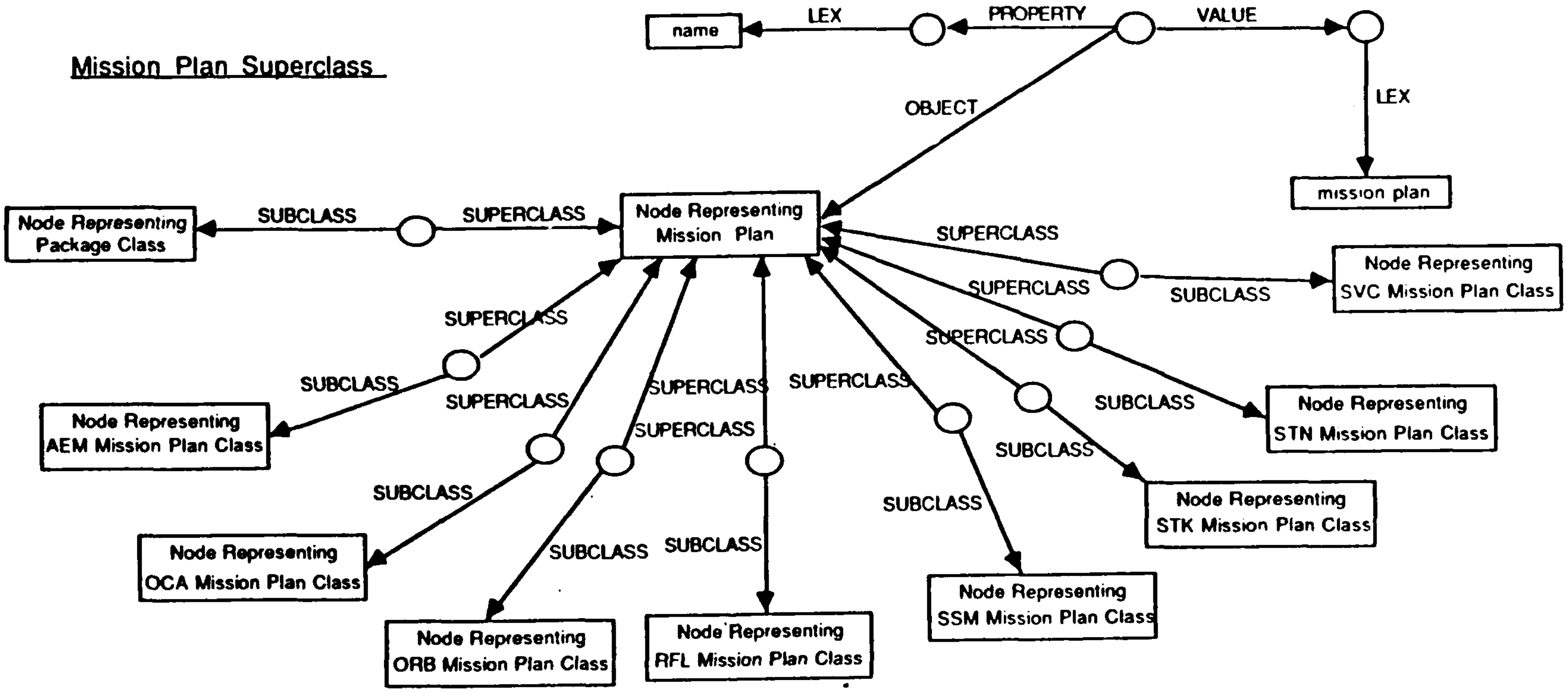
SA-2 Class


Node Representing SAM Superclass
SUPERCLASS
SUBCLASS
Node Representing SA-2 Class
CLASS
CLASS
CLASS
MEMBER
MEMBER
MEMBER
Node Representing SA-2 Instance 1
Node Representing SA-2 Instance 2
Node Representing SA-2 Instance N
OBJECT
PROPERTY
VALUE
LEX
LEX
name
SA-2
OBJECT
PROPERTY
VALUE
META-ATTRIB
LEX
LEX
LEX
mobility
low
inheritable

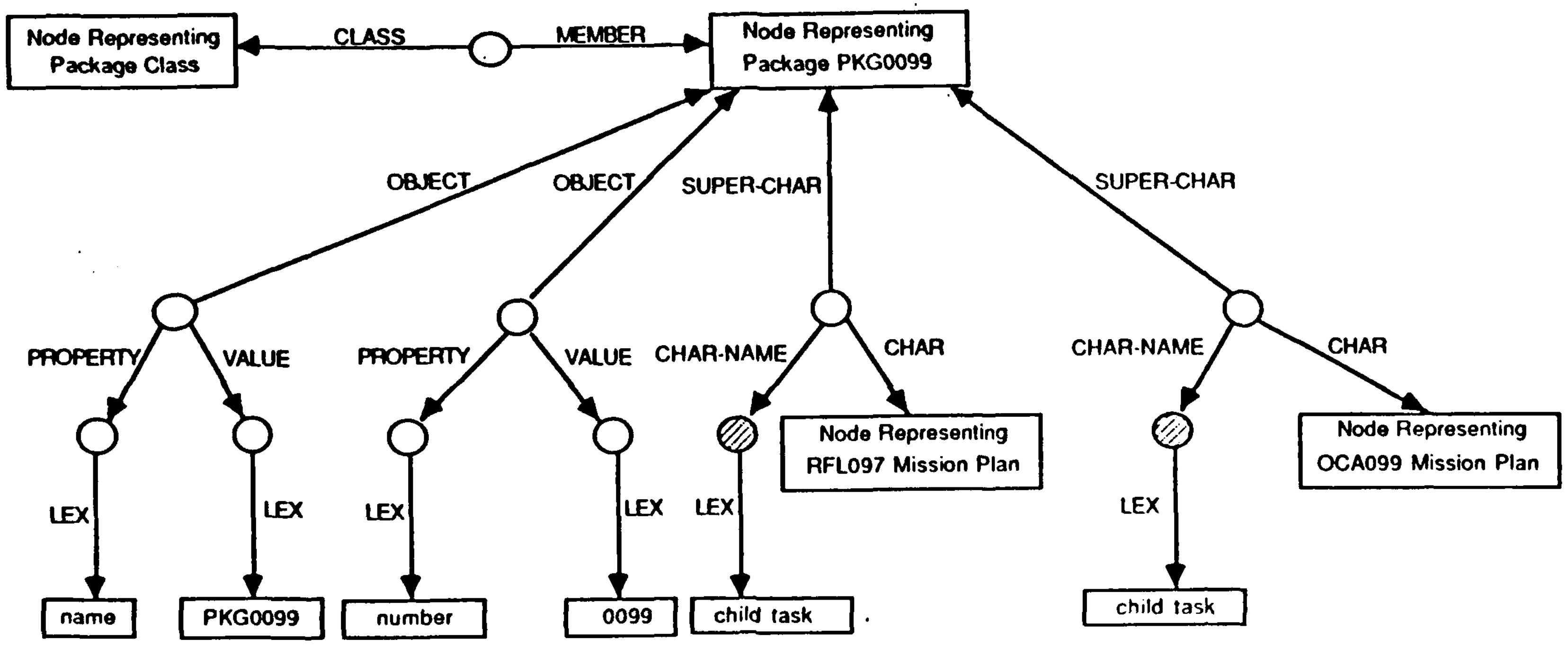
Sample SA-2


Node Representing SA-2 Class
CLASS
MEMBER
Node Representing SA-2 Instance
OBJECT
PROPERTY
VALUE
LEX
location
LATI
LONG
DIRECTION
DIRECTION
VALUE
VALUE
LEX
LEX
LEX
LEX
51 316
N
E
12 716
OBJECT
PROPERTY
VALUE
LEX
LEX
disposition
enemy
OBJECT
PROPERTY
VALUE
LEX
LEX
color
red

Mission Plan Superclass

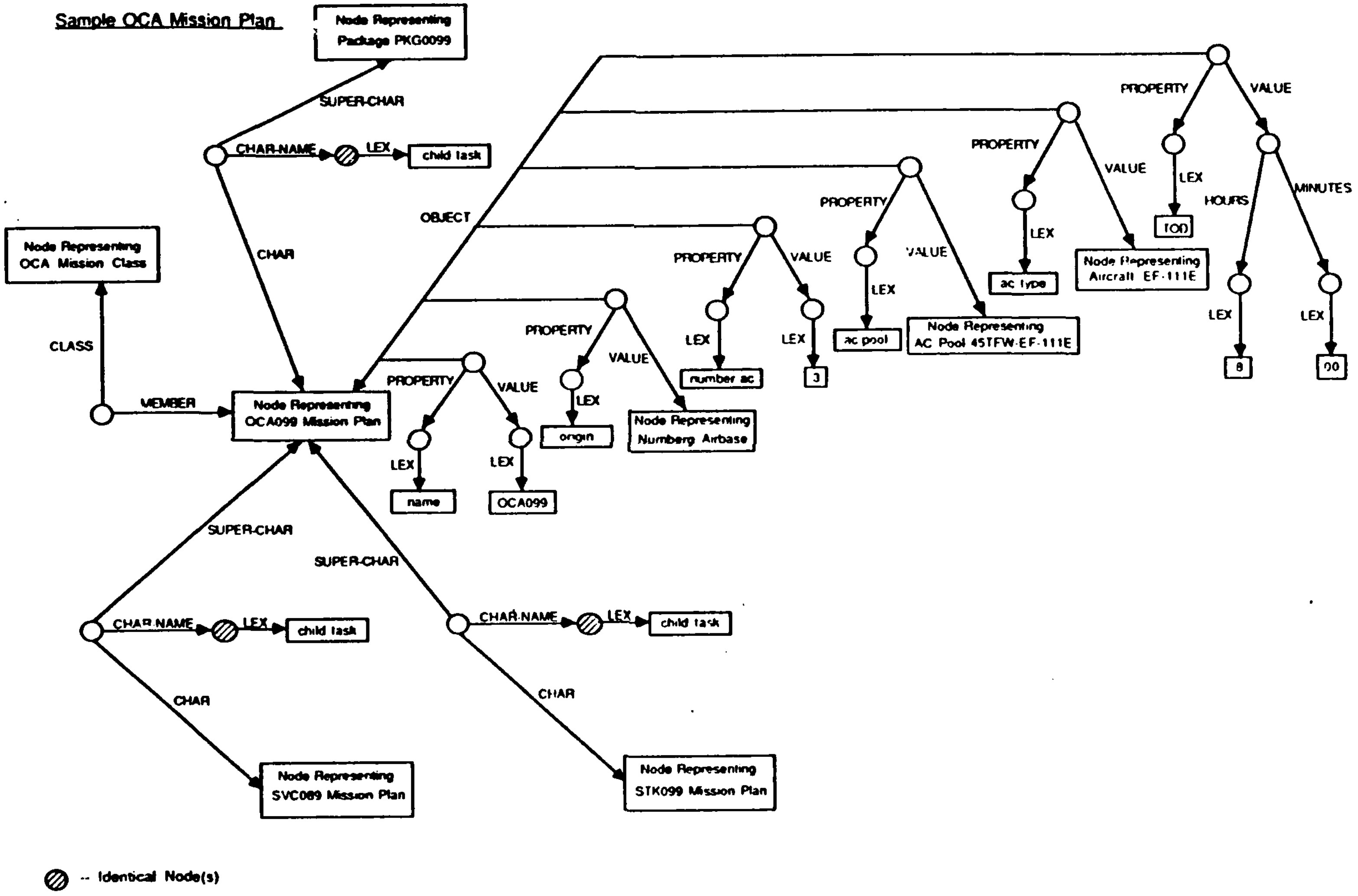

name
LEX
PROPERTY
VALUE
LEX
mission plan
OBJECT
Node Representing Mission Plan
SUBCLASS
SUPERCLASS
Node Representing Package Class
SUPERCLASS
SUBCLASS
Node Representing SVC Mission Plan Class
SUPERCLASS
SUBCLASS
Node Representing STN Mission Plan Class
SUPERCLASS
SUBCLASS
Node Representing STK Mission Plan Class
SUPERCLASS
SUBCLASS
Node Representing SSM Mission Plan Class
SUPERCLASS
SUBCLASS
Node Representing RFL Mission Plan Class
SUPERCLASS
SUBCLASS
Node Representing ORB Mission Plan Class
SUPERCLASS
SUBCLASS
Node Representing OCA Mission Plan Class
SUPERCLASS
SUBCLASS
Node Representing AEM Mission Plan Class

Sample Package



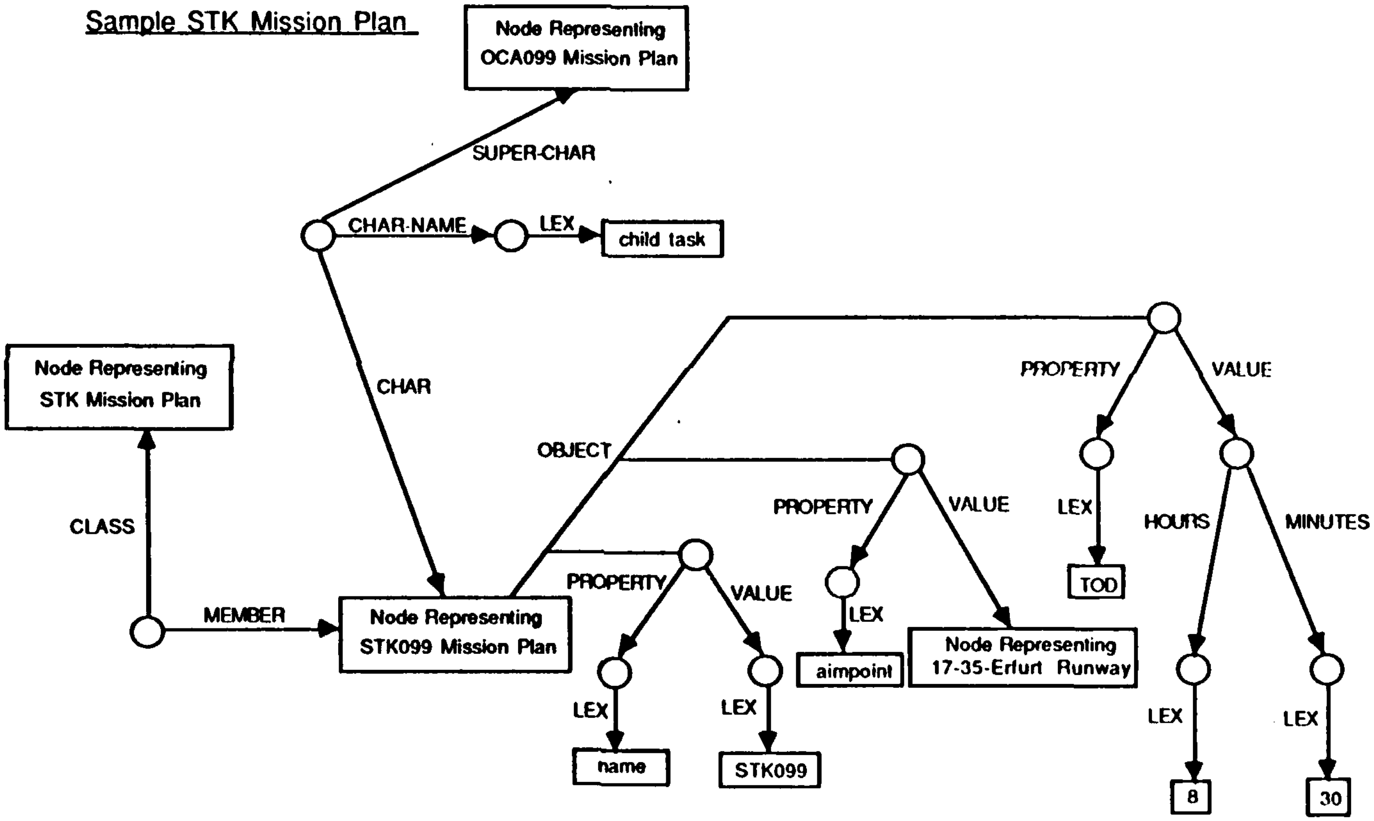
▨ -- Identical Node(s)

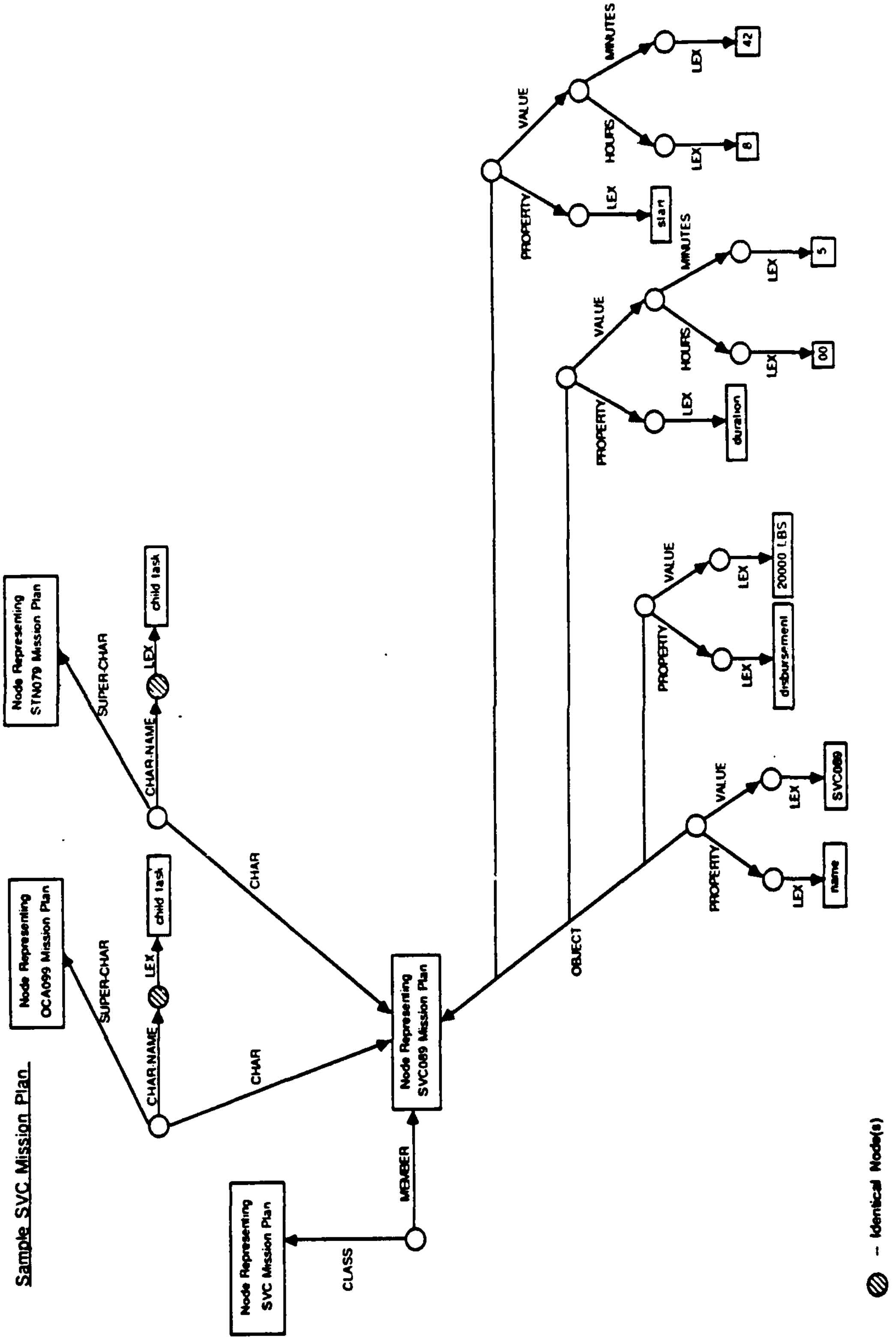
Sample OCA Mission Plan



Node Representing Package PKG0099
SUPER-CHAR
CHAR-NAME
LEX
child task
CHAR
Node Representing OCA Mission Class
CLASS
MEMBER
Node Representing OCA099 Mission Plan
OBJECT
PROPERTY
VALUE
LEX
name
OCA099
origin
Node Representing Numberg Airbase
number ac
3
ac pool
Node Representing AC Pool 45TFW-EF-111E
ac type
Node Representing Aircraft EF-111E
TOT
HOURS
MINUTES
8
00
SUPER-CHAR
CHAR-NAME
child task
CHAR
Node Representing SVC089 Mission Plan
Node Representing STK099 Mission Plan
▨ -- Identical Node(s)

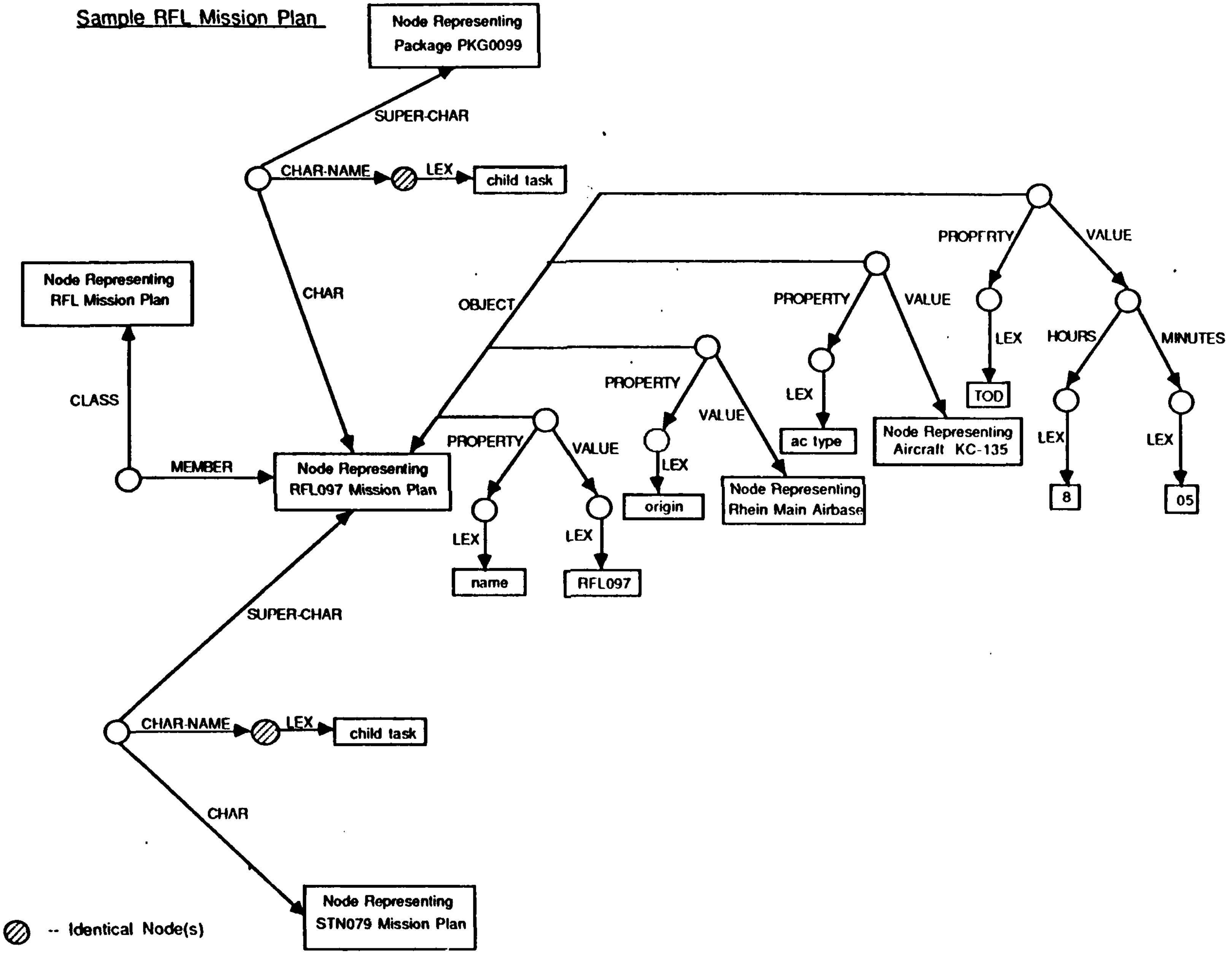
Sample STK Mission Plan



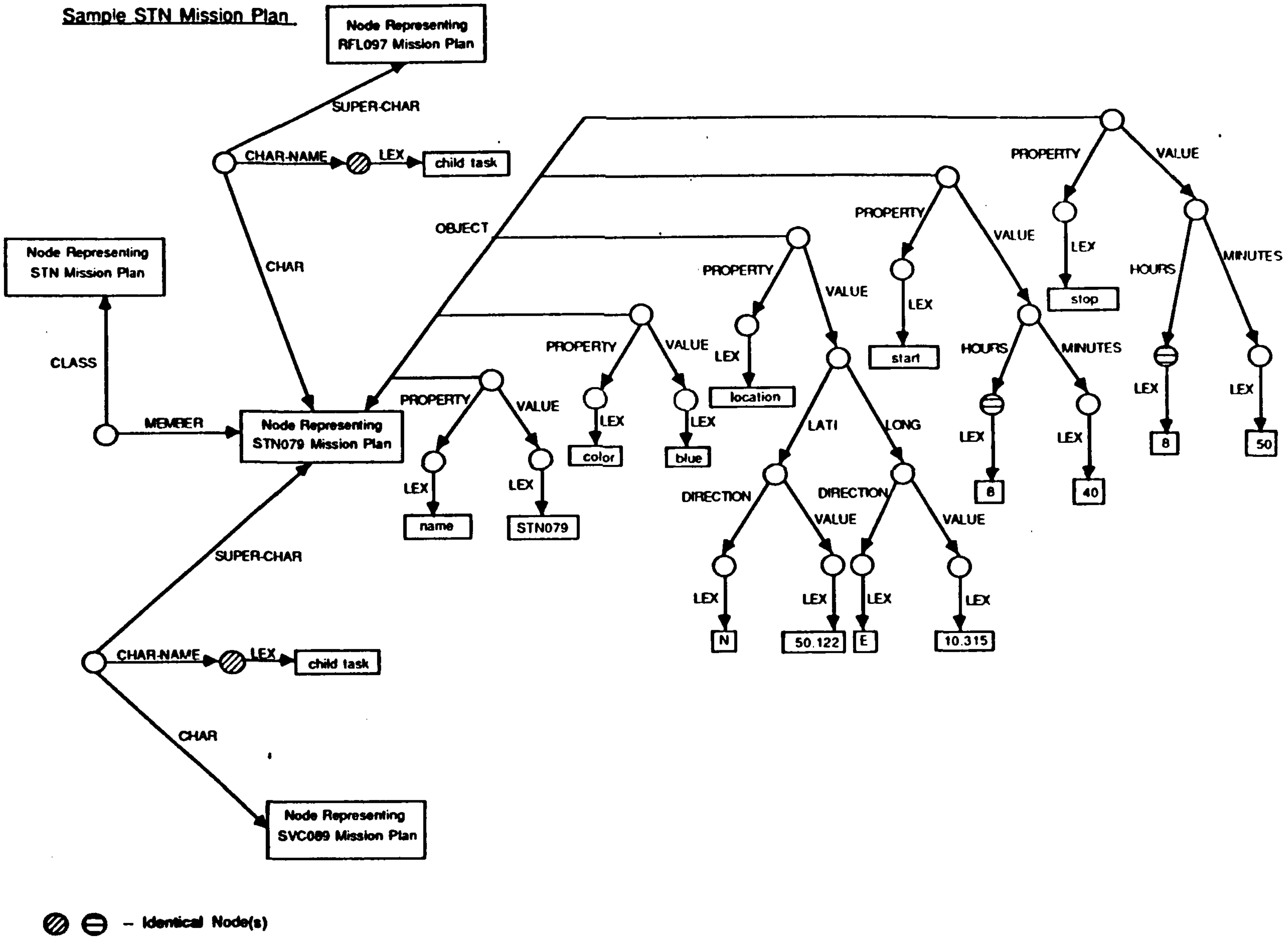


Sample SVC Mission Plan

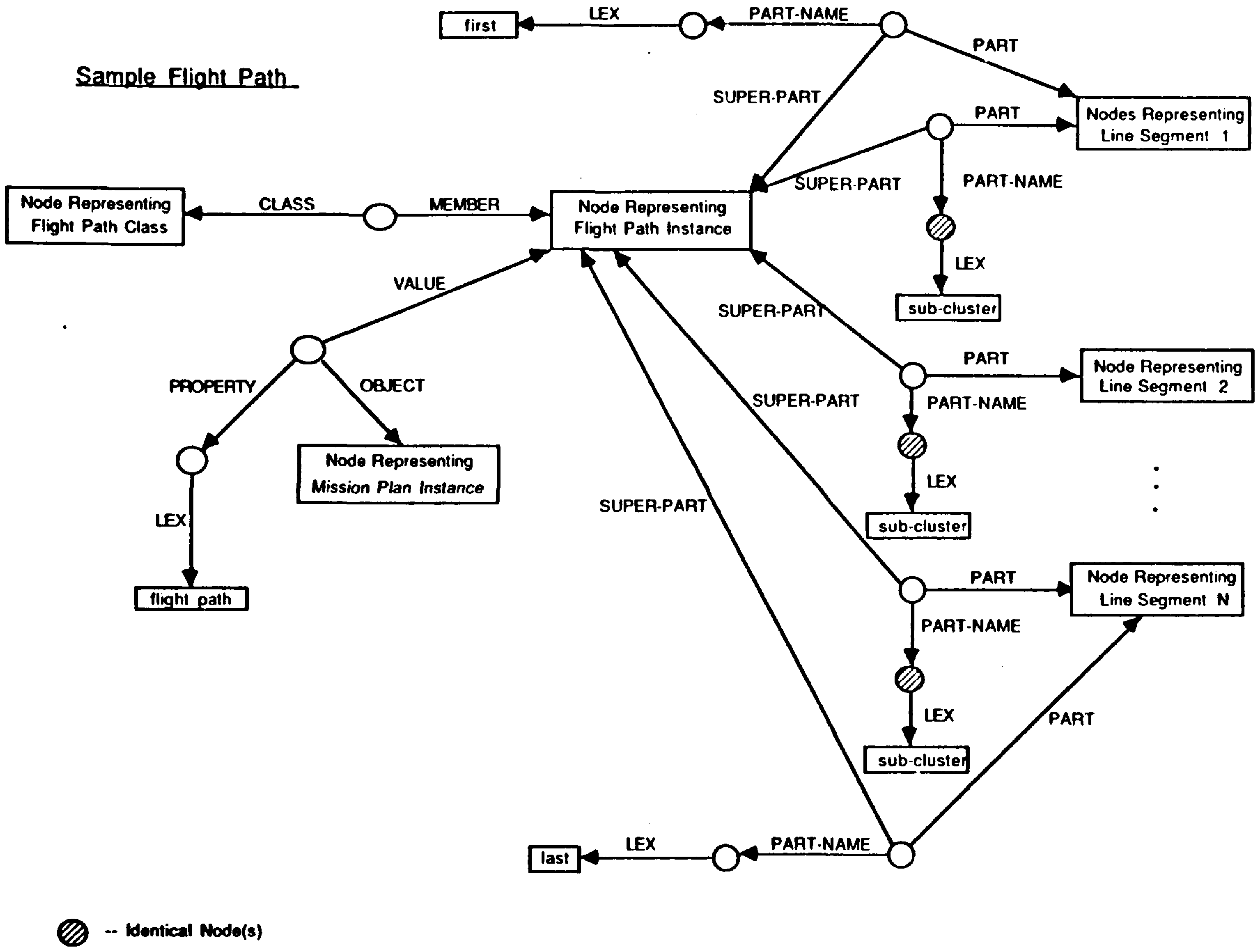
Sample RFL Mission Plan


Node Representing Package PKG0099
SUPER-CHAR
CHAR-NAME
LEX
child task
CHAR
Node Representing RFL Mission Plan
CLASS
MEMBER
Node Representing RFL097 Mission Plan
OBJECT
PROPERTY
VALUE
LEX
name
RFL097
origin
Node Representing Rhein Main Airbase
ac type
Node Representing Aircraft KC-135
PROPFRTY
TOD
HOURS
MINUTES
8
05
SUPER-CHAR
CHAR-NAME
LEX
child task
CHAR
Node Representing STN079 Mission Plan
-- Identical Node(s)

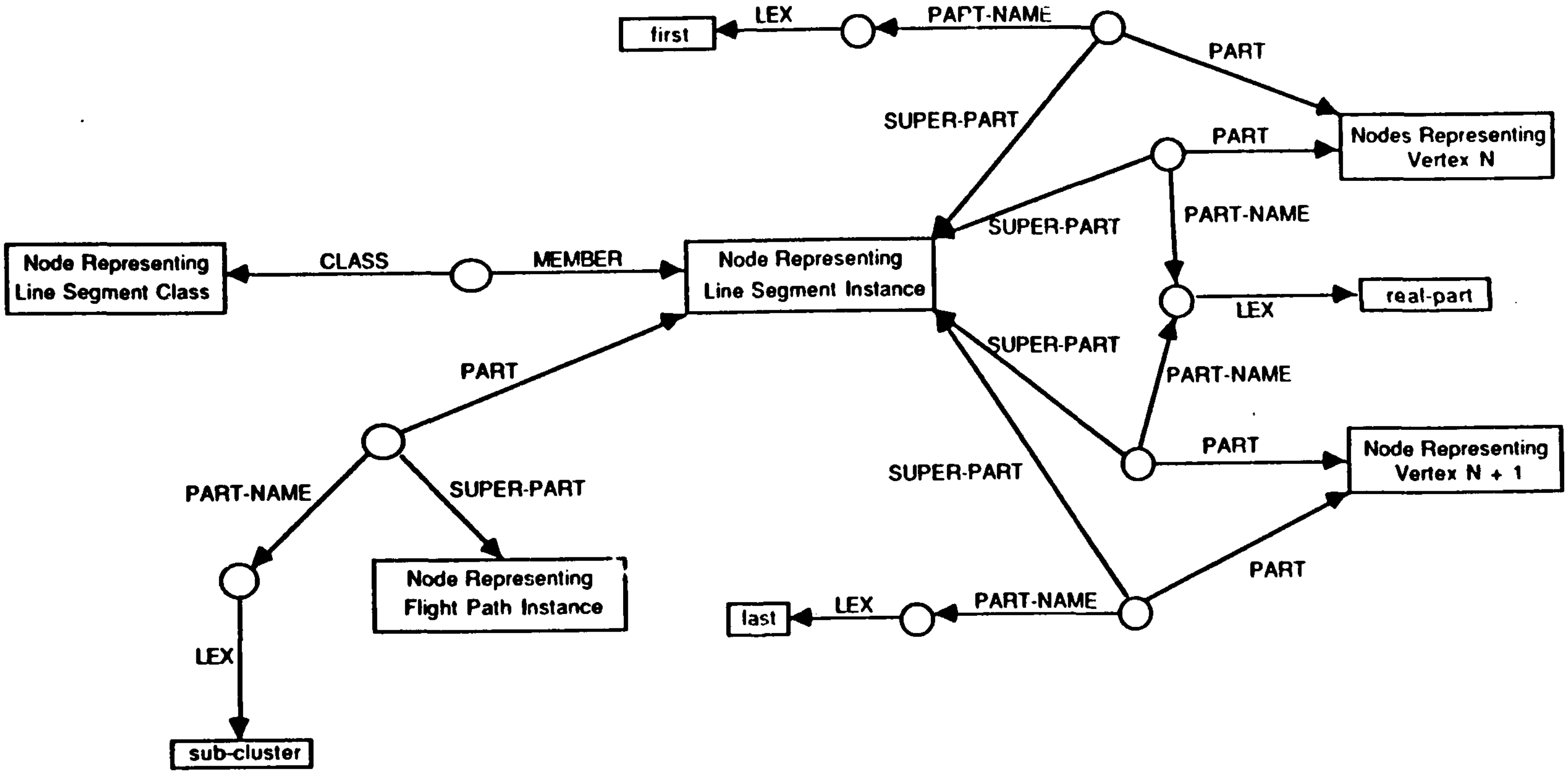
Sample STN Mission Plan



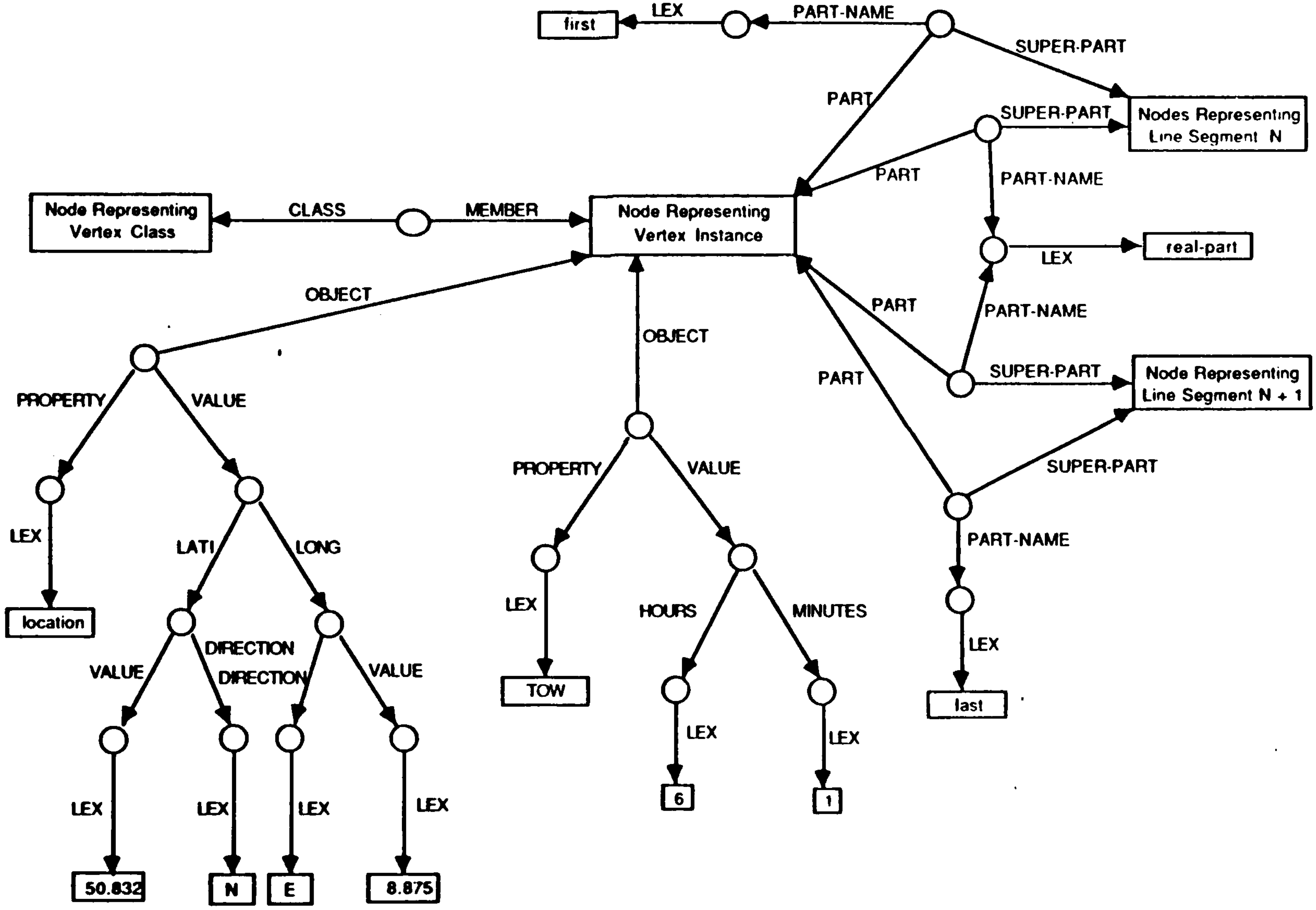
Sample Flight Path



Sample (Intermediate) Line Segment


first
LEX
PAPT-NAME
PART
SUPER-PART
PART
Nodes Representing Vertex N
SUPER-PART
PART-NAME
Node Representing Line Segment Class
CLASS
MEMBER
Node Representing Line Segment Instance
LEX
real-part
SUPER-PART
PART-NAME
PART
PART
Node Representing Vertex N + 1
PART-NAME
SUPER-PART
SUPER-PART
Node Representing Flight Path Instance
PART
LEX
PART-NAME
last
LEX
sub-cluster

Sample (Intermediate) Vertex



APPENDIX D

References to Published Technical Papers Describing CUBRICON and the Research Conducted Under the Intelligent Multi-Media Interfaces Project

The following is a list of CUBRICON related technical papers that were published in technical journals and conference proceedings. These papers were written and presented through the support of the Intelligent Integrated Multi-Media Interfaces Project.

- Neal, J.G., Shapiro, S.C., & Smith, Y. 1987. Intelligent Integrated Interface Technology, *Proceedings of the 1987 Tri-Service Data Fusion Symposium*, JHU-APL, Laurel, MD.
- Neal, J.G., Bettinger, K.E., Byoun, J.S., Dobes, Z., & Thielman, C.Y. 1988. An Intelligent Multi-Media Human-Computer Dialogue System, *Proceedings of the Workshop on Space Operations, Automation, and Robotics (SOAR-88)*, Wright State University, Dayton, OH.
- Neal, J.G., Dobes, Z., Bettinger, K.E., & Byoun, J.S. 1988. Multi-Modal References in Human-Computer Dialogue, *Proceedings of AAAI-88*, pages 819-823.
- Neal, J.G. & Shapiro, S.C. 1988. Intelligent Multi-Media Interface Technology, *Architectures for Intelligent Interfaces: Elements and Prototypes*, J.W. Sullivan & S.W. Tyler (eds.), Addison-Wesley Pub. Co., pages 69-91.
- Neal, J.G., Thielman, C.Y., Funke, D.J., & Byoun, J.S. 1989. Multi-Modal Output Composition for Human-Computer Dialogues, *Proceedings of the 1989 Artificial Intelligence Systems in Government Conference*, Wash. D.C., pages 250-257.
- Neal, J.G., Thielman, C.Y., Dobes, Z., Haller, S.M., & Shapiro, S.C. 1989. Natural Language with Integrated Deictic and Graphic Gestures. *Proc. of the 1989 DARPA Workshop on Speech and Natural Language*, Harwich Port, MA, Morgan Kaufmann Publishers, pages. 410-423.

- Neal, J.G., Thielman, C.Y., Dobes, Z., Haller, S.M., Glanowski, S., & Shapiro, S.C. 1989. CUBRICON: A Multi-Modal User Interface. *Proc. of the GIS/LIS '89 Conference*, Orlando, FL.
- Neal, J.G., Thielman, C.Y., Lammens, J., & Shapiro, S.C. 1990. *CUBRICON: A Knowledge-Based Multi-Modal Interface System*, Journal paper in preparation.

APPENDIX E

Evaluation Training Material and Results

This section contains material used to support the CUBRICON evaluation and also contains data generated during the evaluation. The CUBRICON evaluation support material includes training material and CUBRICON work aids.

E.1 Training Material and Work Aids

A few of the materials used to train the Evaluators in the use of CUBRICON, the evaluation goals, and evaluation procedures are included in this section. The training materials included are:

- Training Outline
- Voice Training List (used to guide Evaluators through voice recognition system training)
- CUBRICON grammar (see Appendix A)
- Evaluator Script (script used for guiding the evaluation which exercised all important CUBRICON features)

E.2 Completed Air Force Application-Oriented Evaluation Questionnaire

This sections contains the completed Air Force Application-Oriented Evaluation Questionnaire.

E.3 Completed Human Factors Evaluation Questionnaire and Checklist

This sections contains the completed Human Engineering Evaluation Questionnaire and Checklist. It includes both the top-level summary portions as well as the more detailed and supporting checklist item responses. The numbered items within the evaluation categories are cross-referenced to the top-level project goals from the SOW. The results of the evaluation organized according to SOW goals are available upon request from the authors.

Training Outline

This Appendix contains an outline of training to be provided in support of the CUBRICON evaluations. The training is virtually identical for both stages of the evaluation program. Items that pertain to one or the other stage exclusively are noted parenthetically.

Lesson Number: 1

Title: Introduction

Contents:

- Purpose and design philosophy
  - Application independant.
  - Multi-media input and output.
  - AI-based.
  - Research tool.
  - Integrated system (use desired modality)
- Purpose of evaluation
  - Assess whether CUBRICON meets current requirements.
  - Assess viability of design approach.
  - Make recommendations for improvment (includes suggestions for fine tuning to recommendations for altered design approach).
- Test approach and schedule
  - Two evaluation stages (engineering evaluation and naive user test).
  - Test schedule.
  - Guidelines for applying the evaluation checklist (Stage 1 only).
  - Guidelines for applying evaluation script (Stage 1 only).
  - Guidelines for free-form evaluation (i.e., fully explore evaluation items and stress system to identify weaknesses) (Stage 1 only).
  - Description of the sample problem and guidelines for working its solution (Stage 2 only).

  - System is slow, expect delays.

Lesson Number: 2

Title: The Baseline Application

Handout: Overview description of Application Data/Knowledge Base

Contents:

- Overview of application area
  - Types of tactical planning supported
  - Specific tasks supported (e.g., querying and assessing enemy assets, designating targets, locating and assigning friendly assets, coordinating interdependent missions, etc.)
- The Application Data/Knowledge Base
  - Purpose.
  - Structure and general contents.
  - Specific information available.
  - Review form.
  - Breakdown of Packages.

Lesson Number: 3

Title: Interactive Features

Contents:

- Describe and demonstrate input techniques (with student hands-on).
  - Mouse.
  - Voice.
  - Text.
  - Forms.

  - Combinations of above.

- Describe and demonstrate output features (with student hands-on).
  - Automatic windowing approach (hybrid tiled/overlapping layout, used window bin, user override).
  - Map presentation system.
    - Symbols and icons.
    - The map area (limits).
    - Color code.
    - Map features (e.g., roads, rivers, cities, etc.).
    - Labeling (e.g., of icons, pointing boxes, text boxes, etc.).
  - Text windows.
  - Tables and forms.
  - Voice outputs.

- Describe and demonstrate interactive control (with student hands-on).
  - Sample commands available (e.g., zoom in, request information about icons, query data base).
  - Range of command techniques (e.g., options for combining media, examples of different approaches to obtain the same information).
  - Flight path generation and subsequent presentation.

Lesson Number: 4

Title: **Natural Language (NL) Input with Coordinated Pointing Gestures**

Handouts: **Grammar specification, catagorized vocabulary list**

Contents:

- Overview of NL input via voice recognition system and/or keyboard with mouse point gestures
- Grammar available.
- Vocabulary available.

Lesson Number: 5

Title: Interactive Practice

Contents:

- Supervised practice using vocabulary and grammar together with mouse, keyboard, and various displays.

Lesson Number: 6

Title: Summary/Question and Answer

Contents:

- Present course outline as review.
- Entertain questions.

Voice Training List

Vocabulary Word for Training	Phonetic Definition	Comment
a	"a"	
ac-pool	"ay-cee-pool"	Aircraft pool or unit at an airbase.
ac-pools	"ay-cee-pools"	Aircraft pools or units at an airbase.
all	"all"	
am	"am"	
an	"an"	
and	"and"	
allstedt	"Allstedt"	Name of city and airbase in E. Germany.
aimpoints	"aimpoints"	
airbase	"airbase"	
any	"any"	
are	"are"	
around	"around"	
assign	"assign"	
at	"at"	
battalion	"battalion"	
battalions	"battalions"	
blink	"blink"	
by	"by"	
cieling	"cieling"	
color-graphics-display	"color graphics display"	
current	"current"	
display	"display"	
dresden	"Dresden"	Name of city and airbase in E. Germany.
duration	"duration"	
enter	"enter"	Enter input for evaluation (at the end of a sentence).
enterq	"enter"	Enter input for evaluation (at the end of a question).
enterverb	"enter"	
erfurt	"erfurt"	
ew-germany	"east-west germany"	Name of region that can be displayed by CUBRICON.
expose	"expose"	
factory	"factory"	
fg	"ef-gee"	Short for Fulda Gap, the name of a region that can be displayed by CUBRICON.
fighter	"fighter"	
flight	"flight"	
fourty-fifth-tfw-ef-111e	"forty fifth tee-ef-dub'l-u ee-ef one-eleven-ee"	45th Tactical Fighter Wing of EF-111Es.
fourty-ninth-tfw	"forty ninth tee-ef-dub'l-u"	49th Tactical Fighter Wing.
fourty-ninth-tfw-f-111f	"forty ninth tee-ef-dub'l-u ef one eleven-ef"	49th Tactical Fighter Wing of F-111Fs.
for	"for"	
form	"form"	
forms	"forms"	

Franz	"Franz"	
from	"from"	
fuel	"fuel"	
fulda-gap	"fulda gap"	Fulda Gap, name of region that can be displayed by CUBRICO
here	"here"	
highlight	"highlight"	
in	"in"	
is	"is"	
it	"it"	
heliport	"heliport"	
heliports	"heliports"	
kc-135	"kay-cee one-thirty-five"	KC-135 aircraft.
lbs	"lbs"	Pounds.
lindsey	"lindsey"	
list	"list"	
location	"location"	
make	"make"	
map	"map"	
merseberg	"Merseberg"	Name of a city and airbase in E. Germany.
minutes	"minutes"	
mission	"mission"	
mobility	"mobility"	
munition	"munition"	
munitions	"munitions"	
name	"name"	
nationality	"nationality"	
near	"near"	
nurenberg	"Nuernberg"	Name of a city and airbase in W. Germany.
oca	"oh-cee-ah"	
oca345	"o-cee-ah three forty five"	Offensive Counter Air Mission Number 345.
oca445	"o-cee-ah four forty five "	Offensive Counter Air Mission Number 445.
oca555	"o-cee-ah five fifty five"	Offensive Counter Air Mission Number 555.
of	"of"	
on	"on"	
package	"package"	
packages	"packages"	
path	"path"	
pause	"pause"	Command that deactivates speech recognition system (see also resume command).
pkg0023	"pee-kay-gee oh-oh-two-three"	Package Number 0023.
pkg0066	"pee-kay-gee oh-oh-two-three"	Package Number 0066.
plan	"plan"	
plans	"plans"	
point	"point"	
present	"present"	
refueling	"refueling"	

region	**"region"**	
remove	**"remove"**	
reset	**"reset"**	Command that starts new word sequence in speech recognition system, deleting previously started word sequence.
resume	**"resume"**	Command that reactivates speech recognition system (after a pause command)
rfl345	**"ar-ef-el three forty five"**	Refueling Mission Number 345.
rhein-main	**"Rhein Main"**	Name of a city and airbase in W. Germany.
sam	**"sam"**	Surface-to-air-missile.
seven	**"seven"**	7:00
seven-fifty-five	**"seven fifty five"**	The time 7:55.
seven-fourty-five	**"seven forty five"**	The time 7:45.
seven-o-two	**"seven oh two"**	The time 7:02.
seven-twenty	**"seven twenty"**	The time 7:20.
six	**"six"**	The time 6:00.
six-fifty	**"six fifty"**	The time 6:50.
stargard	**"Stargard"**	Name of a city and airbase in Poland.
start	**"start"**	
starting	**"starting"**	
stn	**"es-tee-en"**	Short for station.
stn002	**"es-tee-en oh-oh-two"**	Station Number 002.
stn244	**"es-tee-en two-two-" four"**	Station Number 224.
storage	**"storage"**	
strike	**"strike"**	
svc	**"SVC"**	
svc001	**"SVC001"**	
svc345	**"SVC345"**	
svc445	**"SVC445"**	
table	**"table"**	
tank	**"tank"**	
tanks	**"tanks"**	
target	**"target"**	
targets	**"targets"**	
this	**"this"**	
the	**"the"**	
these	**"these"**	
threat	**"threat"**	
threats	**"threats"**	
time	**"time"**	
to	**"to"**	
twenty-one-thousand-nine sixty	**"twenty one thousand nine sixty"**	The number 21960.
twenty	**"twenty"**	The number 20.
twenty-thousand-nine-fourty-two	**twenty thousand nine fourty two**	The number 20942.
type	**"type"**	
types	**"types"**	
units	**"units"**	
nit	**"uni"**	
value	**"value"**	
what	**"what"**	
where	**"where"**	
will	**"will"**	
window	**"window"**	
within	**"within"**	
soom-in	**"soom in"**	Command that causes the system to display more detailed view of a map area.

Script

(1) Display the Fulda Gap region.

(2) Where is the Erfurt airbase?

(3) Where is the Nuernberg airbase?

(4) Where is the Stargard airbase?

(5) Blink the heliports.

(6) What are the threats around the Dresden airbase?

(7) Zoom in on this point <point>.

(8) List the packages.

(9) Make PKG0023 the current package.
or
Make this <point-on-package-table> the current package.

(10) Display the forms window.

(11) Enter OCA445 here <oca-#2>.

(12) Expose this window <pt-main-map>.

(13) Enter <pt-numberg> here <oca-origin2>.

(14) Enter 6:00 here <oca-tod2>.

(15) What are the aimpoints within the Dresden airbase?

(16) What are the aimpoints within the Dresden airbase?

(17) What is the mobility of these <point-more-4-sams>?

(18) Enter this <pnt-to-sam> here <pnt-form-aimpt-slot>.

(19) What is the mobility of these <point-less-5-sams>?

(20) Is this <point-on-Dresden-aimpoint-window> a sam?

(21) What are the aimpoints within the Erfurt airbase?

(22) Highlight this <point> on the table.

(23) Highlight this <point> on the {map/color-graphics display}.

(24) Enter this <pnt-to-Dresden-runway> here <pnt-form-aimpt-slot>.

(25) Enter 7:02 here <strike-TOT>.

(26) What ac-pools are at the Nuernberg airbase?

(27) Enter this <point-ac-pool> here <oca-ac-pool2>.
or
Enter the ac-pool 45TFW-EF-111E here <oca-ac-pool2>.

(28) Enter SVC445 here <svc-#2>.

(29) Enter 7:45 here <svc-start>.

(30) Enter 20 minutes here <svc-durat2>.

(31) Enter the value 20942 lbs here <svc-disbur2>.

(32) Enter STN244 here <svc-stn-#1>.

(33) Enter STN244 here <svc-snt-#2>.

(34) Enter this location <click on map> here <stn-orbit>.

(35) Plan a flight path for OCA345.

(36) Plan a flight path for OCA445.
(In the near future, the user should be able to enter the way points of the flight path via mouse-points on the map.)

(37) Present the OCA345 and OCA445 mission plans.

(38) Remove the flight paths.

(39) Make PKG0066 the current package.

(40) Zoom in on this point <point-on-map>.

E.2 Completed Air Force Application-Oriented Evaluation Questionnaire

This sections contains the completed Air Force Application-Oriented Evaluation Questionnaire.

Air Force User Evaluation

This section contains the questionnaire completed by the Air Force User Evaluator. Evaluator responses are presented in italics.

The following instructions were provided at the top of the User Evaluation Questionnaire.

> This questionnaire is intended to provide general information about usability and applicability of CUBRICON within military mission planning applications, and to solicit suggestions for improvement. Answer the following questions and be prepared to discuss your answers.

The questionnaire along with the answers provided by the Air Force Representative, are contained in their entirety in this section.

1. Do you think an interface like CUBRICON would provide an effective tool for working with computer-resident data bases and related military mission planning tasks? Why?

   Yes. If the speech capability would support continuous speech, it could be faster and more efficient than typing. In general, any of the capabilities that would allow the planner to work faster could be helpful.

2. What aspects of CUBRICON did you find to be especially efficient and helpful in accomplishing desired actions? Explain.

   I thought that the speech understanding and parsing had the most interesting potential. Another area that has potential is the automatic removal and handling of windows for the user. The idea that a pop-up window would not cover up a portion of the map that was recently referenced was a good idea. Some of the other concepts, I take for granted because of my work with the Symbolics and TEMPLAR, but these represent a significant improvement over existing command and control capabilities. One area I really liked was the language learning. For terms that I thought were to long like fourty-fifth-tfw-ef-111, I abbreviated it as fourty-fifth and it then expanded it to the required term for the command line interface. I also think that the idea of expanding out the targets to show the aimpoints is an excellent idea and helps to declutter the map from all the targets.

3. What aspects of CUBRICON did you find to be especially difficult to use or inefficient? Why?

   In general I didn't like the interface to the forms and tables. Often the easiest way to use a form would be to mouse on a slot and type or speak to enter a value. The user should have to do a minimal number of entry modes, during execution of a specific process.

   If I am talking and moving the mouse to point to things I don't like the idea that I have to switch modes to keyboard to enter a function-X to mouse on something on the map or a table. The tables should pop up on the monochrome display so they can be moused with just a click. It would be nice to be able to display only the part of the form you were working on ie. a single mission in a window and be able to iconize it when you are done. Then when you want to look at the package show them in an integrated way. The system queries after they were parsed seemed much to slow and might make CUBRICON difficult to demonstrate unless it speeds up significantly on a larger Symbolics. Since this is a 6.1 effort designed to show something else it is not really an issue but does detract from CUBRICONs use.

4. How would you compare CUBRICON's approach to working with computers to other more conventional computer interfaces? Describe specific features of both types of systems used in your comparison, and state whether CUBRICON was better, worse, or about the same in terms of its capabilities.

   Once again my answer to this one is slanted because I use a Symbolics at work and an Amiga at home, so I am very familiar with window/icon/mouse/pointer interfaces in addition to animation. In general CUBRICON windows are better than the ASCII type displays that still dominate existing command and control systems an IBM PC type interfaces, but are not as good as some of the windowing systems I have seen that work interactively (i.e., dynamically tracking resources available in one window while changes are being made in another). This was one of the key complaints about TEMPLAR that you couldn't do stuff like that because of the single form format of the monochrome screen.

5. Did you encounter any problems in using CUBRICON? What were they? How could they be avoided (e.g., better training, redesigning)?

   Yes, Several bugs in software. If I deviated much from the script things tended not to work (like asking for ac-pools at Rhein Main). Once in a while I would have trouble *getting the DEC talk to understand certain words. Parsing speed appeared very good, but execution of the commands was too slow.*

6. Consider the following specific and comment on how well or poorly CUBRICON performed with respect to them:

   - Organizing outputs for understanding.

     - There was a tendency to repeat things, i.e., Blink a Base, point at it with a label, then re-displaying a table with it added to it, talking and printing text in the text screen seemed like overkill.

   - Keeping track of routine information.

     - The Iconize screen would be helpful if they were labeled with something understandable and could be de-iconize. Also see my comment on tracking the resources in 4.

   - Keeping you informed about the overall situation and progress towards task accomplishment.

     It keeps you informed about the overall progress (possibly over-informed). It doesn't really tell you what you have to do to complete the overall task (the order in which things have to be done).

   - Allowing you to make inputs easily and efficiently.

     Speech was easy but not always more efficient than typing in a value. It would be real nice to be able to click on a slot form, say the value and have it entered in the slot. Saying "Enter six here ¡point and click¿ enter" was to slow. I also liked the ability to extract thing with the mouse, but here it might be easier to point and click to extract it then slide over to the window and click another mouse button to put it somewhere. The system currently requires "Enter $< typefunction - X >< Click - nurnberg >< typefunction - X >$ *here* $< clickOCA - origin2 >$ *enter."*

   - Allowing you to focus on the application.

Overall it tends to keep you focused on the application unless the speech is failing or response time is too slow.

- Meeting your personal preferences for problem solving approach and information display approaches.

  Partially; better windows and faster speech would be nice.

7. How can we make CUBRICON better?

Windows and icons should be exposeable with just a mouse click. Continuous speech would be nice. More understandable (human-like) output speech. The table information should high-light when you have the mouse on it and bold when selected. It would be nice if the color and mono screens were connected to each other, i.e., if you move up/down/left/right on one screen you end up down/up/right/ left on the other screen instead of function-X. Some map features like display the object name in a small sub-window when the mouse is over it and instant tracking of the lat-long in another would be nice. Being able to do inquiry about an object with a mouse click or having the ability to assign functions to user assignable hot-keys also would be nice.

E.3 Completed Human Factors Evaluation Questionnaire and Checklist

This sections contains the completed Human Engineering Evaluation Questionnaire and Checklist. It includes both the top-level summary portions as well as the more detailed and supporting checklist item responses. The numbered items within the evaluation categories are cross-referenced to the top-level project goals from the SOW. The resultc of the evaluation organized according to SOW goals are available upon request from the authors.

Interface Engineering Evaluation Checklist

Rate CUBRICON's performance with respect to the evaluation categories. The numbered items (-1, -2, etc.) within each category will help in making your assessments. These numbered items are not intended to serve as the sole basis upon which to make your assessments. All observations you believe are relevant should be considered. State the rationale on which you base your ratings.

Refer to Smith and Mosier (1986) to guide your evaluation. Many of the numbered items include references to Smith and Mosier. These references are listed within parentheses at the end of the items. Bear in mind that CUBRICON is built using new technology. Its approach to user-interface design is new and inovative. The guidelines in Smith and Mosier were developed for conventional interfaces. If CUBRICON violates any of the Smith and Mosier guidelines, ask yourself whether the violations could represent an improvement over conventional user-interface approaches, or whether they are the result of poor design.

Finally, be critical! Don't be afraid to tell us what you think (good and bad). Stress the system. Find out where its weak points are and tell us how we can make it better. If you need more time, take it. The results of this evaluation will guide future design efforts.

Note: The numbered items within the evaluation categories are also cross-referenced to the top-level CUBRICON goals that were stated in the SOW. These are noted using the *number* format. These references are not meant to be used during the hands-on portion of the CUBRICON evaluation but will be used during subsequent analysis and reporting.

1. The Efficiency and Effectiveness of Making INPUTS to CUBRICON.

1.1 Rate the general ease, naturalness, and effectiveness of making inputs to CUBICON:

Rating	
Excellent	.....
Very Good	.....
Adequate	..X..
Poor	.....
Extremely Poor	.....

Comments

Frequent misinterpretations of speech input was inefficient. However, a better speech recognition system and increased training and practice could alleviate this problem. Allocation of mouse to screen (color graphics or monochrome) via the keyboard was cumbersome. This could be improved by using the right and left buttons on the mouse to select the desired screen. The use of specific command verbs to initiate

specific actions was difficult to remember, particularly when the verbs have similar meanings (e.g., display and present). A more generalized use of verb commands would lessen memory load.
The option of input media (speech, text, pointing) or combinations of media made the system enjoyable to use. It accomodates differences in task demands and user preferences.

-1 Inputs to CUBRICON could be made using the most convenient and desired media/modalities and in a manner that seemed natural and efficient. *1*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Some limitations include: no pointing at form except for input; allocation of mouse to screen was cumbersome (Assignment of mouse, i.e., cursor, to screens with mouse button rather than keyboard would be more efficient); only mouse click could be used to specify location (not speech) with zoom-in command (speech or text didn't work); it is inconvienient when speech is misinterpreted requiring reset (text has to be cleared and the statement must be repeated from the beginning).

-2 Inputs to CUBRICON were correctly understood the first time without clarification or reformating. *1*. Circle one: Always, Usually, Sometimes, *Rarely*, Never. Comments:

Frequent misinterpretation of speech required clarification.

-3 Verbal reference to objects within the CUBRICON data base could be made using desired and natural terminology (3.1.6.5, 3.1.7.1). *1*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Structure of command sentences was somewhat rigid (e.g., "zoom-in on this -mouseclick- point" was acceptable, "zoom-in on this -mouseclick- location" and "zoom-in on this -mouseclick- " were not). The system would tolerate ommissions of "the", however. Also, command verbs tied to specific actions were difficult to remember.

-4 CUBRICON provided for efficient specification and input of spatial/geographic information (e.g., flightpaths, putting objects at desired locations) (1.6.2 - 1.6.9). *1*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Did not put objects at locations. No feedback for first location specified for flightpath was provided until second location was specified.

-5 The use of data entry forms was straightforward and not prone to errors (e.g., areas for data entry were clearly delineated and movement between them was natural and efficient) (1.0.6, 1.4.all, 2.2.all, 3.1.2.1 - 3.1.2.4). *1*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

It was not clear which areas were required to be completed or whether there was a hierarchical order to fill them. Format, and movement between areas was straightforward.

-6 Pointing at desired objects could be accomplished equally well on the various types of windows displayed (e.g., tables, maps) and on the monochrome display as well as the color display. *2*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Pointing worked well for maps and for text. Pointing on the form was only enabled for input. When numerous icons were displayed in close proximity, more than one icon was picked-up by point and the system crashed.

Reference: Also consider items: 1.3-4.

1.2 Rate the ability of CUBRICON to accept, integrate, and understand inputs that were made using multiple media/modalities:

Rating		Comments
Excellent	.....	*Being able to point at several objects as part of an input, the combined use of speech and pointing for an input, and the use of multiple windows for an input are all excellent features that made the system easy to use.*
Very Good	..X..	
Adequate	.....	
Poor	.....	
Extremely Poor	.....	

-1 The ability to point and speak at the same time was helpful in making inputs. *1*. Circle one: *Always*, Usually, Sometimes, Rarely, Never. Comments:

Easy to use.

-2 Mouse points were correctly related to the intended objects described via natural language. *6*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

-3 It was possible to point at multiple objects as part of an input, and these were correctly integrated and understood within the dialogue by CUBRICON. *6*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

When the system was querried on the mobility of several icons, one of which had no mobility, the voice response called it a miss while the text said it had no mobility.

-4 It was possible to make inputs efficiently using multiple windows (e.g., pointing at objects in different windows when defining a target list). *1*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Unable to point at the form except for input.

-5 The ability to define inputs on one window by pointing at objects on another window (e.g., in completing forms) was efficient and easily accomplished. *1*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Process was easy, except for allocating mouse to screen.

Reference: Also consider items: 1.1-1.

1.3 Rate the abilitv of CUBRICON to understand inputs based on the dialogue context:

Rating		Comments
Excellent	.....	*Formatting of speech input was somewhat rigid. The system was not very tolerant of deviation from this structure. Allowing verb commands with similar meanings (e.g.,*
Very Good	.....	
Adequate	..X..	
Poor	.....	

Extremely Poor ..... *display and present) to be used interchangably would be helpful.*

-1 The formulation of inputs to CUBRICON flowed naturally from the context of the displays and dialogue and did not require translation in in order to achieve acceptable structure and formats (e.g., the terminology acceptable for data control and input was consistent with the style, terminology, and format used for output) (2.0.7, 3.1.8.5). *1*. Circle one: Always, Usually,

Sometimes, Rarely, Never. Comments:

Structure of commands requires memorization. Formulation of data input to the form was automatically structured by the system (e.g., "arrival time is six" formatted as 6:00) making it convienient to use.

-2 Inputs to CUBRICON could be made within the ongoing dialogue without invoking special procedures or calling special displays (1.0.2). *1*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Most input done on one screen containing form, command scroll area, and system.

-3 Inputs that are illogical based on the task and data context, were noted by CUBRICON and communicated (1.6.19, 1.7.1). *5*. Circle one: Always, Usually, Sometimes, *Rarely*, Never. Comments:

Very little error trapping. The system would proceed with next command without recognition that required information was omitted. Error feedback messages were not informative.

-4 CUBRICON provided prompts or reminders based on the task being performed (e.g., guides for accomplishing complicated procedures were available when needed)(3.1.8.6, 3.1.8.7, 3.2.4, 3.2.5). *7*. Circle one: Always, Usually, Sometimes, *Rarely*, Never. Comments:

No prompts were available for filling out form (e.g., what areas were required to plan a flight path, guidance for possible hierarchy of form entries). Also, no prompts or error checking.

-5 CUBRICON was able to correctly relate pronouns and indefinate references to their proper referent (based on the preceding dialogue). *8*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Voice output used pronouns correctly. Pronouns not accepted as input.

-6 CUBRICON was able to correctly interpret inputs based on the context of the dialogue (e.g., requests for information produced outputs relevant to the dialogue; requests that made no sense based on the context were questioned). *8*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

-7 Ambiguous mouse points were correctly resolved by CUBRICON based on the context of the dialogue. *5*. Circle one: Always, *Usually*, Sometimes, Rarely, Never.

- Inaccurate points were correctly resolved.
- Incorrect points that made no sense were corrected or questioned.

Comments:

Had problem when selected icon was in close proximity to other icons.

-8 The CUBRICON vocabulary and grammar was sufficient for expressing desired concepts and data. *1* Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

The vocabulary and grammar seemed appropriate the the application.

Reference: Also consider items: 1.2-2, 1.2-3.

2. The Efficiency and Effectiveness of CUBRICON OUTPUTS.

2.1 Rate the general understandability, effectiveness, and smoothness of CUBRICON outputs:

Rating		Comments
Excellent	.....	*Voice output was sometimes difficult to interpret. Additional labelling of maps and tables would be helpful. Outputs were generally clean and easy to understand.*
Very Good	..X..	
Adequate	.....	
Poor	.....	
Extremely Poor	.....	

-1 CUBRICON outputs were clear and understandable without requests for clarification. Information was presented in a form that could be clearly and unambiguously understood and could be related to the task being performed (2.0.3, 2.4.9). *4*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Voice was difficult to interpret on many occasions, however, this problem could disappear with increased training and usage.

-2 Information needed for interpreting displays was readily available (e.g., a key defining the meaning of symbols used on a map, appropriate supplementary information presented via an appropriate media) (2.0.1). *4*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Maps and tables were rarely uniquely identified. A key wasn't available but I'm not sure it is necessary.

-3 CUBRICON displays employed labels that were clear, consistent, and helpful. This included labels within displays as well as labels identifying the display itself (2.2.3 - 2.2.10, 2.3.7 - 2.3.9, 2.3.11, 2.4.11, 2.4.1.1 - 2.4.1.9, 2.7.1.2 - 2.7.1.4, 2.7.5.6). *4*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Maps and tables are not uniquely identified. Labels within the form would probably be more meaningful to the mission planner.

-4 CUBRICON displays employ coding schemes that were clear, consistent, and adequately captured the important distinctions among display elements (2.6.3 - 2.6.38). *4*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Need to code or label maps and tables. Map icons were easy to distinguish from each other. Boxing of highlighted items on tables were difficult to distinguish. Bold face type would be easier to see. Also, red arrow pointer was difficult to see in enemy territory. Use of a distinct color would make it easier to distinguish.

-5 When items were selected (by the user or the system) this was clearly conveyed to the user (1.6.7, 3.4.6). *5*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

The first selection when drawing a flightpath isn't indicated until second paint is selected. Selected items on tables were boxed on color graphics display. This was hard to decifer. Bold face would stand out better.

-6 CUBRICON clearly communicated its activities especially when processes were not immediate (2.7.1.7, 3.0.14, 3.0.15). *7*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

System status line would generally say "run" or "user input". Specific activities weren't that were ongoing were not identified.

-7 Spatial relationships among graphic elements (e.g., elements on a map) were clearly presented (e.g., it was possible to accurately judge distances or query for exact distances (1.6.1.5, 1.2.2.4, 2.4.1.11, 2.4.1.12, 2.4.8.3, 2.4.8.3). *5*. Circle one: Always, Usually, Sometimes, Rarely, Never. Comments:

Unable to querry for exact distances. Option to impose a grid on map would be helpful. Part of scale was obscured at the origin.

-8 The general organization and layout of windows was efficient for the tasks at hand. *5*. Circle one: Always, *Usually*, Sometimes, Rarely, Never.

- The relative location of related windows allowed necessary comparisons and efficient accomplishment of tasks involving multiple windows.
- Window sizes were sufficient for presentation of the information that each needed to present.

Comments:

Window size seemed appropriate. Location of windows was sufficient for task completion except when automatic deletion removed a map still in use (Note: this occurred following a permanent zoom-in when the context map was removed).

2.2 Rate the appropriateness and effectiveness of CUBRICON media/modality selection and integration:

Rating Comments

Excellent	.....
Very Good	.....
Adequate	..X..
Poor	.....
Extremely Poor	.....

I think the integration of media was effective. Its effectiveness would probably be more apparent with a faster system response time and heavier workload conditions. Voice messages about map display changes were convienient since it allows the user to remain fixated on the display while changes are described.

-1 Speech, graphic, and textual outputs were used appropriately and in the right proportion to clearly, concisely, and efficiently accomplish the necessary communications (4.0.26 - 4.0.29). *6*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

On some occasions presentation of both voice and text message seemed overly redundant. Processing of voice and text message seemed to increase information processing load. Graphic output was clear and easy to use. An exception to this was the flashing of data in the text window after a flight path was presented.

-2 Outputs were presented using media/modalities that were appropriate for the content and context of the communication (2.4.1 - 2.4.3, 2.4.6.1, 2.4.6.2, 2.4.8.1). *4*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

The ability to obtain a hard copy would enhance the system.

-3 Information that was presented for comparative purposes was displayed in a manner suited for such comparisons (e.g., side by side in a table, highlighted on a map using clear distinguishable codes, etc.) (2.3.1, 2.3.5, 2.4.2, 2.5.13). *4*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Comparison of maps could be made easier by allowing the user to pull a map up from storage. Also, two different forms cannot be displayed side-by-side for comparison, but this may be necessary.

-4 Tables presented information in a manner that facilitated efficient use (e.g., tables were organized by the parameters with which the information was to be accessed or it was a simple matter to reorganize the table to meet this requirement) (2.3.3, 2.3.4, 2.3.12 - 2.3.17, and to a lesser degree all of 2.3). *1*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Ordering of column information may need to be rearranged. Feedback from military typ^e would be helpful (e.g., name of item column imbedded between other columns and was the 5th of 6 columns).

-5 Maps were presented in a way that facilitated their effective and understanding (2.4.8.2 - 2.4.8.9, 2.4.8.15, 2.4.8.17, and to a lesser degree all of 2.4.8). *4*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Occasionally areas of the maps were overly cluttered with icons that couldn't be differentiated. Due to lack of labels, there was no way to quickly see which tables were asociated with which maps. Also the map scale was ambiguous.

-6 CUBRICON made unambiguously clear, which graphically displayed objects were referred to via an associated media/modality. (e.g., verbal outputs were related to associated displayed items in a clear and unambiguous fashion). *6*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Identification was promarily made by change in color code and flashing on graphics display.

-7 It was possible to relate items in tables or on forms to their graphic representations (e.g., on a map). *4*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Maps and tables need to be uniquely identified and the association between them demonstrated (e.g., a line connecting them or coded in some way.

-8 CUBRICON speech output was helpful in providing orientation to other system outputs (e.g., created or modified maps, tables, etc.). *6*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Voice output may be more useful when system response time is faster or workload increases.

-9 Auditory and voice coding was employed effectively to communicate important distinctions among auditory displays (2.6.39 - 2.6.42). *6*. Circle one: Always, Usually, Sometimes, *Rarely*, Never. Comments:

Chimes used to indicate declutter of graphic display as opposed to voice. This was the only auditory coding used.

-10 When relations among information components are important, integrated displays (individual or multi-media) that show those relations were provided (2.5.7, 2.7.2.1). *6*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

The relationship between map and entity tables wasn't demonstrated.

-11 Information that was needed temporarily was made available on a temporary basis (rather than cluttering displays with such information) (2.7.5.1). *7*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

The use of window overlays served this purpose. The ability to display and remove windows as desired, would enhance the system.

Reference: Also consider items: 2.1-1, 2.1-2.

2.3 Rate CUBRICON's effectiveness at selecting and controlling output quantity, level of detail, and resolution:

Rating		Comments
Excellent	.....	*Need to indicate scrolling option on tables. On the whole, maps were easy to read and use. Text in tables on graphics display was difficult to read.*
Very Good	.....	
Adequate	..X..	
Poor	.....	
Extremely Poor	.....	

-1 Map displays contained an appropriate amount of area at an appropriate scale (without resizing) for task accomplishment (e.g., zoomed-in or out to correct amount of detail and area coverage) (1.6.5). *3*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

User control of zoomed area size would enhance system, along with addition of zoom-out feature.

-2 Map and other graphic displays, and symbols used within them, were large enough to provide the resolution needed to resolve objects and determine necessary relationships among objects. *3*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Cluster of icons couldn't be deciphered, otherwise, maps and symbology were easy to read. Removable grid would be useful for distance relationships. Tables on graphic display were difficult to read due to text size.

-3 CUBRICON responses to requests for information provided the information in a level of detail consistent with the request and the context of the request (e.g., only necessary information was displayed, yet sufficient detail was provided for the task) (2.0.2, 2.4.5). *3*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

-4 When a request for information resulted in a large volume of information, CUBRICON provided a means for dealing with the information in an organized and efficient manner, and/or helped the user rescope the request (2.2.14, 2.4.6.3, 2.5.4, 2.7.2.2 - 2.7.2.6, 2.7.2.10). *3*. (see below for examples). Circle one: Always, Usually, *Sometimes*, Rarely, Never.

- Displayed scrolling window with scroll bar (and perhaps a slider). **No, not completed**
- Presented an indication of the percentage of the information included in and above the displayed window. **No, not provided**
- Indicated the number of items that satisfied a query, and perhaps provided an opportunity to focus the request. **No**
- Presented summary or top-level map of the information. **No**
- And so forth . . .

Comments:

It wasn't clear that information on tables could be scrolled or how scrolling would be accomplished.

-5 An appropriate means for highlighting critical information was used (considering the nature of the critical information, the task context, and other coding schemes in use) (2.4.0.6, 2.4.0.8, 2.4.0.19, 2.4.6.4, 2.6.1). *3*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Required information on the form (for planning a flight path) wasn't indicated. Highlighting on the map was effective. Flashing of items in some instances was excessive and inappropriate.

-6 It seemed that the information being displayed was well controlled (e.g., it was never overwhelming). *3*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Automatic declutter prevented this.

Reference: Also consider items: 2.2-10.

2.4 Rate how well CUBRICON maintained context clarity:

Rating		Comments
Excellent	.....	*Marking of original zoomed-in area*
Very Good	.....	*boundary on original map was helpful.*
Adequate	..X..	*The relationship between maps and*
Poor	.....	*tables needs to be made explicit.*
Extremely Poor	.....	*Labels on maps and tables to uniquely*
identify contents are needed.		

-1 When displayed information is relevant only in a certain context, this is clearly communicated (e.g., dynamic information includes time stamp, available weapons indicate compatible platforms). *5*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Not all pertinent time stamps were given on the map (e.g., target strike time) (Note: this occurred during flight path presentation).

-2 Adequate contextual information was available for the proper interpretation and use of displayed information (2.0.11, 2.4.18, 3.4.1, 3.4.7). *5*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

-3 When information was best interpreted relative to some significant level or critical value, this comparison was clear from the display (2.4.7, 2.4.8.18). *5*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Some critical times (departure time, strike time) missing from flight path map display.

-4 CUBRICON communicated information in a manner in which the structure of and relationships among the data being entered or displayed was clear (e.g., hierarchical relationships) (1.0.31, 1.6.18, 1.8.12, 2.2.1, 2.3.10, 3.1.6.3) *5*.

Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Hierarchical relationships among data on the form wasn't clear.

-5 When displays are changed (e.g., removing windows or information, zoom-in or out, panning, scrolling), adequate cues are provided for maintaining orientation to the larger context (2.0.11, 2.4.16, 2.4.17, 2.4.18, 2.4.8.2, 2.4.8.11, 2.4.8.16, 2.6.2, 2.7.2.14 - 2.7.2.17, 2.7.3.2, 2.7.3.4, 3.3.5). *5*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Zoom-in area noted with box on original map. No cues provided for scrolling context. For example, when tables have been scrolled, table and column labels scroll off the display and context is lost.

-6 Output formats were consistent with expectations based on the preceding dialogue and the context of pre-existing displays (3.0.16, 3.1.1). *2*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

-7 Standard displays used standard formats that were readily identifiable and useable (e.g., standard information was contained and consistently organized in display headings) (2.0.6, 2.0.13, 2.0.14, 2.0.15, 2.1.3, 2.2.13, 2.4.4, 2.4.10, 2.4.12, 2.5.1). *2*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

(Windows) not uniquely identified with labels.

-8 CUBRICON provided prompts to help in making standard or required inputs or when omissions were inadvertantly made (1.0.24). *2*. Circle one: Always, Usually, Sometimes, Rarely, *Never*. Comments:

Error feedback was not informative/diagnostic.

-9 Windows were managed in a way that minimized and disruption to display context. *5*. Circle one: Always, *Usually*, Sometimes, Rarely, Never.

- The most important windows were kept on the screen. Windows that were removed were less important and not critical to the ongoing task.
- The largest windows were used for the most important information or when large amounts of detail had to be presented.

Comments:

Reference: Also consider items: 2.1-2, 2.1-3, 2.1-4, 2.1 5, 2.4-8, 2.3-5.

2.5 Rate the appropriateness and effectiveness of voice output as used within CUBRICON integrated outputs:

Rating	
Excellent	.....
Very Good	.....
Adequate	..X..
Poor	.....
Extremely Poor	.....

Comments

Evaluating the use of voice output is problematic because system response time was slow and this magnified the feeling that voice output was extrinsic to the task. A frequent user of the system would not require as much voice feedback as is currently provided and a means of adjusting the level/amount of voice feedback should be addressed in future enhancements of the system.

-1 Rate the appropriateness of CUBRICON's decisions about when to use speech output. Relate your answer within the following categories *7* : **Adequate**

- CUBRICON speech output did not interrupt user inputs (especially speech inputs) and allowed the user to interrupt if desired or necessary.

  User was unable to interrupt the speech generator. Most of the time speech occurred when processing was taking place (and user could not input data) so speech didn't interrupt user activity. Future enhancements could include a method for stopping voice output when user input occurs.

- Speech output was used when there was a requirement for rapid two-way exchanges of information.

  This was hard to evaluate since system response was slow.

- Speech output was used when the information to be presented dealt with a future time requiring some preparation, and especially when it was intended for immediate use.

  Speech was used to inform the user about display events that were about to happen, and to present information about displayed items.

- Speech was used when it was important to elicit attention from other tasks or activities.

  Speech was used to draw attention to the appropriate display.

- Speech was used when information needed to be presented independant of head or eye movement.

  Speech presentation allows user to fixate on map display activity while receiving information about the display via voice output. At other times the voice output seemed extraneous and overly redundant.

-2 Rate how well CUBRICON constructed speech outputs (i.e., were speech outputs constructed in a manner that maximized overall communication efficiency and understandability?) Relate your answer within the following categories *7* : **Very good**

- Standard and consistent terminology was used for expressing common concepts.

  The use of terminology was standardized and consistent.

- Terminology that was meaningful to the user population was used.

  The terminology seemed appropriate to the application.

- Consistent phraseology was employed throughout all parts of the interface.

  Phraseology was consistent.

- Speech output vocabulary was coordinated and consistent with speech recognition capabilities.

  Vocabulary/terminology of speech input and output was similar.

-3 Voice outputs were constructed in a manner that facilitated accurate perception and understanding. Relate your answer within the following categories *4*: **Sometimes**

- Important words were placed near the end of messages so that surrounding sentence structure would provide context and facilitate intelligibility.

  This was the case in most instances. However, when voice output was given regarding mission duration the message, "the duration is this", was given, thus failing to provide the usre with critical information.

- Multi-syllable words were selected when possible to provide linguistic redundancy and reduce phonemic uncertainty within any given word.

  Multiple syllable words were used when appropriate. It may be the case *that frequent usres would want to abbreviate multi-syllable words or multi-word inputs.*

- Voice outputs were kept as short as possible.

  Voice messages were generally shorter than text messages. It may be possible to shorten them some more.

-4 CUBRICON speech outputs were coordinated with ongoing tasks and related outputs using other modalities *6*. circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

This was a problem because of slow system response. See comments in 2.5.

-5 It was possible to have messages repeated when needed *1*.

Users didn't have the option to have speech messages repeated (in speech) but all speech messages were represented in text messages (this provided for this function).

3. Sequence and System Control Issues

3.1 Rate the efficiency and effectiveness with which the CUBRICON user-interface was controlled:

Rating		Comments
Excellent	.....	*User control of windowing was limited, although some features that would improve user control (e.g., zoom-out) were not enabled yet.*
Very Good	.....	
Adequate	..X..	
Poor	.....	
Extremely Poor	.....	

-1 It was possible to clearly and easily specify desires for control and transformation of maps (e.g., specify area for zoom-in and zoom-out, and panning) (1.6.5, 1.6.6, 1.6.8, 2.4.8.10, 3.0.4, 3.2.1) *1*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Zoom-out, recall of stored maps/tables weren't enabled. User option to selectively declutter or redisplay maps would enhance the system.

-2 It was possible to customize displays to meet personal preferences (e.g., reorganize table columns, redefine area, displayed on a map, redefine window layout). *2*. Circle one: Always, Usually, Sometimes, Rarely, *Never.* Comments:

Features not available.

-3 The automatic management of windows (e.g., positioning, sizing, and removal) was accomplished in a way that facilitated their use while allowing user intervention to achieve alternative window organizations when desired (3.0.1, 3.0.2, 3.0.4, 3.0.5, 3.2.1) *2*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

User intervention was limited.

-4 Display changes did not disrupt the ongoing dialogue (e.g., did not remove needed windows, display changes were consistent with expectations) (3.0.7) *4*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Unable to input data while display changes in progress. Occasional E-W Germany map would be removed when I would have liked it available for reference.

-5 Feedback about CUBRICON's acceptance and understanding of inputs was sufficiently quick and clear (3.0.9, 3.0.14, 3.0.15). *5*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

System response was slow. Acceptance of first point on flight path wasn't clear.

-6 There was a simple means for indicating to CUBRICON when verbal inputs were meant for CUBRICON and when they were not (e.g., ignore and continue) (1.0.37). *5*. Circle one: *Always*, Usually, Sometimes, Rarely, Never. Comments:

-7 When not all windows were active, CUBRICON clearly indicated which were active and provided an efficient means for selecting desired windows (2.7.5.7, 2.7.5.8). *6*. Circle one: Always, Usually, Sometimes, Rarely, Never. Comments:

Recall of inactive windows not enabled.

-8 It was possible to cancel partially completed inputs (including voice inputs) and ongoing CUBRICON processes by invoking an explicit CANCEL command (1.0.11, 1.0.35, 3.3.1, 3.3.3). *7*. Circle one: *Always*, Usually, Sometimes, Rarely, Never. Comments:

-9 Control of the CUBRICON interface was handled effectively. Updating of displays was efficient and did not require excessive effort, while at the same time the ultimate control of the interface was available to the user (2.0.8, 2.0.9, 2.7.1, 2.7.1.1, 2.7.1.5, 2.7.1.8, 2.7.1.9, 2.7.2.13, 2.7.3.5 - 2.7.3.8, 2.7.5.1, 2.7.5.2, 3.0.4, 3.2.1). *7*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Very little control of windowing operations.

-10 System operations logically reflected user inputs inputs and desires (3.0.16). *2*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

-11 Issuance of commands to CUBRICON was efficient and easy (3.1.5.2, 3.1.5.21, 3.1.5.22). *1*. Circle one: Always, Usually, *Sometimes*, Rarely, Never. Comments:

Speech system difficult to use due to frequency of recognition errors.

-12 It was possible to maintain control over dynamic displays (e,g., PAUSE and CONTINUE commands) (3.3.8 - 3.3.11). *7*. Circle one: Always, Usually, Sometimes, *Rarely*, Never. Comments:

No real-time control over flight path presentation (the only truly dynamic display).

-13 The response time for voice, text, and graphics inputs was sufficiently fast to ensure efficient, continuous dialogues 3.0.18, 3.1.2). *4*. Circle one: Always, Usually, Sometimes, *Rarely*, Never. Comments:

System response time was too slow.

-14 Measure and note typical response times for the following inputs:

- Voice input requesting highlighting of currently displayed information. Note the following: Feedback that request was understood - *No feedback*; Response complete - *1-2 seconds.*
- Voice input requesting a new map to be created (e.g., zoom-in). Note the following: Feedback that request was understood - *3 seconds*; Response complete - *45 seconds (map) plus 20 seconds (table).*
- Selection of a menu item (e.g., select a package from list of available packages).Note the following: Feedback that request was understood - *60 seconds*; Response complete - *62 seconds.*
- Issuance of typed command requesting a standard display (e.g., request display of the forms menu). Note the following: Feedback that request was understood - *70 seconds*; Response complete - *75 seconds.*
- Typical graphic interaction (e.g., point at graphically displayed item for selection). Note the following: Feedback that request was understood - *4 seconds*; Response complete - *4 seconds.*

Reference: Also consider items: 1.1-7, 1.3-4, 2.1-6, 2.4-5, 2.4-8.

3.2 Rate the efficiency and effectiveness of error management and control within the CUBRICON user-interface:

Rating		Comments
Excellent	.....	*Making corrections to text input may be easier if user has knowledge of EMACS. Without this knowledge, retyping of whole lines is required as text can't be inserted. This may be, in part, a function of current changes being made to the system.*
Very Good	.....	
Adequate	.....	
Poor	..X..	
Extremely Poor	.....	

-1 The process of making corrections and "on-the-fly" changes during input was straightforward and efficient (1.0.7, 3.1.5.23, 3.5.12). *1*. Circle one: Always, Usually, Sometimes, *Rarely*, Never. Comments:

It was extremely difficult or impossible to change or undo something, such as adjusting points on the flight path. Also, incorrect text due to speech recognition errors was tiresome to correct.

-2 A requirement for an explicit ENTER action prior to CUBRICON processing of user inputs was imposed when necessary to permit user review or reconsideration (1.0.9, 1.4.1, 3.0.5, 3.1.5.25, 3.5.7,). *2*. Circle one: Always, *Usually*, Sometimes, Rarely, Never. Comments:

Text input didn't did not require "enter". Punctuation served as enter. This is not a typical method. Most users will probably be accustomed to the use of the ENTER key.

3.3 Rate how well CUBRICON performs the functions of data protection:

Rating Comments

Excellent .....
Very Good .....
Adequate .....
Poor .....
Extremely Poor ..X..

-1 There was ample *protection* against actions that result in the deletion or significant altering of information (e.g., warnings, undo capability, feedback about results of change prior to action, etc.) (1.3.12, 1.3.13, 2.0.10, 3.5.7, 3.5.8, 3.5.10). *7*. Circle one: Always, Usually, Sometimes, Rarely, *Never*. Comments:

APPENDIX F

Working Paper on Human Factors Issues Related to the Use of Computer Speech Generation

The following working paper describes literature relating to human factors issues of using computer generated speech. It contains general guidelines on when to use, and how to construct, computer generated speech. These guidelines are related to types of messages to be generated by CUBRICON. Specific CUBRICON issues are also discussed.

This paper was delivered to RADC and DARPA earlier in this program. It is included here for completeness.

1. Introduction

This working paper summarizes the results of a review of the literature on computer speech generation. This review was conducted specifically to support the CUBRICON development effort, and the results are reported within the context of CUBRICON needs.

2. General Guidelines for Deciding When and How to Use Speech

This section presents general guidelines for using computer speech generation which applies across all categories of speech output (described below). The literature upon which these guidelines are based are cited.

2.1 When to Use Speech

The following guidelines relate to deciding when to use speech generation.

- Don't interrupt the human user (McCauley, 1984), but allow the user to interrupt the speech generator (Chapanis, 1975).
- Use speech when there is a requirement for rapid two-way exchanges of information (Simpson et al., 1985; Deatherage, 1972).
- Use speech when the information deals with a future time requiring some preparation (Simpson et al., 1985; Deatherage, 1972), and especially when it is intended for immediate use (Simpson et al., 1985).
- Use speech when the message must elicit attention from other tasks or activities (Simpson et al., 1985)*.
- Use speech when eyes and hands are occupied and unavailable for communicating (Simpson et al., 1985; McCauley, 1984), or information must be presented independent of head movement (Simpson et al., 1985)*.

2.2 How to Construct Speech Output

These guidelines relate to the construction of speech output in a way that maximizes efficiency of use and understandability.

- Standardize vocabulary.
  - Use standard and consistent terminology for expressing common concepts (Cooper, 1987; Bucher et al., 1984; Miller et al., 1951).
  - Select terminology that is meaningful to the user population (Smith and Mosier, 1986).

Also applies to auditory signals in general.

  - Use consistent phraseology throughout all parts of the interface (Smith and Mosier, 1986).
  - Coordinate speech output vocabulary with voice recognition capabilities. Users will adapt terminology used by the computer (Zoltan-Ford, 1984).

- Provide supporting context.
  - Put important words near the end of messages. This allows the surrounding sentence structure to provide context to improve intelligibility (Merva and Williges, 1987; McCauley, 1984; Miller et al. 1951).
  - Use multiple syllable words (they have more linguistic redundancy and reduce sequential phonemic uncertainty present in a word (Bucher et al., 1984; Simpson, 1976).

- Make messages as short as possible (Simpson et al., 1985; Zoltan-Ford, 1984).

- Messages of highest priority (i.e., warnings) should be preceded with a beep or other alerting sound (unless computer generated speech is used exclusively for warnings) (Simpson et al., 1985; Simpson and Williams, 1980; Hakkinen and Williges, 1984).

- Communicate message category with (or before) the message (Merva and Williges, 1987). This can be accomplished through the use of different voices or other form of auditory coding (Smith and Goodwin, 1970; Simpson and Williams, 1980) or by selecting message structure in ways that define message category early in the message (e.g., messages describing how to identify information of interest might use a standard form such as: "Displaying mobile SAMS blinking"). This aids in cueing the listener to the importance of messages, and limits the context of the message to improve understandability.

- Coordinate speech output with ongoing tasks (McCauley, 1984).

- Use only one modality when information must be integrated by the user (i.e., use only speech if speech is selected as the best media). Do not use a multimodal expression under this condition (Wickens and Goettle, 1984). For example, if speed and altitude must be used together within a decision-making process, they should be presented in the same modality.

- Provide straightforward method for users to have message repeated. (Smith and Goodwin, 1987).

- Make sure that spoken messages are highly reliable (Simpson et al., 1985).

3. Computer Generated Speech and the CUBRICON Application

3.1 Categories of Generated Speech

Several categories of speech have been defined for use by the CUBRICON system. Rules for generating speech are being defined for each of these categories drawing from applicable literature. The categories are:

- Warnings. Computer initiated presentation of information that requires immediate action and which, if not acted upon, could lead to serious consequences.
- Advisories. Computer initiated presentation of information that is important for the user to know but does not necessarily require immediate action.
- Dialogue. Presentation of information that is part of rapid two way exchanges. The category of "Dialogue" is further defined by the following subcategories:
  - *Query response.* Presentation of information in direct response to a user request. Query responses are always application specific (e.g., How many SAMS are there in the area of interest?).
  - *Dialogue guidance.* Presentation of information to assist in the dialogue itself. Dialogue guidance can be of four types:
    - Feedback. Informing the user that user inputs have been accepted, understood, and so forth.
    - Focus. Providing information about the interpretation or use of presented information (e.g., telling the user where to look).
    - Prompt. Specific direction tot he user about user inputs to facilitate efficient dialogue (e.g., system request for information).
    - Status. Informing the user about the status of the system/dialogue (e.g., "generating new map," "defining new task").

These generated speech outputs can be independent of other media/modalities, or as part of multimedia output, where generated speech provides explanation or enhancement of other outputs. In fact, all CUBRICON outputs can be classed within these categories. It should be noted that dialogue output can fall into more than one of the subcategories.

3.2 A Framework for Defining CUBRICON Speech Generation Logic

This section presents guidelines for deciding when and how to use computer generated speech within the context of the CUBRICON system. These guidelines are defined for each of the speech categories defined above. They are expressed in rule-like fashion to facilitate further specification, and incorporation within the CUBRICON system. References to literature upon which each implementation approach is based are also given. References are cited by number with a key given at the end of the section. Table 1 presents these guidelines.

Table 2 summarizes the approach being implemented in CUBRICON for presenting messages of differing types/priority. Two voices are being used to distinguish between high ard low priority messages. An auditory beep will be used to alert users when warnings are to be issued. Other presentation characteristics that distinguish the three message categories are also summarized in Table 2.

4. References

Bucher, N.M., Voorhees, J., and Werner, E. (1984). Alerting prefixes for speech warning messages. In Proceedings of the National Aerospace and Electronics Conference (NAECON). New York: IEEE Press, 924-931.

Chapanis, A. (1975). Interactive human communication. Scientific American, 232 (3), 36-42

Cooper, M. (1987). Human factor aspects of voice input/output. Speech Technology, March/April 1987.

Davis, G. and Stockton, G. (1982). F-16 voice message system study. In NAECON Proceedings. New York: IEEE Pres, 324-331.

Deatherage, B.H. (1972). Auditory and other sensory forms of information presentation. In H.P. VanCott and R.G. Kinkade (eds.), Human Engineering Guide to Equipment Design. Washington, DC: U.S. Government Printing Office.

Hakkinen, M.T. and Williges, B.H. (1984). Synthesized warning messages: effects of an alerting cue in single- and multiple-function voice synthesis systems. Human Factors, 26 (2), 185-195.

Harvey, D.S. (1988). Talking with airplanes. Air Force Magazine. 88-96.

Luce, P.A., Feustel, T.C., and Pisoni, D.B. (1983). Capacity demands in short-term memory for synthetic and natural speech. Human Factors, 25 (1), 17-32.

McCauley, M.E. (1984). Human factors in voice technology. In F.A. Muckler, A.S. Neal, and L. Strother (eds.) Human Factors Review: 1984. Santa Monica. CA: The Human Factors Society, Inc.

Merva, M.A. and Williges, B.H. (1987). Context, repetition and synthesized speech intelligibility. In Proceedings of the Human Factors Society — 31st Annual Meeting. 961-965.

MIL-STD-1472C. (1981). Military Standard: Human Engineering Design criteria for Military Systems, Equipment and Facilities.

Miller, G.A., Heise, G.A., and Lichten, W. (1951). The intelligibility of speech as a function of the context of the test materials. Journal of Experimental Psychology, 41, 329-335.

Table 1
SUMMARY OF RULES FOR USING GENERATED SPEECH

Category	When to Use	How to Implement
Warnings	Need to communicate information that requires immediate action which if not taken will lead to serious consequences.	• Precede message with alerting sound (e.g., beep). (Ref. 1,2,3) • Use high priority voice (Ref. 2,6). • State message twice (Ref. 4,5,7,9). • Keep message short (Ref. 1,8) (4-5 syllables) (Ref. 1). • Refer to additional information (presented separately from warning). • Interrupt ongoing processes. If voice I/O is underway, break at logical breaking point (tbd). • Augment with visual display (Ref. 1).
Advisories	Need to communicate information that is important to the user, but does not necessarily require immediate action	• Use high priority voice. (Ref. 2,6). • Keep message as short as possible (Ref. 1,8) (no more than 10 words) • Refer to additional information (presented separately from warning). • Present at end of ongoing communication (most recent user request is satisfied) (i.e., don't interrupt, Ref. 10).
Dialogue	—	—
Query Response	User has requested information which is best expressed, at lease in part, using speech.	• Use low priority voice (Ref. 2,6). • Keep message as short as possible (Ref. 1,8). • Provide clear reference to related information on the screen (Ref. 10).
Dialogue Guidance	—	—
Feedback	User has made inputs to the system, and, There is no immediate system response that is perceivable by the user (e.g., requires time for processing, no response required).	• Use low priority voice (Ref. 2,6). • Keep message as short as possible (Ref. 1,8) (no more than 4 or 5 words).
Focus	Expressions require amplifying or orientational information to assure proper understanding or use.	• Use low priority voice. (Ref. 2,6). • Keep message as short as possible (Ref. 1,8) (no more than 10 words). • Provide clear reference to expression being amplified (Ref. 10).
Prompt	It is necessary to solicit information from the user (e.g., to allow completion of process, to clarify user request).	• Use low priority voice (Ref. 2,6). • Keep message as short as possible (Ref. 1,8) (no more than 10 words)
Status	It is necessary to inform the user about the system or dialogue status (e.g., that a new task is being defined, system is performing a task).	• Use low priority voice (Ref. 2,6). • Keep message as short as possible (Ref. 1,8) (no more than 10 words)

Key to References:

1) Simpson et al., 1985
2) Simpson and Williams, 1980
3) Hakkinen and Williges, 1984
4) Davis and Stockton, 1982
5) MIL-STD-1472C
6) Smith and Goodwin, 1970
7) Merva and Williges, 1987
8) Zoltan-Ford, 1984
9) Miller et al., 1951
10) McCauley, 1984

Table 2
CUBRICON APPROACH TO DISTINGUISHING SPEECH CATEGORIES

Category	Priority	Voice	Beep	State Message Twice	Augment With Visual	Interrupt Ongoing Process
Warnings	1	A	Y	Y	Y	Y
Advisories	2	A	—	—	D	—
Dialogue	3	B	—	—	D	—

Key:
A = Voice A (for messages of high relative importance)
B = Voice B (for messages of low relative importance)
1,2,3 = High, medium, and low priority, respectively
Y = Yes
— = No
D = Depends on specifics of message

Schwab, E.C., Nusbaum, H.C., Pisoni, D.B. (1985). Some effects of training on the perception of synthetic speech. Human Factors, 27 (4), 395–408.

Simpson, C.A. (1976). Effects of linguistic redundancy on pilots' comprehension of synthesized speech. In Proceedings of the Twelfth Annual Conference on Manual Control (TMX–73170). Moffett Field, CA: NASA, 294–308.

Simpson, C.A., and Marchionda-Frost, K. (1984). Synthesized speech rate and pitch effects on intelligibility of warning messages for pilots. Human Factors, 26 (5), 509–517.

Simpson, C.A., McCauley, M.E., Roland, E.F., Ruth, J.C., and Williges, B.H. (1985). System design for speech recognition and generation. Human Factors, 27 (2), 115–141.

Simpson, C.A. and Williams, D.H. (1980). Response time effects of alerting tone and semantic context for synthesized voice cockpit warnings. Human Factors, 22 (3), 319–330.

Smith, S.L. and Goodwin, N.C. (1970). Computer-generated speech and man-computer interaction. Human Factors, 12 (2), 215–223.

Smith, S.L. and Mosier, J.N. (1986). Guidelines for Designing User Interface Software. ESD-TR-86-278.

Wickens, C.D. and Goettle, B. (1984). Multiple resources and display formatting: The implications of task integration. In Proceedings of the Human Factors Society 28th Annual Meeting. 722–726.

Zoltan-Ford, E. (1984). Reducing variability in natural language interactions with computers. In Proceedings of the Human Factors Society — 28th Annual Meeting. 768–772.

DISTRIBUTION LIST

addresses	number of copies
RADC/COES ATTN: Ms Sharon M. Walter Griffiss AFB NY 13441-5700	5
Calspan-U of Buffalo Rsch Ctr P.O. Box 400 4455 Genesee Street Buffalo NY 14225	5
RADC/DOVL Technical Library Griffiss AFB NY 13441-5700	1
Administrator Defense Technical Info Center DTIC-FDAC Cameron Station Building 5 Alexandria VA 22304-6145	5
Defense Advanced Research Projects Agency 1400 Wilson Blvd Arlington VA 22209-2308	2
AFCSA/SAMI ATTN: Miss Griffin 10363 Pentagon Washington DC 20330-5425	1
HQ USAF/SCTT Pentagon Washington DC 20330-5190	1
SAF/AQSC Pentagon 4D-267 Washington DC 20330-1000	1

Air Force Info for Industry Office 1
Tri-Service Industry Info Ctr
5001 Eisenhower Ave
Alexandria Va 22333-0001

Air Force Info for Industry Office 1
1030 East Green Street
Pasadena Ca 91106

Air Force Info for Industry Office 1
AFWAL/GLIST (AFIFIO)
Wright-Patterson AFB OH 45433

HQ AFSC/XTS 1
Andrews AFB MD 20334-5000

HQ SAC/SCPT 1
OFFUTT AFB NE 68113-5001

DTESA/RQE 1
ATTN: Mr. Larry G.McManus
Kirtland AFB NM 87117-5000

HQ TAC/DRIY 1
ATTN: Mr. Westerman
Langley AFB VA 23665-5001

HQ TAC/DOA 1
Langley AFB VA 23665-5001

HQ TAC/DRCA 1
Langley AFB VA 23665-5001

ASD/AFALC/AXAE 1
ATTN: W. H. Dungey
Wright-Patterson AFB OH 45433-6533

WRDC/AAAI 1
Wright-Patterson AFB OH 45433-6533

AFIT/LDEE 1
Building 640, Area B
Wright-Patterson AFB OH 45433-6583

WRDC/MLTE 1
Wright-Patterson AFB OH 45433

WRDC/FIES/SURVIAC 1
Wright-Patterson AFB OH 45433

AAMRL/HE 1
Wright-Patterson AFB OH 45433-6573

AUL/LSE 1
Maxwell AFB AL 36112-5564

HQ Air Force SPACECOM/XPYS 1
ATTN: Dr. William R. Matoush
Peterson AFB CO 80914-5001

C3 Division Development Center 2
Marine Corps
Development & Education Command
Code DIOA
Quantico VA 22134-5080

US Army Strategic Defense Command 1
DASD-H-MPL
PO Box 1500
Huntsville AL 35807-3801

Commanding Officer 1
Naval Avionics Center
Library
D/765
Indianapolis IN 46219-2189

Commanding Officer 1
Naval Ocean Systems Center
Technical Library
Code 9642B
San Diego CA 92152-5000

Commanding Officer 1
Naval Weapons Center
Technical Library
Code 3433
China Lake CA 93555-6001

Superintendent 1
Naval Post Graduate School
Code 1424
Monterey CA 93943-5000

Commanding Officer 2
Naval Research Laboratory
Code 2627
Washington DC 20375-5000

Space & Naval Warfare Systems COMM 1
PMW 153-3DP
ATTN: R. Savarese
Washington DC 20363-5100

Commanding Officer 2
US Army Missile Command
Redstone Scientific Info Center
AMSMI-RD-CS-R (Documents)
Redstone Arsenal AL 35898-5241

Advisory Group on Electron Devices 2
Technical Info Coordinator
ATTN: Mr. John Hammond
201 Varick Street - Suite 1140
New York NY 10014

Los Alamos Scientific Laboratory 1
Report Librarian
ATTN: Mr. Dan Baca
PO Box 1663, MS-P364
Los Alamos NM 87545

Rand Corporation 1
Technical Library
ATTN: Ms. Doris Helfer
PO Box 2138
Santa Monica CA 90406-2138

USAG 1
ASH-PCA-CRT
Ft. Huachuca AZ 85613-6000

1839 EIG/EIET 1
ATTN: Mr. Kenneth W. Irby
Keesler AFB MS 39534-6348

JTFPO-TD 1
Director of Advanced Technology
ATTN: Dr. Raymond F. Freeman
1500 Planning Research Drive
McLean VA 22102

HQ ESC/CWPP 1
San Antonio TX 78243-5000

AFEWC/ESRI 3
San Antonio TX 78243-5000

485 EIG/EIR 1
ATTN: M Craft
Griffiss AFB NY 13441-6348

ESD/XTP 1
Hanscom AFB MA 01731-5000

ESD/AVSE 2
Building 1704
Hanscom AFB MA 01731-5000

HQ ESD SYS-2 1
Hanscom AFB MA 01731-5000

Director 1
NSA/CSS
T513/TDL
ATTN: Mr. David Marjarum
Fort George G. Meade MD 20755-6000

Director 1
NSA/CSS
W166
Fort George G. Meade MD 20755-6000

Director 1
NSA/CSS
DEFSMAC
ATTN: Mr. James E. Hillman
Fort George G. Meade MD 20755-6000

Director 1
NSA/CSS
R5
Fort George G. Meade MD 20755-6000

Director 1
NSA/CSS
R8
Fort George G. Meade MD 20755-6000

Director 1
NSA/CSS
S21
Fort George G. Meade MD 20755-6000

Director 2
NSA/CSS
R523
Fort George G. Meade MD 20755-6000

ORA 1
Attn: Dr Richard Kittredge
301-A Dates Drive
Ithaca NY 14850-1313

University of Pennsylvania 1
Attn: Dr Aravino Joshi
Dept of Comp and Info Science
Philadelphia PA 19104

Hewlett-Packard Laboratories 1
Attn: Mr Dan Flickinger
Palo Alto CA 94306

University of Pennsylvania 1
Attn: Dr Mitch Marcus
Dept of Comp and Info Sciences
Philadelphia PA 19104

Honeywell, Inc. 1
Attn: Ms Karen Ryan
CSDD MS 63-B160
1000 Boone Ave N
Minneapolis MN 55427

University of Pennsylvania 1
Attn: Dr Mark Steedman
Dep of Comp and Info Sciences
Philadelphia PA 19104

DARPA/ISTO 1
Attn: Dr Charles Wayne
1400 Wilson Blvd
Arlington VA 20375

Knowledge Models & Cognitive Sys 1
Attn: Dr Henry Hamburger
National Science Foundation
1800 G Street NW
Washington DC 20550

Dr David McDonald 1
9 Whittier Street
Cambridge MA 02140

Language Systems, Inc. 1
Attn: Dr Christine Montgomery
6269 Variel Ave, Suite 200
Woodland Hills CA 91367

SRI 1
Attn: Dr Robert Moore
333 Ravenswood Ave
Menlo Park CA 94025

Calspan Corporation 1
Attn: Dr Jeanette Neal
P.O. Box 400
Buffalo NY 14225

BBN Inc. 1
Attn: Dr Salim Roucos
10 Moulton St
Cambridge MA 02138

USC/ISI 1
Attn: Dr Ed Hovy
4676 Admiralty Way
Marina Del Ray CA 90292

Naval Ocean Systems Center 1
Attn: Ms Beth Sundheim
Code 444
San Diego CA 92152

Navy Ctr for Appl Res in AI 1
Naval Research Laboratory
Attn: Dr Ken Wauchope
Code 5512, 4555 W Overlook Ave SW
Washington DC 20375-5000

University of Pennsylvania 1
Attn: Dr Bonnie Webber
Dept of Comp and Info Sciences
Philadelphia PA 19104

Intelligent Business Systems, Inc. 1
Attn: Dr Richard Cullingford
51 Elm Street
New Haven CT 06510

Courant Institute 1
New York University
Attn: Dr Ralph Grishman
251 Mercer Street
New York NY 10012

Northeastern University 1
College of Computer Science
Attn: Dr Carole D. Hafner
Boston MA 02115

SRI International 1
Attn: Dr Jerry Hobbs
333 Ravenswood Ave
Menlo Park CA 94025

Dragon Systems, Inc. 1
Attn: Mr Jim Baker
90 Bridge St
Newton MA 02158

BBN Inc. 1
Attn: Dr Madelaine Bates
10 Moulton St
Cambridge MA 02138

National Science Foundation 1
Linguistics
Attn: Mr Paul Chapin
1800 G St NW
Washington DC 20550

Cognitive Systems Inc. 1
Attn: Dr Anatole Gershman
234 Church St
New Haven CT 06510

Dr Jonathan Slocum 1
1302 Darter Lane
Austin TX 78746

Texas Instruments 1
Attn: Dr Harry R. Tennant
MS 238
1350 North Central Expressway
Dallas TX 75268

Hewlett-Packard Laboratories 1
Attn: Dr John Nerbonne
Palo Alto CA 94304

BBN Inc. 1
Attn: Dr Ralph Weischedel
10 Moulton St
Cambridge MA 02238