

AD-A165 300

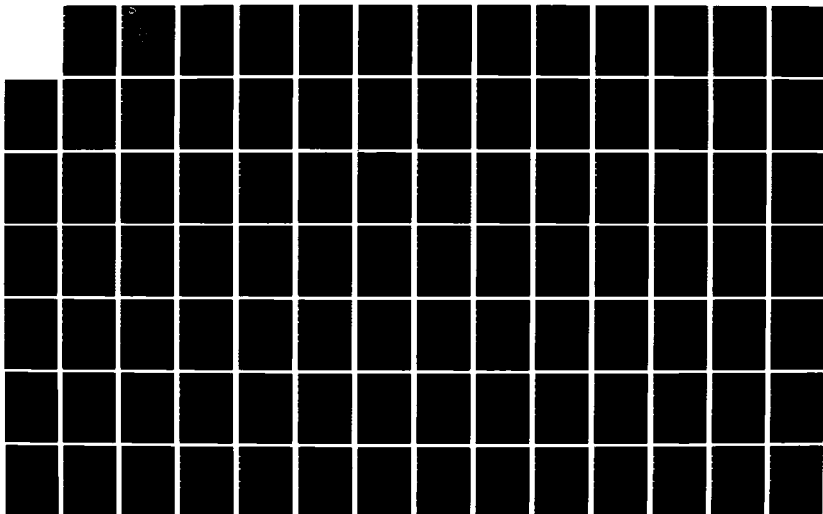
ADA (TRADEMARK) TRAINING CURRICULUM SOFTWARE
ENGINEERING METHODOLOGIES M201 TEACHER'S GUIDE VOLUME 1
(U) SOFTECH INC WALTHAM MA 1986 DAAB07-83-C-K506

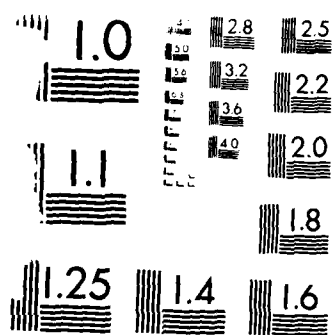
1/5

UNCLASSIFIED

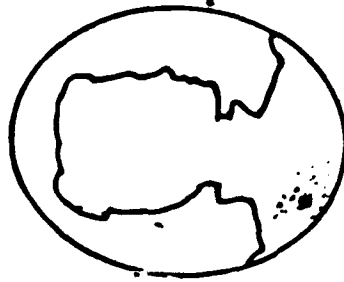
F/G 5/9

NL



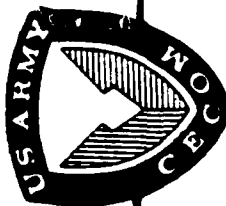


MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A



Ada® Training Curriculum

1986



AD-A165 300

Software Engineering Methodologies

M201

Teacher's Guide

Volume I

DTIC FILE COPY

DTIC
ELECTE

M-R 1 2 1986

B

U.S. Army Communications-Electronics Command
(CECOM)

Contract DAAB07-83-C-K506

Prepared By:

SOFTECH, INC.

460 Totten Pond Road
Waltham, MA 02154

INSTRUCTOR NOTES

- INTRODUCE YOURSELF AND OTHER INSTRUCTOR TO THE CLASS. GIVE CLASS A BRIEF HISTORY OF YOURSELF AS IT RELATES TO THIS COURSE (E.G. METHODOLOGY EXPERIENCE).
- IF CLASS IS OF REASONABLE SIZE (LESS THAN 25), HAVE EACH STUDENT BRIEFLY INTRODUCE THEMSELVES, STATING WHAT THEY HOPE TO GET OUT OF THE COURSE.
- NOTE THAT KNOWLEDGE OF Ada IS NOT REQUIRED. WE WILL BE RELATING THE VARIOUS METHODOLOGY TO Ada AS WE GO ALONG.
- YET Ada WAS DESIGNED TO BE USED WITH A METHODOLOGY. THEREFORE, THIS IS THE MOTIVATION FOR INCLUSION OF THIS MODULE IN THE CURRICULUM.

Copyright by SofTech, Inc. 1984. This material may be reproduced by or for the U.S. Government pursuant to the copyright license under DAR clause 7-104.9(a) (May 81).

Section 1
INTRODUCTION

Accession No.	
REF. NO.	
FILE NO.	
Unreferred	
Justified	
By	
Disc.	
Avail.	
Dist.	A-1

VG 778.1

INSTRUCTOR NOTES

EMPHASIZE THE SURVEY NATURE OF THIS COURSE.

EMPHASIZE THE NEED TO ADAPT THE METHODS TO A PARTICULAR ORGANIZATIONS REQUIREMENTS.

GO OVER THE GOALS CAREFULLY. ASK IF EVERYONE UNDERSTANDS THEM AND IS COMFORTABLE WITH THEM. FIELD CONCERNS, BUT NOTE THAT THE COURSE WAS DESIGNED TO MEET THESE GOALS; IT WILL NOT BE MODIFIED "ON THE FLY," THAT IS, WHILE IT'S BEING GIVEN. STRESS THAT THIS IS AN OVERVIEW COURSE. TRY TO COMMUNICATE THAT THE INSTRUCTORS MAY HAVE NOT USED EVERY METHODOLOGY "ON THE JOB."

GOALS

- BY THE END OF THE COURSE, YOU WILL ...
 - SEE A BALANCED (AND UNBIASED) PICTURE OF THE EXISTING MAJOR SOFTWARE DEVELOPMENT METHODOLOGIES
 - UNDERSTAND THE GENERAL CONCEPTS UNDERLYING SEVERAL METHODOLOGIES
 - UNDERSTAND THEIR SCOPE OF APPLICABILITY WITHIN THE SYSTEM LIFE CYCLE
 - UNDERSTAND WHICH METHOD(S) ARE MORE APPROPRIATE FOR YOUR ORGANIZATION THAN THE OTHER METHODS
- BY THE END OF THE COURSE, YOU WILL NOT ...
 - RECEIVE AN ENDORSEMENT FOR ANY PARTICULAR METHODOLOGY
 - BECOME FLUENT IN ANY PARTICULAR METHODOLOGY
 - SEE EVERY EXISTING METHODOLOGY

[illegible][illegible][illegible]

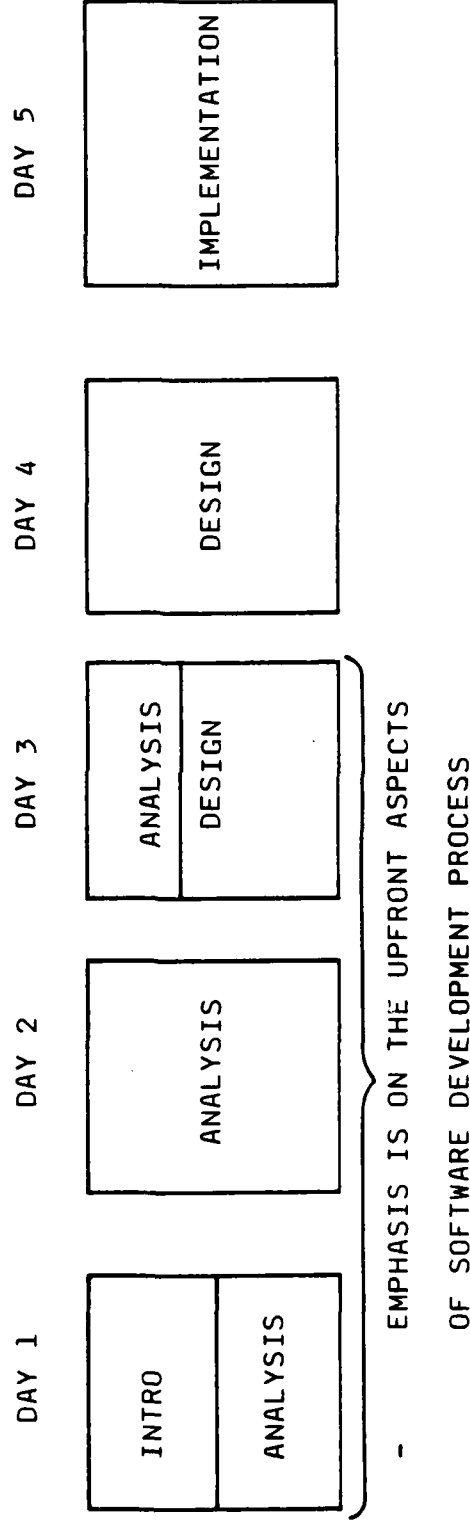
- [illegible]

[illegible]

COURSE ORGANIZATION

- THIS COURSE IS ORGANIZED ACCORDING TO GENERAL SOFTWARE LIFE-CYCLE ACTIVITIES:

- GIVING A BRIEF TREATISE ON EACH METHODOLOGY



INSTRUCTOR NOTES

GO OVER THIS AND THE OUTLINE VERY BRIEFLY. TELL CLASS IT'S FOR THEIR INFORMATION, AND THEY CAN REFER TO IT WHEN THEY WANT.

COURSE CONTENT,
(DAY 1)

SECTION

- 1: INTRODUCTION -
COURSE GOALS,
COURSE ORGANIZATION
COURSE CONTENTS
CLASS PARTICIPATION
(DAY 1 - 15 MINUTES)
- 2: SOFTWARE ENGINEERING(SE),
DEFINITIONS
MOTIVATION FOR SOFTWARE ENGINEERING
(DAY 1 - 20 MINUTES)
- 3: THE SOFTWARE LIFE CYCLE,
DEFINITION AND SCOPE
THE LIFE OF SOFTWARE
(DAY 1 - 20 MINUTES)
- 4: SOFTWARE ENGINEERING METHODOLOGIES,
THE SCOPE OF OUR CONTROL OF THE PROCESS
ATTRIBUTES OF METHODOLOGIES
WHY LEARN METHODOLOGIES
ASPECTS OF AN IDEAL METHODOLOGY
THE COURSE'S VIEW OF METHODOLOGIES
RELATIONSHIP OF Ada AND SE METHODOLOGIES
(DAY 1 - 40 MINUTES)
- 5: ANALYSIS INTRODUCTION,
DEFINITION
REQUIREMENTS ANALYSIS
DOD-STD-SDS VIEW OF ANALYSIS
ANALYSIS PERSPECTIVES AND FORMATS
METHODOLOGIES TO BE COVERED
SUMMARY
(DAY 1 - 40 MINUTES)
- 6: SADT METHODOLOGY,
OVERVIEW
GRAPHIC NOTATION AND CONCEPTS
SAMPLE APPLICATION
SADT EXTENSIONS
EXERCISE 1
(DAY 1 - 120 MINUTES)

6: 1-4

(DAY 1 - 90 MINUTES)
1-3

INSTRUCTOR NOTES

VG 778.1

1-41

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

COURSE CONTENT

(DAY 2)

SECTION

7:

SREM METHODOLOGY,
KEY CONCEPTS AND OVERVIEW
APPLICATION OF SREM METHODS
A SREM EXAMPLE
SREM EXTENSIONS

(DAY 2 - 150 MINUTES)

8:

ENTITY DIAGRAMMING
DESCRIPTION OF THE REFERENCE EXAMPLE
OVERVIEW
KEY CONCEPTS WITH EXAMPLES
DIAGRAM SYNTAX
BACHMAN EXTENSIONS TO BASIC SYNTAX
CONVERSION OF ANNOTATED DIAGRAMS
EXERCISE 2

(DAY 2 - 90 MINUTES)

9:

PSL/PSA
OVERVIEW AND INTRODUCTION
PROBLEM STATEMENT LANGUAGES(PSL)
PSA REPORT TYPES
PSL/PSA SUMMARY

(DAY 2 - 40 MINUTES)

10:

STRUCTURED SYSTEMS ANALYSIS METHODS,
OVERVIEW
DEMARCO DATA FLOW DIAGRAMS
DEMARCO DATA DICTIONARY
GANE AND SARSON PICTURES
GANE AND SARSON DATA DICTIONARY

(DAY 2 - 20 MINUTES)

INSTRUCTOR NOTES

VG 778.1

1-51

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

COURSE CONTENT

(DAY 3)

SECTION

- 11: SOFTWARE COST REDUCTION PROJECT(SCRP) METHODOLOGY (DAY 3 - 90 MINUTES)
BACKGROUND
PRINCIPLES OF THE METHODOLOGY
BENEFITS
UNDERLYING CONCEPTS
REQUIREMENTS SPECIFICATION TECHNIQUES
- 12: ANALYSIS WRAP-UP (DAY 3 - 40 MINUTES)
MAIN POINTS TO CONSIDER
ANALYSIS PHASE COVERAGE BY METHODOLOGY
METHODOLOGY EVALUATION CRITERIA
COMPARISON OF ANALYSIS METHODOLOGIES
RELATIONSHIP OF Ada AND ANALYSIS METHODOLOGY FEATURES
- 13: DESIGN INTRODUCTION (DAY 3 - 50 MINUTES)
WHAT IS DESIGN
ROLE OF THE DESIGNER
ARCHITECTURAL DESIGN TECHNIQUES
GUIDELINES FOR ARCHITECTURAL DESIGN
ARCHITECTURAL DESIGN AND DOD-STD-SDS
DETAILED DESIGN TECHNIQUES
DETAILED DESIGN AND DOD-STD-SDS
DESIGN METHODOLOGY PERSPECTIVES AND FORMATS
SUMMARY OF METHODOLOGIES TO BE COVERED

INSTRUCTOR NOTES

COURSE CONTENT

(DAY 3) (Continued)

SECTION

(14)

SOFTWARE COST REDUCTION PROJECT(SCRIP) DESIGN METHODS (DAY 3 - 60 MINUTES)
OVERVIEW
ROLE OF ABSTRACTION, MODULARITY AND HIDING
KEY CONCEPTS
SCRIP HISTORY AND USAGE
SCRIP VIEW OF ABSTRACTION, MODULARITY AND HIDING
SCRIP DESIGN DECOMPOSITION
SCRIP DESIGN METHOD DOCUMENTATION
SCRIP WRAPUP

(15)

OBJECT-ORIENTED DESIGN (DAY 3 - 30 MINUTES)
OVERVIEW
OBJECT-ORIENTED DESIGN AND DATA ABSTRACTION
DESIGN CONCEPTS BY EXAMPLE
SUMMARY
EXERCISE (DAY 3 - 60 MINUTES)

INSTRUCTOR NOTES

COURSE CONTENT

(DAY 4)

SECTION

- (16) STRUCTURED DESIGN METHODOLOGY
OVERVIEW
KEY CONCEPTS
(DAY 4 - 60 MINUTES)
- (17) JACKSON METHODOLOGY
OVERVIEW
JACKSON STRUCTURED PROGRAMMING(JSP)
SUMMARY
EXERCISE 4 (DAY 4 - 60 MINUTES)
(DAY 4 - 40 MINUTES)
- (18) WARNIER-ORR METHOD;
OVERVIEW
WARNIER DIAGRAM SYNTAX
DESIGN STRATEGIES
COMPLETING THE DESIGN
(DAY 4 - 20 MINUTES)
- (19) HIGHER ORDER SOFTWARE(HOS) METHOD;
OVERVIEW
FUNDAMENTAL BASIS FOR HOS
SYNTAX
A SIMPLE EXAMPLE
A RADAR SYSTEM EXAMPLE
AUTOMATION OF HOS
(DAY 4 - 30 MINUTES)
- (20) ARCHITECTURAL DESIGN METRICS;
QUALITY FACTORS FOR A DESIGN
COUPLING
COHESION
DESIGN HEURISTICS
(DAY 4 - 60 MINUTES)

COURSE CONTENT

(DAY 4) (Continued)

SECTION

(21)

PROGRAM DESIGN LANGUAGES(PDL),
OVERVIEW

ROLE OF Ada AS A PDL

RATIONALE FOR A PDL

PDL REQUIREMENTS

MAPPING PDL FEATURES TO DESIGN FEATURES

AN Ada PDL USAGE SAMPLER

SUMMARY

(DAY 4 - 40 MINUTES)

(22)

GRAPHICAL DETAILED DESIGN METHODS

HIPO OVERVIEW

NASSI-SCHNEIDERMAN STRUCTURED FLOWCHARTS

(DAY 4 - 20 MINUTES)

(23)

DESIGN WRAP-UP

DESIGN PHASE COVERAGE

COMPARISONS OF THE DESIGN METHODS

SUPPORT OF Ada CONCEPTS AND FEATURES

REMINDERS

(DAY 4 - 30 MINUTES)

INSTRUCTOR NOTES

COURSE CONTENT
(DAY 5)

SECTION

(24)

(DAY 5 - 40 MINUTES)

IMPLEMENTATION INTRODUCTION,
SCOPE OF THE IMPLEMENTATION PHASE
IMPLEMENTATION ISSUES
DOD-STD-SDS VIEW OF IMPLEMENTATION
IMPLEMENTATION PERSPECTIVES AND FORMATS
IMPLEMENTATION METHODOLOGIES TO BE COVERED

(25)

(DAY 5 - 30 MINUTES)

STRUCTURED PROGRAMMING;
MOTIVATION
DEFINITION
SCOPE
CONTROL STRUCTURING GUIDELINES
Ada AND STRUCTURED PROGRAMMING
SUMMARY

(26)

(DAY 5 - 40 MINUTES)

PROGRAM COMPLEXITY MANAGEMENT;
MOTIVATION
COMPLEXITY MANAGEMENT TECHNIQUES AND EXAMPLE
SUMMARY
EXERCISE 5 (DAY 5 - 60 MINUTES)

(27)

PROGRAM CORRECTNESS;
DEFINITION AND MOTIVATION
BUILDING IN CORRECTNESS
CORRECTNESS CONCEPTS
AN EXAMPLE
PROGRAM CORRECTNESS FROM AN OPTIMIST'S VIEW
PROGRAM CORRECTNESS FROM AN PESSIMIST'S VIEW
PROGRAM CORRECTNESS FROM AN PRAGMATIST'S VIEW
Ada AND PROGRAM CORRECTNESS
SUMMARY

INSTRUCTOR NOTES

VG 778.1

1-101

4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

COURSE CONTENT

(DAY 5) (Continued)

SECTION

28:	TESTING APPROACHES SCOPE RELATIONSHIP OF TESTING AND OTHER ERROR REMOVAL TECHNIQUES TESTING AND PROGRAM CHARACTERISTICS RELATIONSHIP UNIT TESTING	(DAY 5 - 60 MINUTES)
29:	METHODS OF REVIEW, COMMON QUESTIONS ASKED ABOUT REVIEWS ANSWERS TO THE QUESTIONS	(DAY 5 - 40 MINUTES)
30:	IMPLEMENTATION WRAP-UP, PERSONNEL TRAITS COMPARISON OF TECHNIQUES REMINDERS	(DAY 5 - 20 MINUTES)
31:	COURSE WRAP-UP, OPEN DISCUSSION FORMAT (NO LECTURE)	(DAY 5 - 60 MINUTES)

INSTRUCTOR NOTES

- THIS COURSE MODULE ENCOURAGES CLASS PARTICIPATION AND SHOULDN'T BE INSTRUCTOR DRIVEN. ENCOURAGE QUESTIONS.
- "FEEL" IS BETTER THAN "WORKING KNOWLEDGE."
- VISUALIZING HELPS GIVE A FEEL FOR A METHODOLOGY.
- TOO MANY GRAPHICS: DON'T TRY TO LEARN THEM ALL!

PARTICIPATION

- CLASS PARTICIPATION TAKES THESE FORMS:
 - NOTES
 - EXERCISES ARE DONE INDIVIDUALLY BY EACH STUDENT; ANSWERS ARE PRESENTED BY THE INSTRUCTOR AFTER AN APPROPRIATE TIME PERIOD.
 - DISCUSSIONS ARE INTENDED TO STIMULATE CLASS DISCUSSIONS; ALSO, EXAMPLES, EXERCISES, AND NOTES MAY STIMULATE TALK.
- TRY TO GET A FEEL FOR EACH METHODOLOGY
- TRY VISUALIZING THE PRODUCT OF EACH METHODOLOGY
- IT WILL BE HARD TO KEEP ALL THE GRAPHICS STRAIGHT IN YOUR MIND; DON'T TRY!

1
 2
 3
 4
 5
 6
 7
 8
 9
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 40
 41
 42
 43
 44
 45
 46
 47
 48
 49
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
 100
 101
 102
 103
 104
 105
 106
 107
 108
 109
 110
 111
 112
 113
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154
 155
 156
 157
 158
 159
 160
 161
 162
 163
 164
 165
 166
 167
 168
 169
 170
 171
 172
 173
 174
 175
 176
 177
 178
 179
 180
 181
 182
 183
 184
 185
 186
 187
 188
 189
 190
 191
 192
 193
 194
 195
 196
 197
 198
 199
 200
 201
 202
 203
 204
 205
 206
 207
 208
 209
 210
 211
 212
 213
 214
 215
 216
 217
 218
 219
 220
 221
 222
 223
 224
 225
 226
 227
 228
 229
 230
 231
 232
 233
 234
 235
 236
 237
 238
 239
 240
 241
 242
 243
 244
 245
 246
 247
 248
 249
 250
 251
 252
 253
 254
 255
 256
 257
 258
 259
 260
 261
 262
 263
 264
 265
 266
 267
 268
 269
 270
 271
 272
 273
 274
 275
 276
 277
 278
 279
 280
 281
 282
 283
 284
 285
 286
 287
 288
 289
 290
 291
 292
 293
 294
 295
 296
 297
 298
 299
 300
 301
 302
 303
 304
 305
 306
 307
 308
 309
 310
 311
 312
 313
 314
 315
 316
 317
 318
 319
 320
 321
 322
 323
 324
 325
 326
 327
 328
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 368
 369
 370
 371
 372
 373
 374
 375
 376
 377
 378
 379
 380
 381
 382
 383
 384
 385
 386
 387
 388
 389
 390
 391
 392
 393
 394
 395
 396
 397
 398
 399
 400
 401
 402
 403
 404
 405
 406
 407
 408
 409
 410
 411
 412
 413
 414
 415
 416
 417
 418
 419
 420
 421
 422
 423
 424
 425
 426
 427
 428
 429
 430
 431
 432
 433
 434
 435
 436
 437
 438
 439
 440
 441
 442
 443
 444
 445
 446
 447
 448
 449
 450
 451
 452
 453
 454
 455
 456
 457
 458
 459
 460
 461
 462
 463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479
 480
 481
 482
 483
 484
 485
 486
 487
 488
 489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525

2-i

VG 778.1

REFERENCES: J.D. MUSA (EDITOR), "STIMULATING SOFTWARE ENGINEERING PROGRESS - A REPORT OF THE SOFTWARE ENGINEERING PLANNING GROUP", ACM SIGSOFT SOFTWARE ENGINEERING NOTES, VOL 8, NO. 2 APRIL 1983

Section 2
SOFTWARE ENGINEERING

INSTRUCTOR NOTES

NO COMMON, SIMPLE DEFINITION FOR SOFTWARE ENGINEERING.

SEEWG = SOFTWARE ENGINEERING ENVIRONMENT WORKING GROUP, WORKING GROUP TO DEFINE
REQUIREMENTS FOR A NAVY STANDARD SOFTWARE ENGINEERING ENVIRONMENT.

SOME DEFINITIONS

- "SOFTWARE ENGINEERING IS THE APPLICATION OF SCIENCE AND MATHEMATICS BY WHICH THE CAPABILITIES OF COMPUTER EQUIPMENT ARE MADE USEFUL TO MAN VIA COMPUTER PROGRAMS, PROCEDURES AND ASSOCIATED DOCUMENTATION."

BEOHM 1981

- "... A SOFTWARE ENGINEERING PROCESS IS A SET OF ACTIVITIES FOR DEVELOPING AND MODIFYING SOFTWARE THROUGH IT'S LIFE CYCLE"

SEEWG REPORT 1982

- "A METHODOLOGY IS A REPEATABLE HUMAN PROCEDURE WHICH SUPPORTS SOME ASPECT OF AN ACTIVITY."

SEEWG REPORT 1982

INSTRUCTOR NOTES

THIS IS THE DEFINITION OF THE SOFTWARE CRISIS THAT MOTIVATED THE DEVELOPMENT OF ADA.

GIVE SOME PERSONAL EXAMPLES OF SYSTEMS YOU HAVE WORKED ON OR ASK THE CLASS FOR SOME.

MOTIVATION FOR SOFTWARE ENGINEERING
(SOFTWARE CRISIS)

- SOFTWARE FOR COMPLEX MILITARY SYSTEMS
 - IS USUALLY LATE
 - COSTS MORE THAN ORIGINALLY ESTIMATED
 - DOES NOT WORK TO ORIGINAL SPECIFICATIONS
 - IS UNRELIABLE
 - IS DIFFICULT AND COSTLY TO MAINTAIN

INSTRUCTOR NOTES

TALK TO EACH BULLET, GIVING THE CLASS PERSONAL EXPERIENCE OR TRY TO GET THEM TO RELATE THEIR EXPERIENCES.

NOTE: DURING THIS SECTION YOU ARE TRYING TO SET THE STAGE FOR LATER PARTICIPATION BY THE CLASS. MAKE THEM FEEL AS THEY "OWN" THESE PROBLEMS.

MOTIVATION FOR SOFTWARE ENGINEERING

(ADDITIONAL PROBLEMS)

- SOFTWARE IS NOT REUSABLE ON DIFFERENT SYSTEMS
- PROLIFERATION OF METHODS, LANGUAGES AND ARCHITECTURES
- METHODS AND LANGUAGES NOT SUITED FOR CURRENT APPLICATIONS
- SUPPLY OF QUALITY SOFTWARE PERSONNEL NOT ABLE TO MEET
CURRENT SOFTWARE DEMAND
- SOFTWARE TASKS ARE MORE COMPLEX NOW, BUT NO WIDELY USED
METHODS AND TOOLS TO DEAL WITH THE PROBLEM EXIST
- LACK OF ADEQUATE MANAGEMENT AND SOFTWARE DEVELOPMENT
METHODS/TOOLS

INSTRUCTOR NOTES

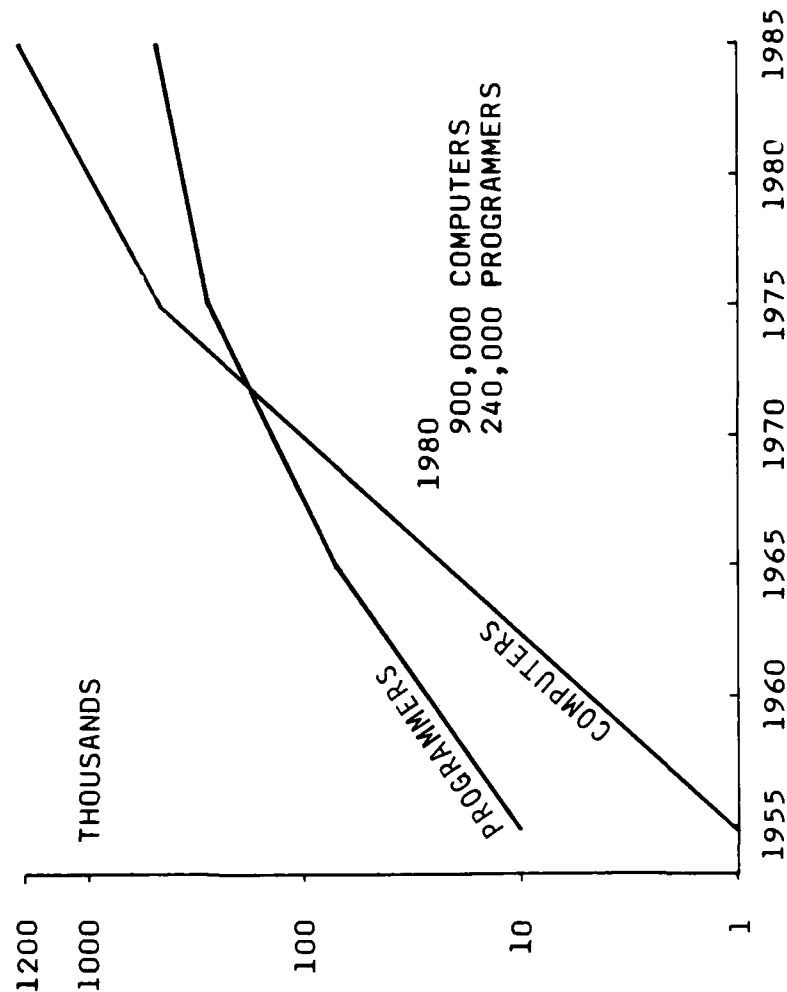
THE KEY POINT IS TO NOTE THE RATE OF GROWTH.

ENVIRONMENT FACING SOFTWARE ENGINEERING

• DEMAND FOR SOFTWARE OUTSTRIPPING DEVELOPMENT CAPABILITY

TOTAL U.S. DATA PROCESSING INDUSTRY EXPENDITURES

Year	Expenditure (billions of 1970 dollars)	Percent of GNP
1970	21	2.1
1975	41	3.2
1980	82	5.2
1985	164	8.3



• PRODUCTIVITY ADVANCES ARE NOT KEEPING UP

SOURCE: REPORT OF THE SOFTWARE ENGINEERING GROUP, 1983

VG 778.1

2-4

INSTRUCTOR NOTES

PRODUCTIVITY IS A MEASURE OF HOW EFFICIENTLY WE CAN PRODUCE SOFTWARE (NOTE THE SLOW RATE OF IMPROVEMENT).

PRODUCTION IS A MEASURE OF HOW MUCH THE DEMAND FOR SOFTWARE HAS INCREASED.

RELATIVE PROGRAMMER PRODUCTIVITY AND TOTAL U.S. YEARLY
CODE PRODUCTION (NORMALIZED TO 1955)

YEAR	PRODUCTIVITY	PRODUCTION
1955	1	1
1960	1.6	5
1965	2.0	16
1970	2.3	38
1975	2.7	59
1980	3.1	85
1985	3.6	119

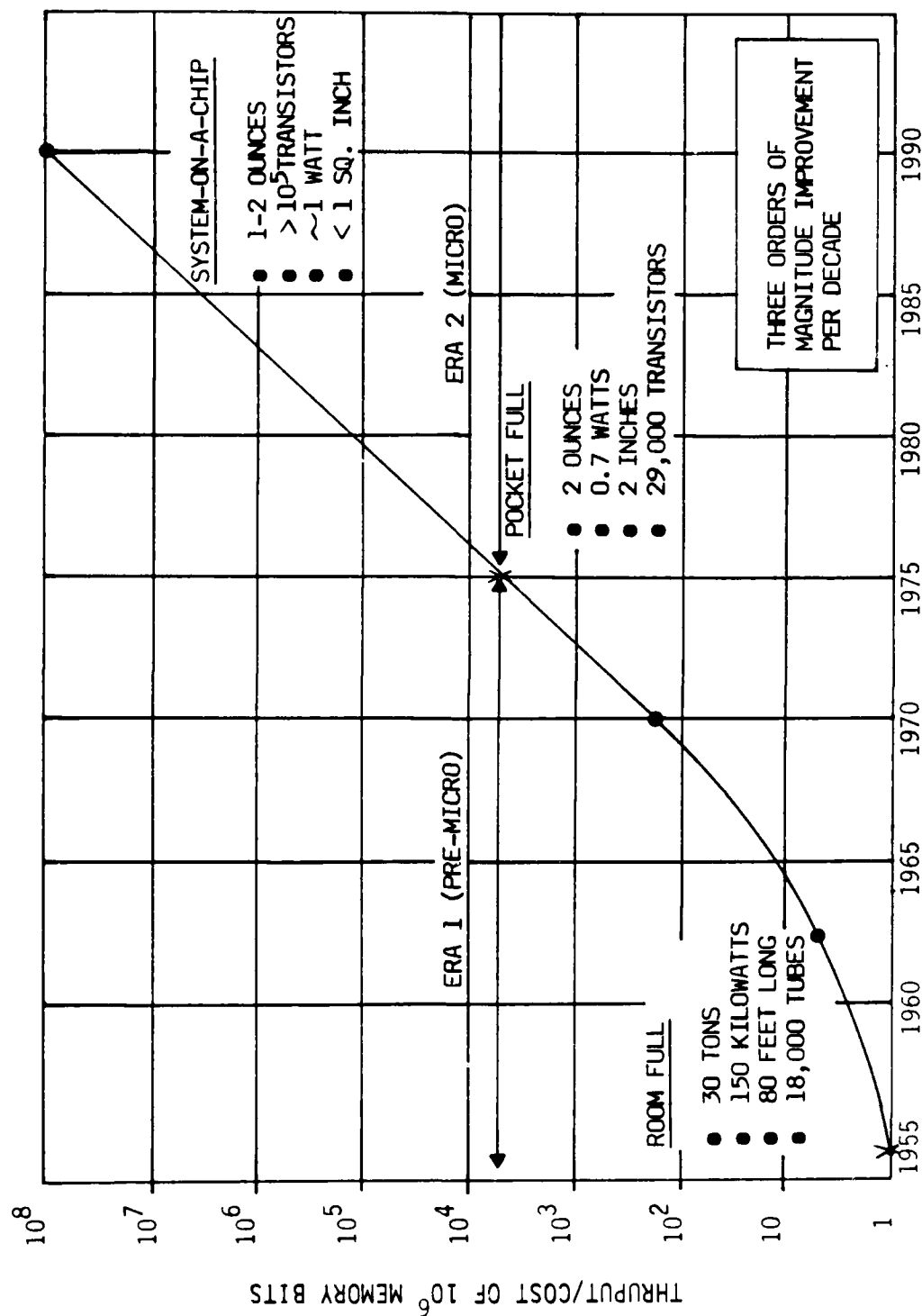
- WE HAVE A SERIOUS DEFICIENCY
 - NEED MORE SOFTWARE DEVELOPERS
 - MUST MAKE EXISTING ONES MORE PRODUCTIVE

INSTRUCTOR NOTES

POINT OUT THAT HARDWARE TECHNOLOGY IS GROWING SO FAST THAT WE DON'T KNOW HOW TO PROPERLY UTILIZE IT.

NOTE THE Y AXIS IS A WAY OF MEASURING THE COMBINED EFFECT OF FASTER HARDWARE TECHNOLOGIES AND HIGHER DENSITY MICRO-CIRCUIT TECHNOLOGIES.

COMPUTER HARDWARE TECHNOLOGY
RELATIVE COST EFFECTIVENESS

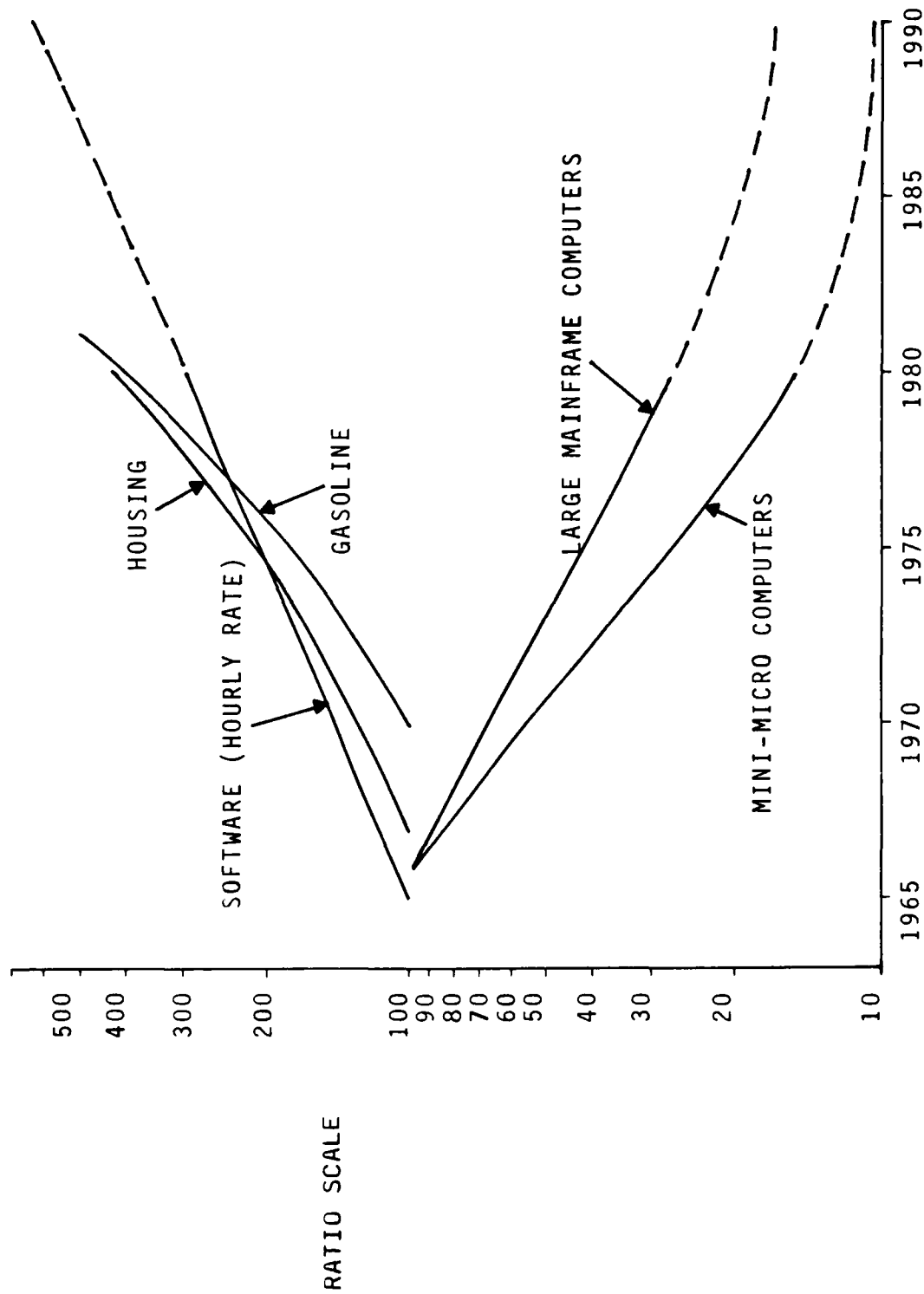


SOURCE: REPORT OF THE SOFTWARE ENGINEERING GROUP, 1983

VG 778.1

2-6

SOFTWARE VS. HARDWARE COST TRENDS



SOURCE: REPORT OF THE SOFTWARE ENGINEERING GROUP, 1983

VG 778.1

2-7

INSTRUCTOR NOTES

THEME: THE SOFTWARE DEVELOPMENT PROCESS CAN BE EXPRESSED IN TERMS OF A SEQUENCE OF PHASES.

PURPOSE: TO PROVIDE THE FRAMEWORK IN WHICH THE METHODOLOGIES WILL BE PRESENTED IN.

REFERENCE: "A SOFTWARE ENGINEERING ENVIRONMENT FOR THE NAVY", REPORT OF THE NAVMAT SOFTWARE ENGINEERING ENVIRONMENT WORKING GROUP, MARCH 1983.

Section 3
THE SOFTWARE LIFE CYCLE

INSTRUCTOR NOTES

THE CONCEPT OF A SOFTWARE DEVELOPMENT LIFE-CYCLE HAS EVOLVED OVER THE LAST 20 YEARS. IT IS VIEWED DIFFERENTLY BY DIFFERENT INDIVIDUALS, ORGANIZATIONS, AND INDUSTRIES, OFTEN BECAUSE THE TYPES OF PRODUCTS THAT ARE PRODUCED ARE RADICALLY DIFFERENT OR BECAUSE OF THE IMPORTANCE OR NON-IMPORTANCE OF THE SOFTWARE USED IN THE PRODUCT DEVELOPMENT.

SOFTWARE LIFE CYCLE

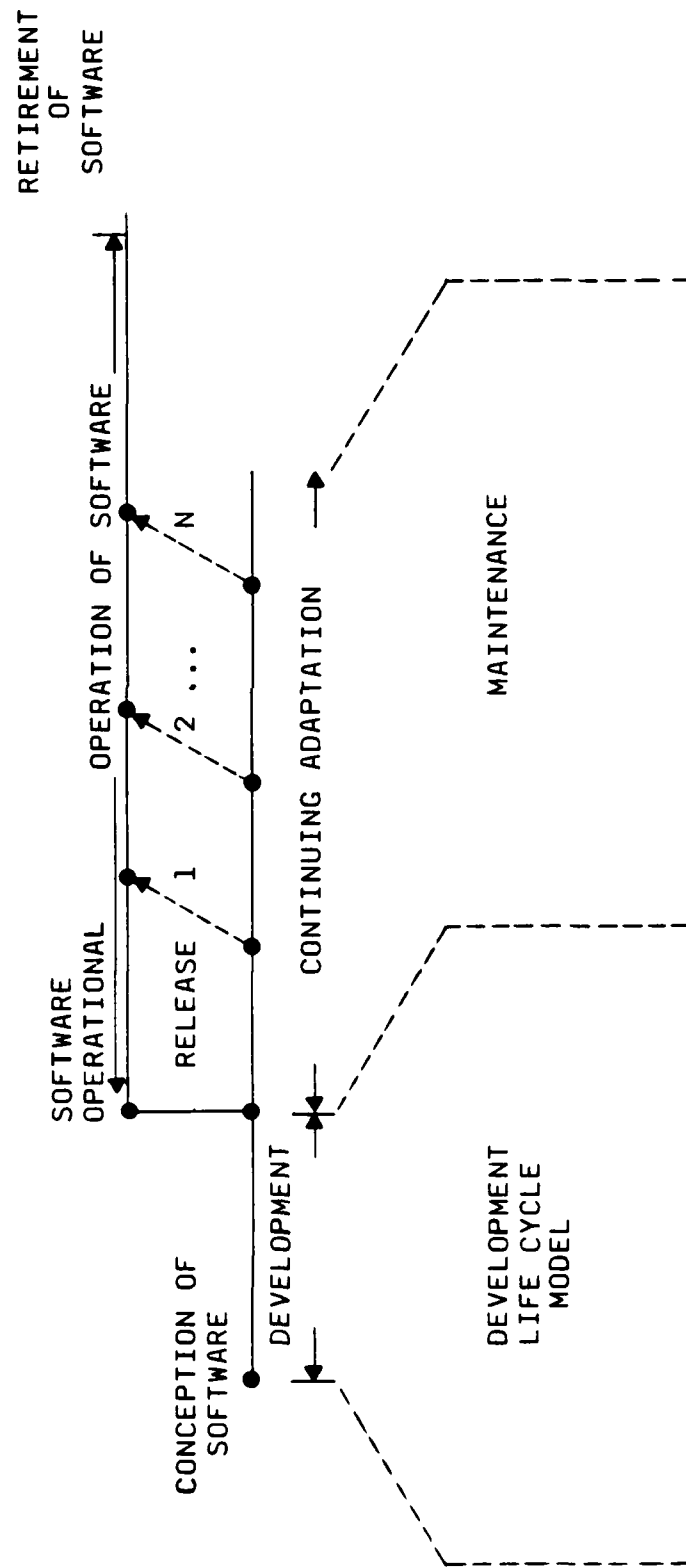
- LIFE CYCLE ORGANIZES THE ACTIVITIES OF BUILDING SOFTWARE
- SOFTWARE DEVELOPMENT IS BROKEN INTO PHASES
- LIFE CYCLE SUMMARIZES SOFTWARE DEVELOPMENT
- THE "LIFE CYCLE" IS NOT STANDARD IN THE INDUSTRY

INSTRUCTOR NOTES

THE TERM "LIFE-CYCLE" PROVIDES US WITH A WAY OF LOOKING AT THE TASKS OF BUILDING A SYSTEM. IT IS NEEDED SO THAT WE CAN TALK ABOUT ASPECTS OF SOFTWARE DEVELOPMENT IN A COMMON LANGUAGE, AND SO THAT WE CAN DETERMINE THE EFFECTS OF CHANGES TO OUR DEVELOPMENT PROCESS.

NOTE THAT SOFTWARE HAS A LIFE OF IT'S OWN THAT EXTENDS WELL BEYOND THE CODING OF AN APPLICATION.

THE LIFE OF SOFTWARE



SOURCE: SEEWG REPORT, 1983

INSTRUCTOR NOTES

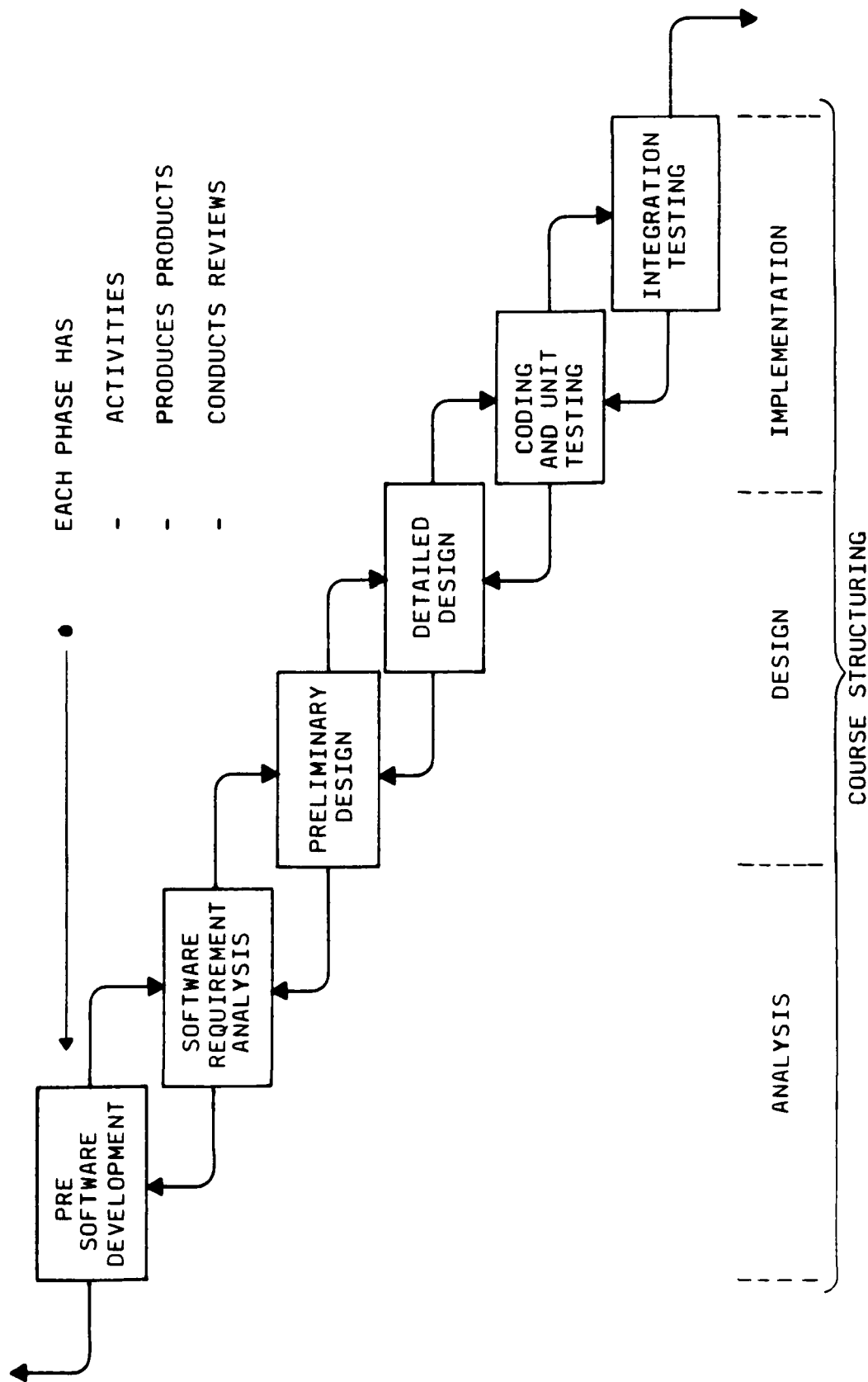
THIS IS REALLY DIVIDING UP THE RESPONSIBILITY OF THE SYSTEM DEVELOPMENT AMONG DIFFERENT GROUPS TO MAKE THE DEVELOPMENT PROCESS TRACTABLE. SOME DIVIDE IT UP INTO 6 PIECES, SOME INTO 4 OR 8 PIECES.

NOTE THE MAPPING TO THE COURSE STRUCTURE. THIS PICTURE IS A MORE CLASSICAL VIEW OF LIFE-CYCLE. THIS MAY (AND PROBABLY WILL) DIFFER FROM THE LIFE-CYCLE MODEL UNDERSTOOD BY EACH STUDENT. THIS IS OFTEN CALLED THE "WATERFALL" MODEL. OUT OF EACH PHASE A DELIVERABLE IS USUALLY PRODUCED, WHICH IS USED AS INPUT INTO THE NEXT PHASE. A REAL SYSTEM OR SOFTWARE DEVELOPMENT IS NEVER THIS CLEAN. THERE EXIST FEEDBACK LOOPS BETWEEN PHASES, THE DELIVERABLES NEED REVIEWS AND CHANGES BEFORE ACCEPTANCE, THE DESIGN PHASE MAY BE SUB-DIVIDED INTO MANY DESIGN PHASES OF INCREASED DETAIL, ETC.

THE KEY WORDS ARE:

- ANALYSIS - THE "WHAT" PHASE
- DESIGN - THE "HOW" PHASE
- IMPLEMENTATION - THE "BUILD" PHASE

DEVELOPMENT LIFE CYCLE MODEL



INSTRUCTOR NOTES

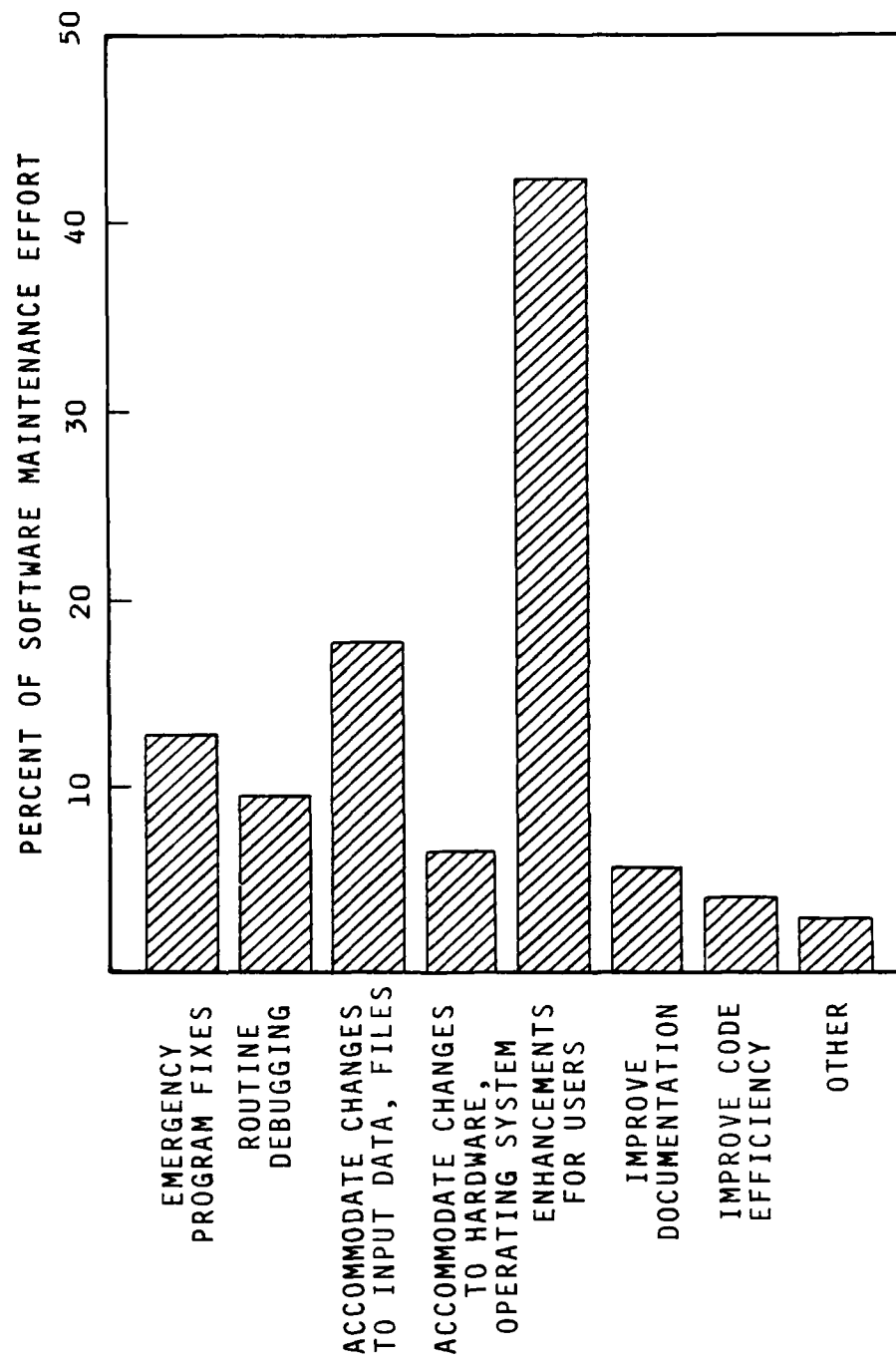
MAINTENANCE ISN'T THE BEST TERM FOR WHAT GOES ON AFTER DEVELOPMENT, SOME PEOPLE USE THE

WORD

"EVOLUTION"

DUE TO THE EMPHASIS ON ENHANCEMENTS ETC.

SOFTWARE MAINTENANCE ACTIVITIES



SOURCE: BEOHM, SOFTWARE ENGINEERING ECONOMICS, 1983

INSTRUCTOR NOTES

POINT OUT THAT NO MODEL CAN TOTALLY REPRESENT THE REAL WORLD.

THE CRITICAL LIMITATION IS BULLET 3 - CORRECTNESS ANALYSIS.

SHORTCOMINGS OF THIS MODEL

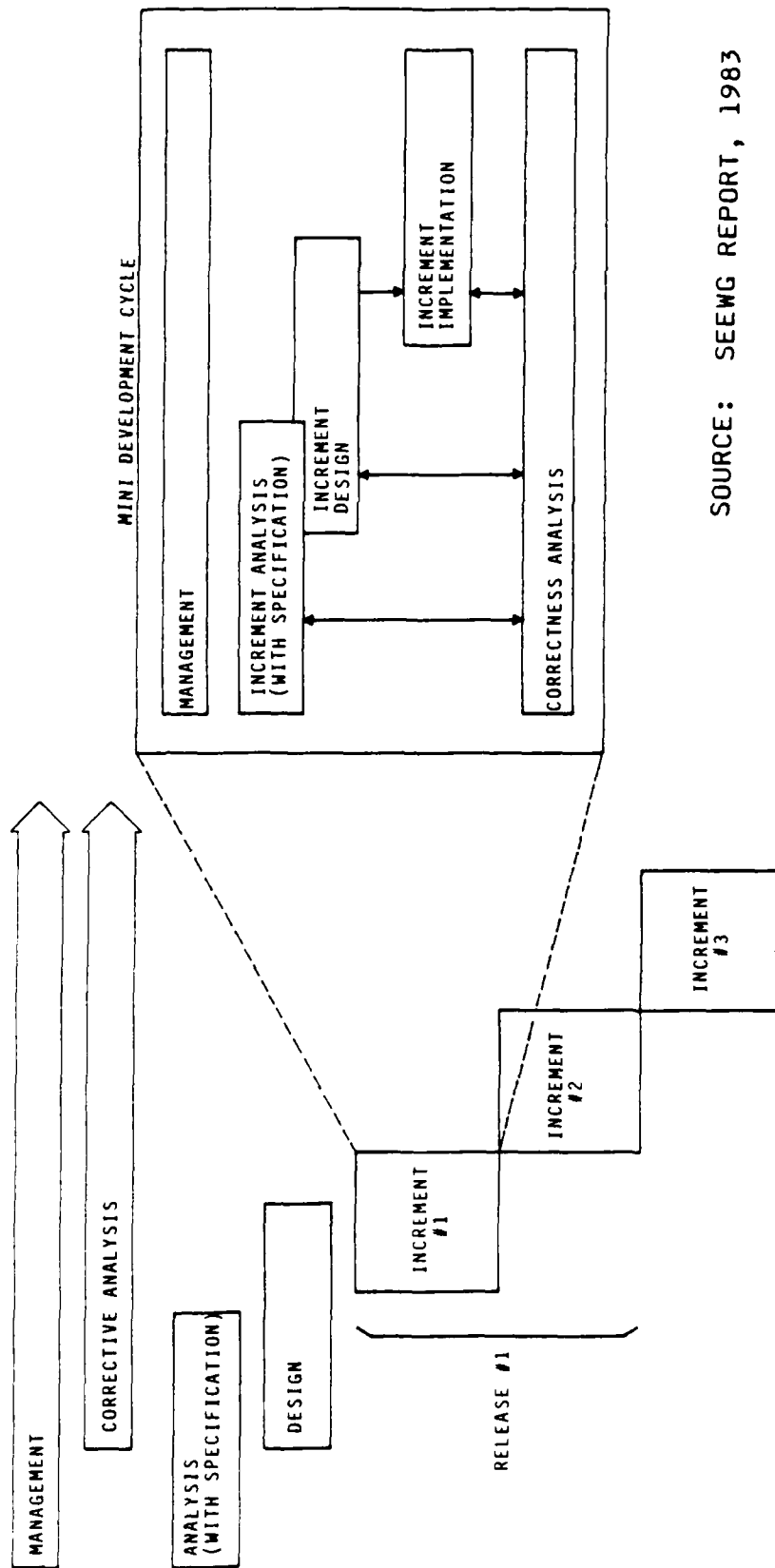
- FINITE ACTIVITY PHASE ARE NOT REALISTIC
 - MOST ACTIVITIES DO NOT HAVE A CLEARLY DEFINED START OR END POINT
- STATIC NATURE OF THE OUTPUTS OF A PREVIOUS ACTIVITY
 - NO OUTPUT CAN BE CONSIDERED AS COMPLETE AND WILL NEED SOME MODIFICATION IN THE NEXT ACTIVITY
 - FEEDBACK PATH HELP TO A LIMITED DEGREE
- CORRECTNESS ANALYSIS IS PRIMARILY DONE IN TESTING ACTIVITIES
- MANAGEMENT IS NOT EXPLICITLY SHOWN

INSTRUCTOR NOTES

EMPHASIZE CONTINUOUS MANAGEMENT AND CORRECTNESS ANALYSIS.

A MODEL THAT ADDRESSES THE SHORTCOMINGS

- INCREMENTAL DEVELOPMENT
 - BUILD SOFTWARE SYSTEM IN SMALL MANAGEABLE INCREMENTS
 - EACH INCREMENT ADDS NEW FUNCTIONS TO THE SYSTEM



SOURCE: SEEWG REPORT, 1983

INSTRUCTOR NOTES

PURPOSE: TO MOTIVATE FOLLOWING SECTIONS.

REFERENCE: "Ada METHODOLOGIES: CONCEPTS AND REQUIREMENTS", (METHODMAN DOCT.) DoD Ada
JOINT PROGRAM OFFICE, NOV. 1982.

Section 4
SOFTWARE ENGINEERING METHODOLOGIES

VG 778.1

INSTRUCTOR NOTES

THIS PICTURE ILLUSTRATES VERY ABSTRACTLY WHAT THE SOFTWARE DEVELOPMENT PROCESS IS ALL ABOUT

"TAKING A SET OF NEEDS AND PRODUCING A SOFTWARE SYSTEM THAT MEETS THE NEED UNDER VARIOUS CONTROLS AND CONSTRAINTS

METHODOLOGIES GUIDE THE WAY THE PROCESS IS PERFORMED.

OUR SCOPE OF CONTROL

MILITARY STANDARDS,
PROCEDURES, BUDGETS

CONTROL/
CONSTRAINTS



EFFECTIVE AND
RELIABLE
SOFTWARE
(OR SYSTEM)

NEEDS

METHODOLOGIES AND TOOLS ARE THE ONLY THINGS WE CAN AFFECT

INSTRUCTOR NOTES

GO OVER EACH POINT CAREFULLY. THIS SETS THE STAGE FOR THE STUDENTS. IT GIVES THEM
ADDITIONAL WAYS TO LOOK AT THE METHODOLOGIES WE WILL BE DISCUSSING IN THIS COURSE.

ATTRIBUTES OF METHODOLOGIES

- A WELL-CONCEIVED METHODOLOGY HAS
 - CREATIVE ASPECTS
 - INTELLECTUAL ASPECTS
 - CLERICAL ASPECTS
 - MECHANICAL ASPECTS
- GOOD SOFTWARE ENGINEERING METHODOLOGIES SHOULD
 - IMPROVE EFFECTIVENESS AND PRODUCTIVITY OF SOFTWARE DEVELOPMENT ACTIVITIES
 - RESULT IN THE CREATION OF RELIABLE SOFTWARE
 - FIT TOGETHER TO FORM AN INTEGRATED SET OF METHODS
 - SEPARATE THE CREATIVE ASPECTS FROM THE MECHANICAL ASPECTS
 - PROMOTE AUTOMATION OF THE CLERICAL ASPECTS OF SOFTWARE DEVELOPMENT

WHY LEARN METHODOLOGIES?

(QUALITY VIEWPOINT)

- INCREASED EFFORT IN THE EARLIER ACTIVITIES OF A DEVELOPMENT WILL BE REFLECTED IN REDUCED COSTS FOR TESTING AND MAINTENANCE*
 - PREVENT ERRORS FROM BEING INTRODUCED
 - DETECT ANY ERRORS AT EARLIEST POSSIBLE TIME
- BY APPLYING A SET OF METHODOLOGIES YOU WILL ACHIEVE HIGHER QUALITY SOFTWARE THAT FULFILLS THE NEEDS

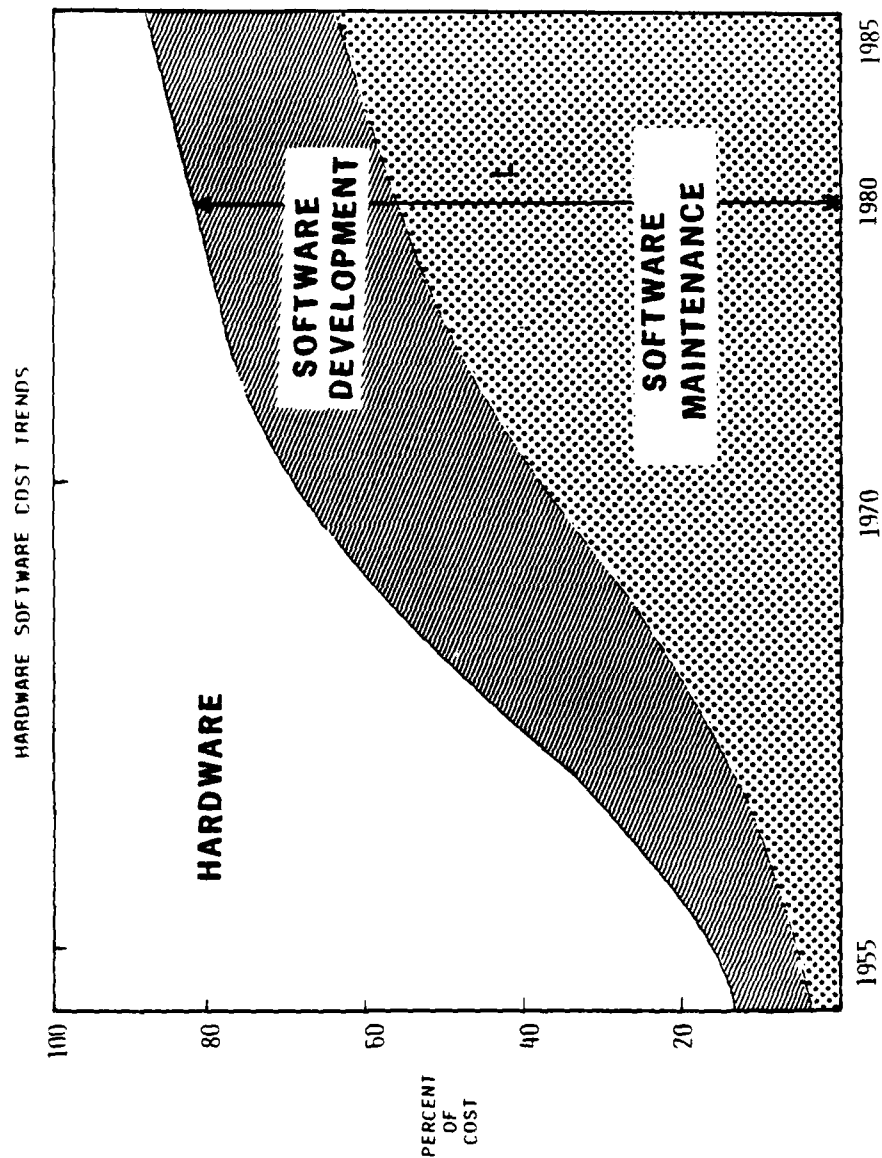
*REPAIR, ADAPTATION AND ENHANCEMENT THAT OCCURS AFTER INITIAL DEVELOPMENT.

INSTRUCTOR NOTES

THIS SLIDE IS CLOSELY LINKED WITH THE PREVIOUS ONE. RE-EMPHASIZE THE IMPORTANCE OF METHODOLOGIES TO MINIMIZE MAINTENANCE COSTS IN THE FUTURE.

WHY LEARN METHODOLOGIES? (COST VIEWPOINT)

- BY 1985 COMPUTER AND INFORMATION PROCESSING WILL REPRESENT 8.3% OF THE GNP



- IN 1980 L $\approx$ \$40,000,000,000

VG 778.1

INSTRUCTOR NOTES

TELL THE CLASS THAT THEY WILL SEE THE DETAILS OF THE DoD-STD-SDS REQUIREMENTS IN THE INTRODUCTORY SECTIONS TO ANALYSIS, DESIGN AND IMPLEMENTATION.

WHY LEARN METHODOLOGIES

(CONSTRAINT VIEWPOINT)

- FUTURE SOFTWARE DEVELOPMENTS WILL BE SO COMPLEX THAT WITHOUT A WELL DEFINED SET OF METHODS THEY WILL BE IMPOSSIBLE TO ACCOMPLISH
- NEW MILITARY STANDARDS ARE REQUIRING THE USE OF A METHODOLOGY
 - DoD-STD-SDS
- THE POOL OF EXPERIENCED AND QUALIFIED SOFTWARE DEVELOPERS IS LIMITED
 - USE METHODOLOGIES TO IMPROVE PRODUCTIVITY

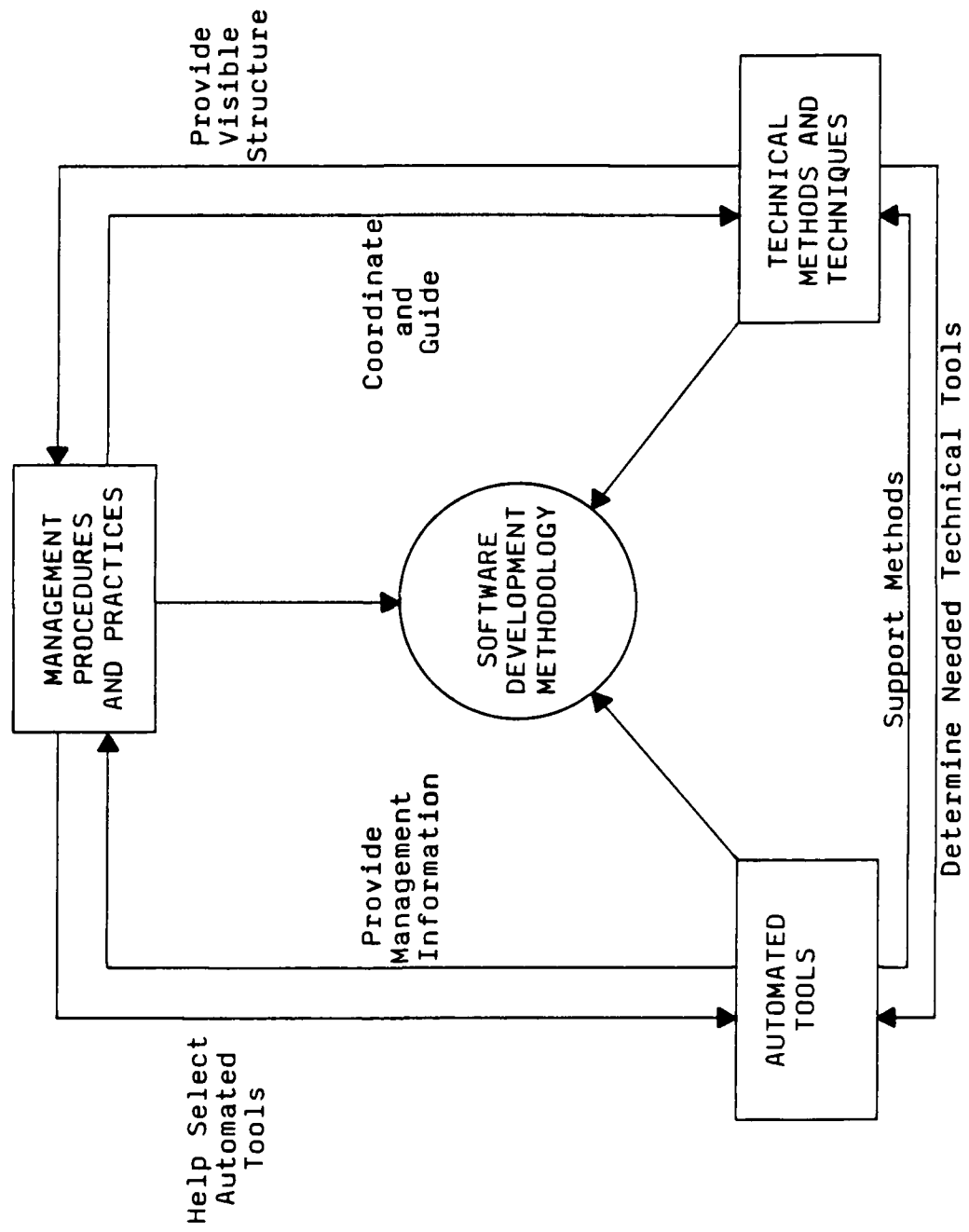
INSTRUCTOR NOTES

NOTE THAT ALL THREE ASPECTS MUST BE ADDRESSED IN A METHODOLOGY.

WALKTHROUGH THE DIAGRAM FROM METHODS, TO MANAGEMENT PRACTICES THEN AUTOMATED TOOLS.

EMPHASIZE THAT THIS COURSE FOCUSES ON THE TECHNICAL METHODS.

ASPECTS OF A METHODOLOGY (IDEAL VIEW)



SOURCE: METHODMAN, 1982

INSTRUCTOR NOTES

THE NEXT TWO SLIDES CHARACTERIZE AN IDEAL DEVELOPMENT METHODOLOGY WHICH WE COULD USE AS A MODEL TO MEASURE THE METHODOLOGIES WE WILL COVER.

WE USE THIS IN THE WRAP-UPS FOR ANALYSIS AND DESIGN TO COMPARE THE VARIOUS METHODS WE WILL DISCUSS.

REQUIREMENTS FOR A SOFTWARE DEVELOPMENT METHODOLOGY

(IDEAL VIEW)

- A METHODOLOGY SHOULD:
 - COVER THE ENTIRE DEVELOPMENT PROCESS, SIMPLIFYING TRANSITIONS BETWEEN PROJECT PHASES
 - ENHANCE COMMUNICATION AMONG ALL INTERESTED PERSONS AT ALL STAGES OF DEVELOPMENT
 - SUPPORT PROBLEM ANALYSIS AND UNDERSTANDING
 - SUPPORT BOTH TOP-DOWN AND BOTTOM-UP APPROACHES TO SOFTWARE DEVELOPMENT
 - SUPPORT SOFTWARE VALIDATION AND VERIFICATION THROUGH THE DEVELOPMENT PROCESS
 - FACILITATE THE CAPTURE OF DESIGN, IMPLEMENTATION, AND PERFORMANCE CONSTRAINTS

INSTRUCTOR NOTES

REQUIREMENTS FOR A SOFTWARE DEVELOPMENT METHODOLOGY

(IDEAL VIEW CONTINUED)

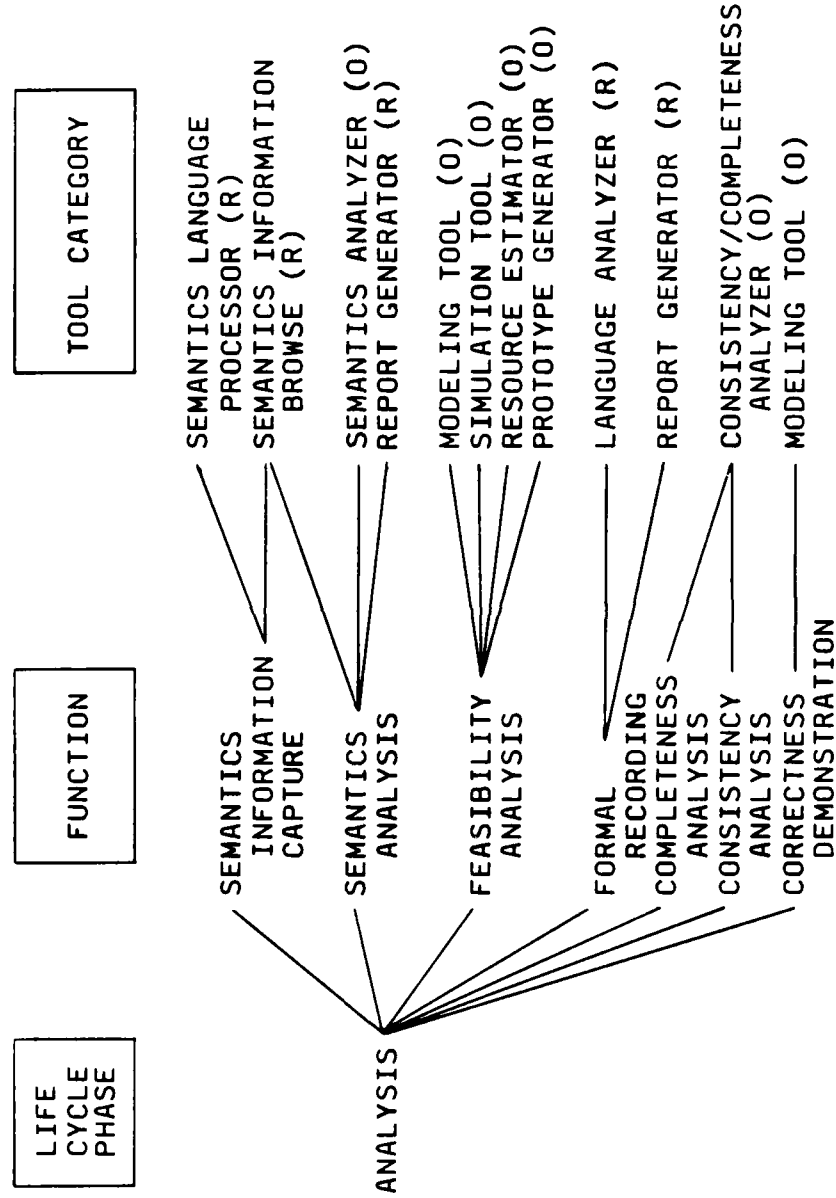
- A METHODOLOGY ALSO SHOULD:
 - SUPPORT THE SOFTWARE DEVELOPMENT ORGANIZATION
 - SUPPORT THE EVOLUTION OF A SYSTEM THROUGHOUT ITS EXISTENCE
 - BE SUPPORTED BY AUTOMATED AIDS
 - MAKE THE EVOLVING SOFTWARE PRODUCT VISIBLE AND CONTROLLABLE AT ALL STAGES OF DEVELOPMENT
 - BE TEACHABLE AND TRANSFERABLE
 - BE OPEN-ENDED

INSTRUCTOR NOTES

THE NEXT FIVE (5) SLIDES SHOW THE RELATIONSHIP BETWEEN A LIFE CYCLE PHASE AND THE TYPES OF FUNCTIONS ONE IS INVOLVED IN AS WELL AS THE TYPES OF AUTOMATED TOOLS ONE NEEDS TO SUPPORT THE FUNCTIONS. THIS IS STILL LOOKING AT THE "IDEAL". EMPHASIZE THAT THIS IS PRESENTED TO GET THE CLASS TO THINK ABOUT THE ISSUES ASSOCIATED WITH SOFTWARE ENGINEERING METHODOLOGIES.

ASPECTS OF METHODOLOGIES

(ANALYSIS PHASE)



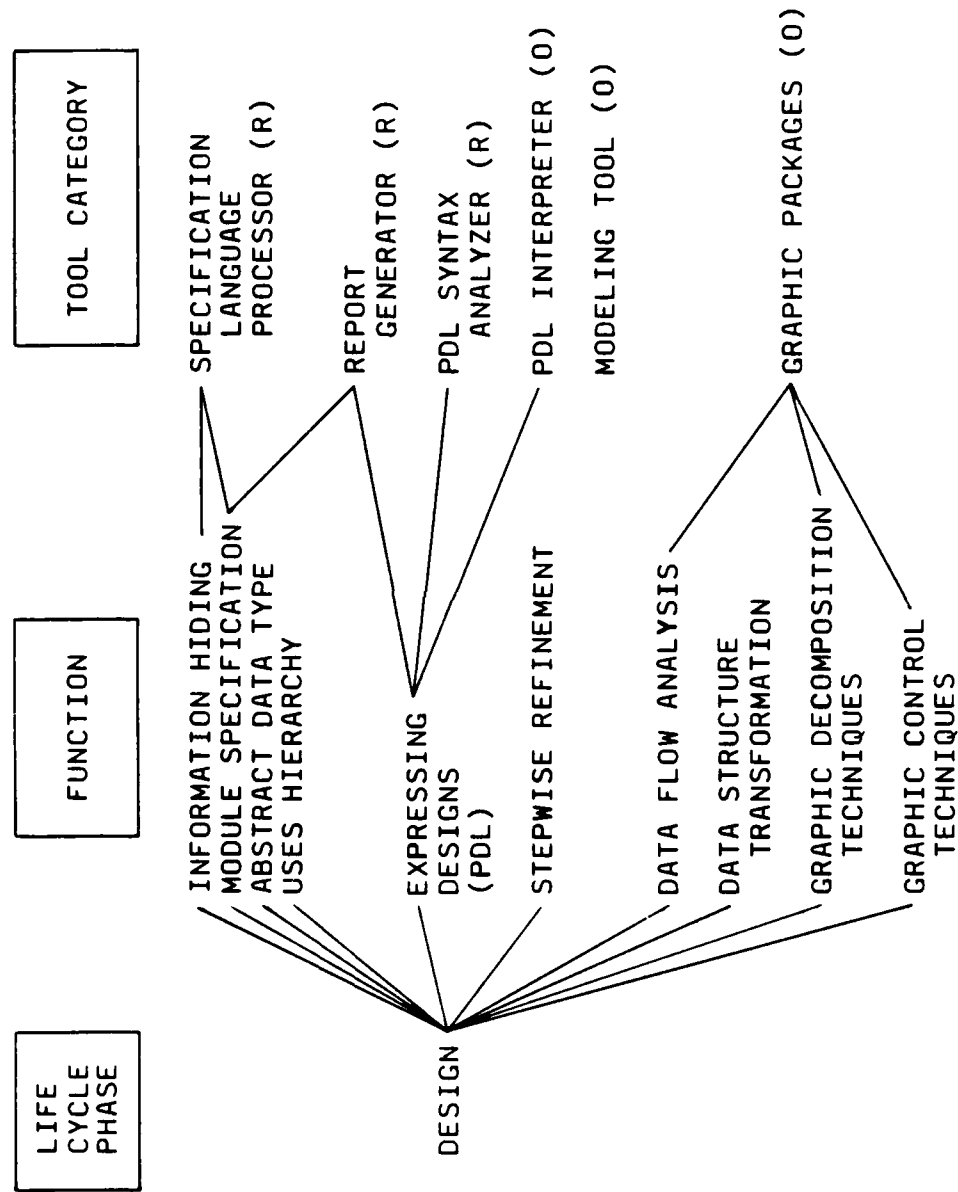
(R) - REQUIRED
(O) - OPTIONAL

SOURCE: SEEWG REPORT, 1983

VG 778.1

INSTRUCTOR NOTES

ASPECTS OF METHODOLOGIES (DESIGN PHASE)



SOURCE: SEEWG REPORT, 1983

VG 778.1

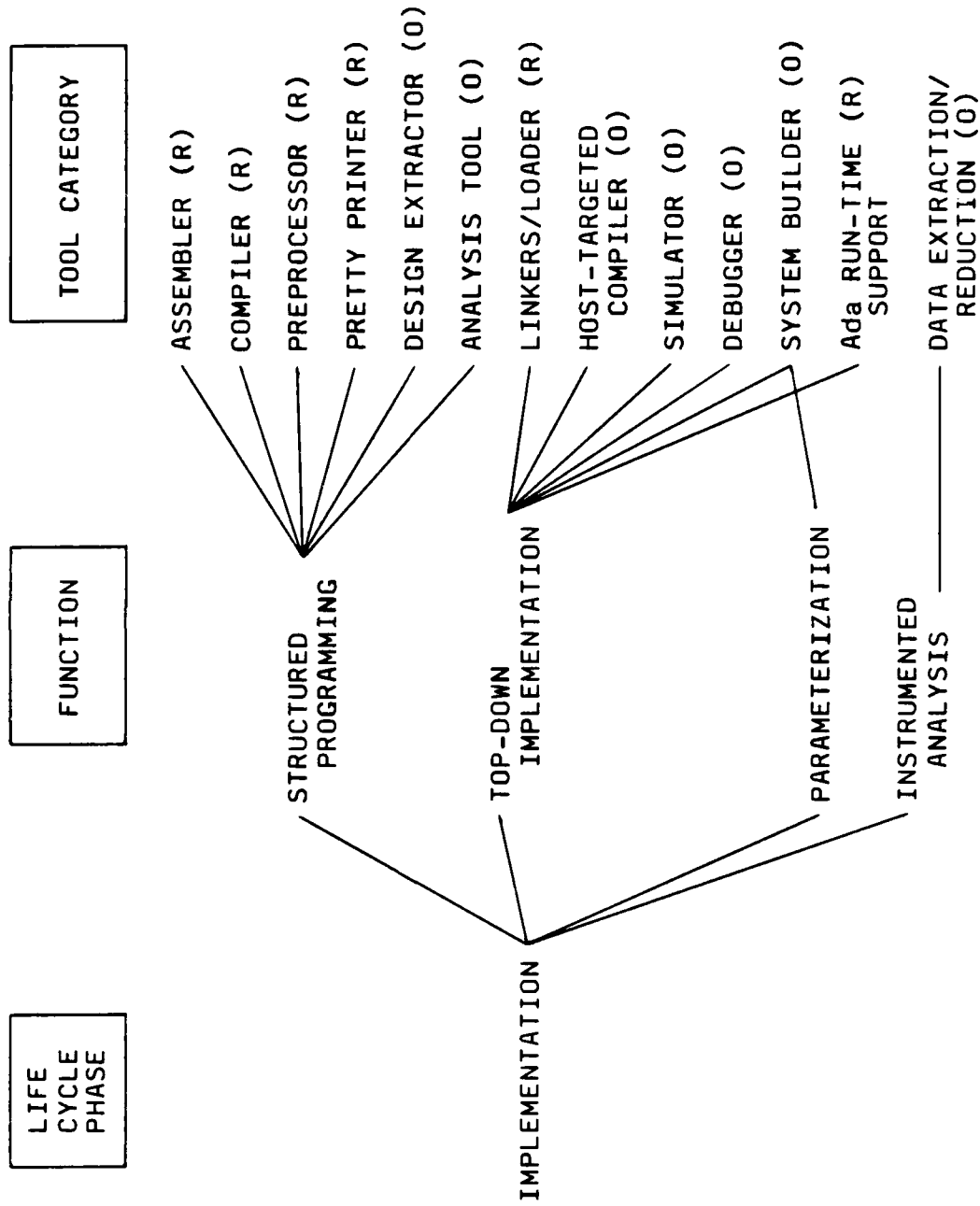
INSTRUCTOR NOTES

VG 778.1

4-111

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

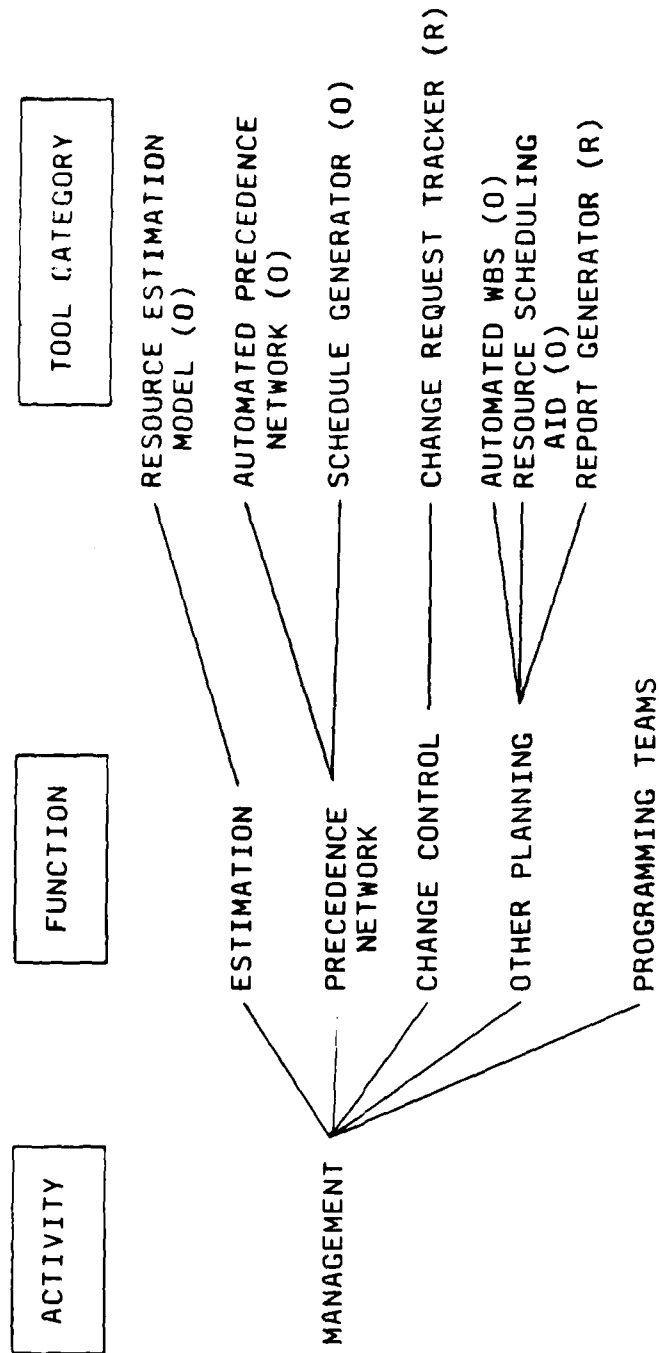
ASPECTS OF METHODOLOGIES (IMPLEMENTATION PHASE)



SOURCE: SEEWG REPORT, 1983

INSTRUCTOR NOTES

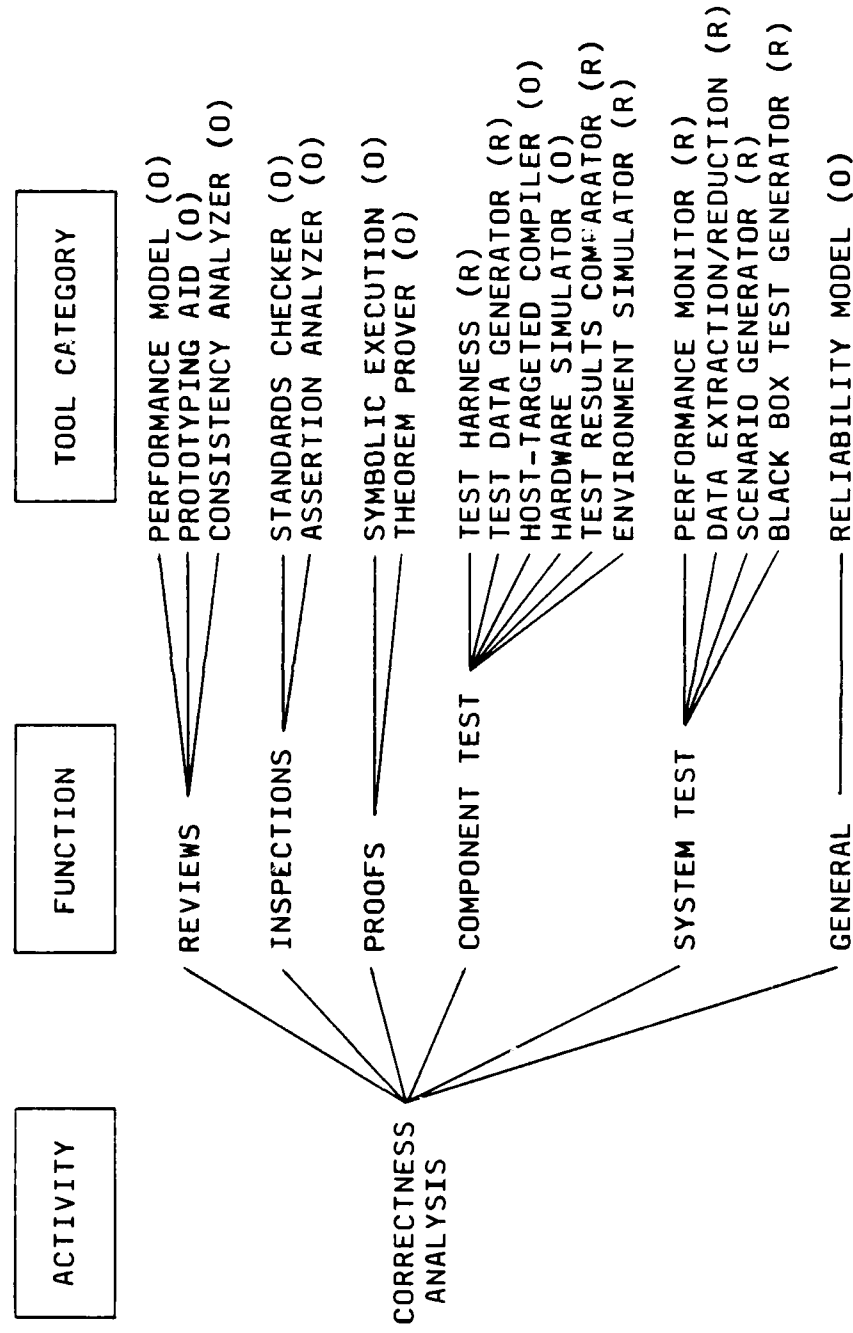
ASPECTS OF METHODOLOGIES (MANAGEMENT)



SOURCE: SEEWG REPORT, 1983

INSTRUCTOR NOTES

ASPECTS OF METHODOLOGIES (CORRECTNESS ANALYSIS)



SOURCE: SEEWG REPORT, 1983

INSTRUCTOR NOTES

EMPHASIZE TO THE CLASS THAT THEY WILL HAVE TO COMBINE METHODS TO MAKE ANY OF THEM EFFECTIVE.

METHODOLOGY COURSE VIEW

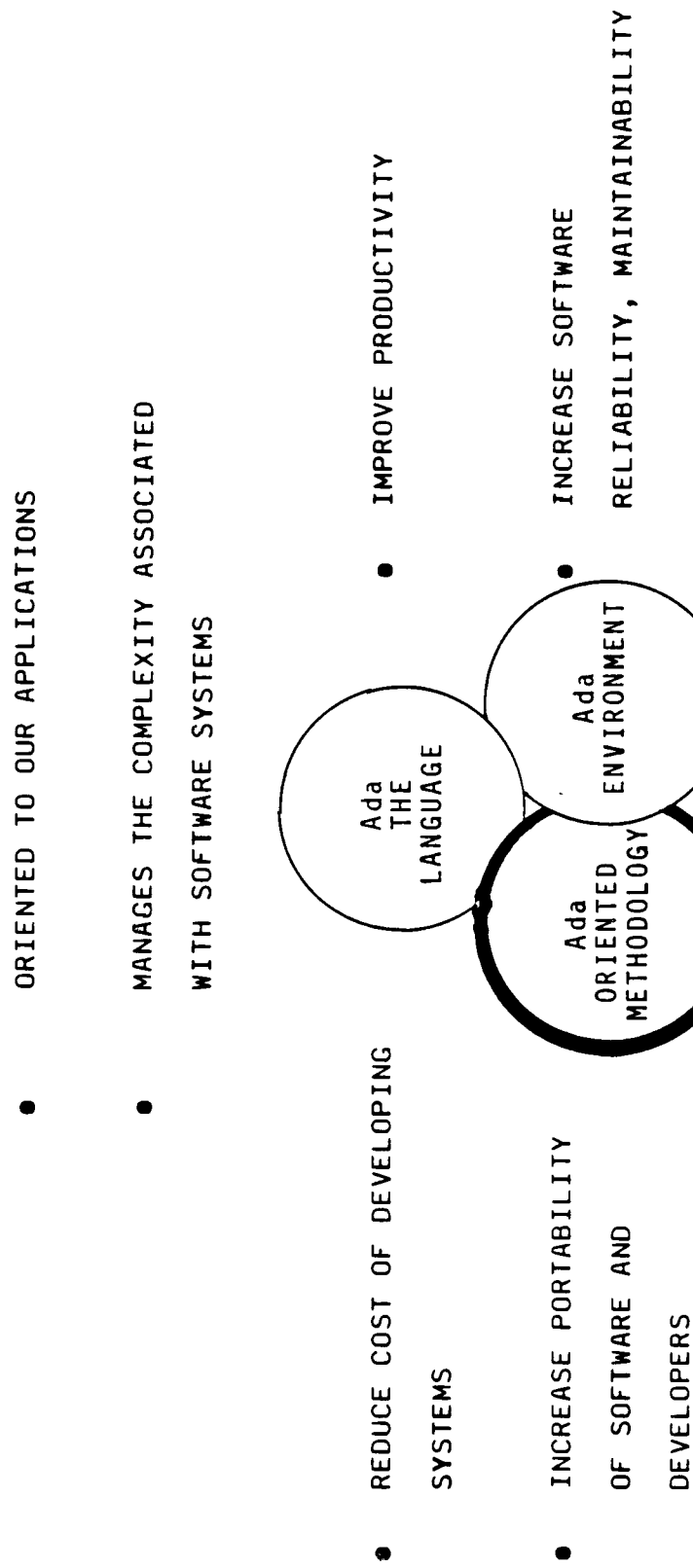
- NO SINGLE EXISTING METHODOLOGY MEETS THE "IDEAL" REQUIREMENTS
 - USED IN COMBINATION WITH OTHER METHODOLOGIES AND SOME ENHANCEMENTS
- WE CAN COME CLOSE

- IN THE COURSE YOU WILL
 - GET AN OVERVIEW OF A LOT OF METHODS
 - GET SOME INSIGHT INTO HOW THEY FIT THIS "IDEAL" SET OF REQUIREMENTS

INSTRUCTOR NOTES

THIS SLIDE SHOWS HOW THE DoD VIEW THE TOTAL Ada EFFORT.

RELATIONSHIP OF "Ada THE LANGUAGE" AND METHODOLOGIES



THREE ASPECTS WHICH ADDRESS THE "SOFTWARE PROBLEM"

INSTRUCTOR NOTES

THESE ARE THE STATED GOALS OF THE METHODMAN DOCUMENT, AND TO SOME EXTENT THE STARS PROGRAM (AS OF EARLY 1985).

Ada ORIENTED METHODOLOGY

- GOAL IS TO ENCOURAGE THE USE OF AND SUPPLEMENT THE DEVELOPMENT OF THE SET OF METHODS USED IN THE PROCESS OF MANAGING, DEVELOPING, AND MAINTAINING SOFTWARE SYSTEMS
- MUST ADDRESS FULL LIFE CYCLE
- KEY ASSUMPTION - "INCREASED EFFORT IN EARLIER STAGES OF DEVELOPMENT WILL BE REFLECTED IN REDUCED COST FOR TESTING AND MAINTENANCE"
- FOCUS COMES FROM
 - METHODMAN
 - STARS (SOFTWARE TECHNOLOGY FOR ADAPTABLE AND RELIABLE SYSTEMS)

INSTRUCTOR NOTES

DETAILS OF METHODMAN GOALS.

"METHODMAN" GOALS

GENERAL GOALS:

- BALANCE BETWEEN SIMPLICITY AND COMPLEXITY
- CONTROL OF COMPLEXITY
- RIGOR
- APPLY TO ANY PROBLEM DOMAIN

TECHNICAL GOALS:

- CRITERIA GIVEN FOR ALL TECHNICAL ASPECTS
- FORMALIZATION OF SPECIFICATIONS AND DESIGN
- VERIFICATION OF SPECIFICATION AND DESIGN DECISIONS
- PROVIDE AN EXPLICIT MODEL OF THE REAL WORLD
- OVERALL OPTIMIZATION OF LOGICAL/PHYSICAL DATA AND PROCESSING ARCHITECTURES
- SUPPORT DESIGN OF CONCURRENT HARDWARE AND SOFTWARE SYSTEMS

AUTOMATION GOALS:

- AUTOMATE LIFE CYCLE PROCESSES THAT ARE CONVENTIONALLY DONE MANUALLY AND REDUNDANTLY
- INTEGRATED FAMILY OF TOOLS
- PROVIDE GRAPHICAL TOOLS

PRODUCT MANAGEMENT GOALS:

- PROVIDE QUALITY CONTROL
- PROVIDE VERSION CONTROL
- PROVIDE CONFIGURATION MANAGEMENT
- PROVIDE AN EXPLICIT MODEL OF THE SOFTWARE DEVELOPMENT PROCESS

PROJECT MANAGEMENT GOALS:

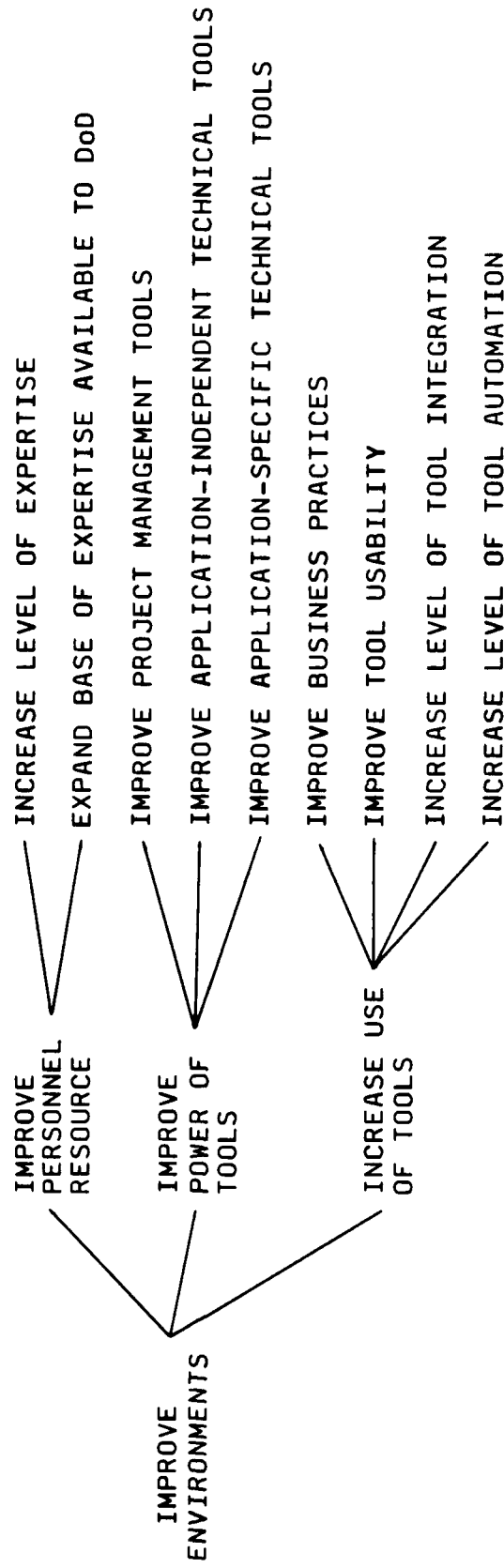
- IMPROVE THE MANAGEABILITY OF SOFTWARE PRODUCTION
- IMPROVE THE EFFICIENCY OF SOFTWARE PRODUCTION
- IMPROVE THE SE PRACTICES OF PROGRAMMERS IN THE ORGANIZATION
- 100% CENTRALLY VERIFIED CONSISTENCY
- IMPROVE PRODUCTIVITY OVER THE ENTIRE LIFE CYCLE

"A GOOD METHODOLOGY"

THE STARS PROGRAM

WILL FOCUS Ada METHODOLOGY DEVELOPMENT

OBJECTIVES



"... TO IMPROVE PRODUCTIVITY WHILE ACHIEVING GREATER SYSTEM RELIABILITY AND ADAPTABILITY"

INSTRUCTOR NOTES

THEME: ANALYSIS IS CRITICAL IN THE DEVELOPMENT OF SOFTWARE, THUS WE MUST HAVE GOOD METHODS TO ADDRESS AND IDENTIFY THE REAL SOFTWARE REQUIREMENTS

PURPOSE: FOCUS THE STUDENTS' ATTENTION ON THE CRITICAL ASPECTS OF REQUIREMENTS ANALYSIS

- REFERENCES:
1. FREEMAN, P., " A PERSPECTIVE ON REQUIREMENTS ANALYSIS AND SPECIFICATION" IBM DESIGN '79 SYMPOSIUM PROCEEDINGS; APRIL 1979
 2. WEINBERG, G., "RETHINKING SYSTEMS ANALYSIS AND DESIGN" LITTLE, BROWN, MA; 1982
 3. DoD-STD-SDS, PROPOSED MILITARY STANDARD FOR DEFENSE SYSTEM SOFTWARE DEVELOPMENT, DEC. 1983 (DoD-STD-1267)

Section 5

ANALYSIS

INSTRUCTOR NOTES

A REQUIREMENT IS A BINDING CONDITION WHICH STATES A MANDATORY CHARACTERISTIC OF AN ABSTRACT OR PHYSICAL OBJECT.

REQUIREMENTS MAY HAVE DIFFERENT FORMS: A SPECIFIC DESCRIPTION, A CONSTRAINT, AN EVALUATION CRITERIA FOR JUDGING QUALITY, OR IT MAY BE IMPLIED BY CONTEXT.

THERE'S ALWAYS MORE THAN ONE PARTY INVOLVED. ACHIEVING CONSENSUS IS AN IMPORTANT ASPECT OF REQUIREMENTS ANALYSIS. STRESS THAT REQUIREMENTS ANALYSIS IS A HUMAN ENDEAVOR.

AD-A165 300

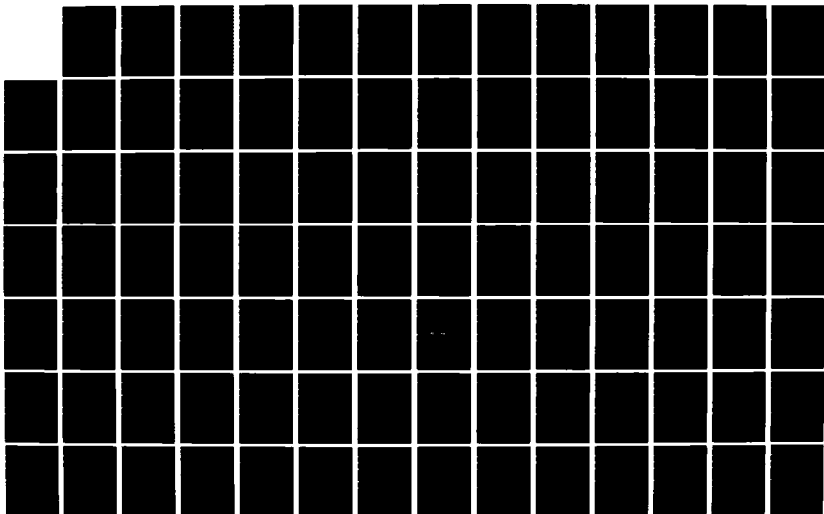
ADA (TRADEMARK) TRAINING CURRICULUM SOFTWARE
ENGINEERING METHODOLOGIES M201 TEACHER'S GUIDE VOLUME 1
(U) SOFTECH INC WALTHAM MA 1986 DAAB07-83-C-K506

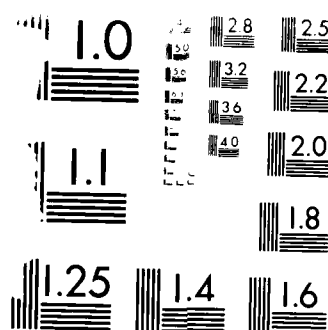
2/5

UNCLASSIFIED

F/G 5/9

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

ANALYSIS IMPLIES BOTH REQUIREMENTS ANALYSIS AND SPECIFICATION

- A REQUIREMENT IS ...
 - AN EXPRESSION OF NEED
 - AN IMPOSED DEMAND
 - SOMETHING SOMEONE WILL PAY FOR
- REQUIREMENTS FORM A "CONTRACT BETWEEN USER AND DEVELOPERS"
- REQUIREMENTS ARE DOCUMENTED IN REQUIREMENTS SPECIFICATION(S)

INSTRUCTOR NOTES

THE ANALYSIS IS USUALLY CONDUCTED WHILE AT THE SPECIFICATION AND/OR DESIGN LEVEL

REQUIREMENTS ANALYSIS

- REQUIREMENTS ANALYSIS IS THE PROCESS BY WHICH THE FEASIBILITY OF REQUIREMENTS ARE DETERMINED, PRIOR TO THE DEVELOPMENT OF THE SYSTEM
- THE ANALYSIS IS TYPICALLY PERFORMED BY A SENIOR SYSTEMS EXPERT OR ANALYST WHO ESTABLISHES AND DOCUMENTS THE REQUIREMENTS

INSTRUCTOR NOTES

GOOD ANALYSTS CAN EXPRESS THE REAL REQUIREMENTS, AS OPPOSED TO THE STATED OR PERCEIVED ONES. IT IS A DIFFICULT JOB, AND MUST BE CAREFULLY THOUGHT OUT.

THE ANALYST IS THE BRIDGE BETWEEN USERS AND DEVELOPERS

- AN ANALYST WILL:

- POSTPONE DESIGN DECISIONS
- PUT THEMSELVES IN THE USER'S CHAIR
- QUESTION THE RATIONALE
- UNDERSTAND THE REAL PROBLEM
- CONSIDER SEVERAL SOLUTIONS
- CLEARLY COMMUNICATE REQUIREMENTS AS WELL AS THE RESULTING IMPLEMENTATION

INSTRUCTOR NOTES

MANY SYSTEMS HAVE BEEN BUILT, TESTED AND MADE OPERATIONAL, ONLY TO BE NOT USED BECAUSE THEY DIDN'T MEET THE "REAL" REQUIREMENTS, ALTHOUGH THEY MET THE STATED REQUIREMENTS. REQUIREMENTS ARE HARD TO FORMULATE CORRECTLY!

CONSEQUENCES OF WRONG REQUIREMENTS

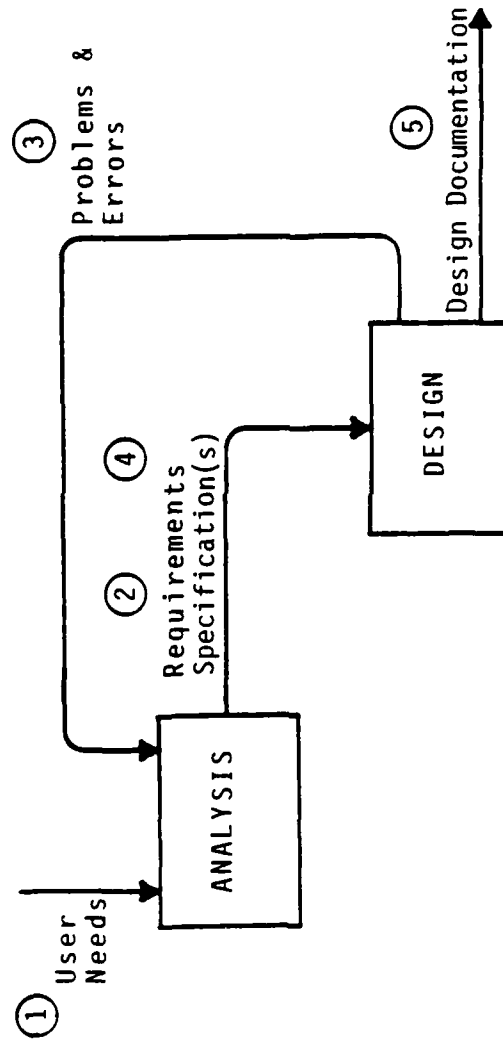
- WRONG REQUIREMENTS CAN CAUSE ...
 - SYSTEM REJECTION
 - SYSTEM PATCHING OR RETROFIT
 - INSTALLATION OF A DANGEROUS SYSTEM
 - FAILURE OF THE PROJECT
 - LOSS OF FUTURE BUSINESS

INSTRUCTOR NOTES

POINT OUT THAT OFTEN SOME LEVEL OF DESIGN IS NEEDED DURING ANALYSIS IN ORDER TO DETERMINE REQUIREMENTS FEASIBILITY.

ANALYSIS AND DESIGN ARE INTERRELATED

- ANALYSIS AND DESIGN ITERATE ...



INSTRUCTOR NOTES

POINT OUT THAT DESIGNERS ALSO CONTRIBUTE TO THE ANALYSIS PHASE. THEY CITE PROBLEMS AND ERRORS FROM THEIR VIEWPOINT OF FEASIBILITY, AND CONVEY THIS TO THE ANALYST. ALSO STATE THAT THERE MAY BE MULTIPLE REQUIREMENT SPECIFICATIONS (SOFTWARE, HARDWARE, SYSTEM, ETC.) THAT NEED TO BE CREATED AND ITERATED. CONFLICTS AMONG THEM NEED TO BE SPOTTED BY THE ANALYST.

DESIGNER'S ROLE IN ANALYSIS

- DURING ANALYSIS, DESIGNERS IDENTIFY ...

- AMBIGUITIES

- INCONSISTENCIES

- INCOMPLETENESS

- POTENTIAL TROUBLE AREAS

- REQUIREMENTS HAVING DISPROPORTIONATE
COST/SCHEDULE IMPACT

INSTRUCTOR NOTES

TO BE EFFECTIVE, THE ANALYST MUST BE ABLE TO UNDERSTAND THE USERS' NEEDS AND WANTS, WHAT IS POSSIBLE TO IMPLEMENT, AND CONVEY THIS TO THE DESIGNERS AND IMPLEMENTERS. REMEMBER, USERS AND IMPLEMENTERS SPEAK TWO DIFFERENT LANGUAGES. THE ANALYST MUST BE FLUENT IN BOTH.

GENERAL GUIDELINES FOR ANALYSIS

- DON'T LIMIT YOUR PERSPECTIVE TO UNCOVERING ONLY SOFTWARE FUNCTIONALITY...
 - DISCOVER THE "WHYS"
 - DISCOVER THE "MOTIVATION FOR" A REQUIREMENT
- CONSIDER ALL ASPECTS OF A "SYSTEM"
 - MISSION NEEDS
 - OPERATIONAL OBJECTIVES AND CONCEPTS
 - SYSTEM PERSONNEL (OPERATORS, CONTROL OFFICERS, ETC.)
 - SOFTWARE FUNCTIONS
 - INFORMATION
 - TIMING CONSTRAINTS
 - HARDWARE INTERFACES AND FUNCTIONS
- ALWAYS MAKE YOUR GOAL A DELIVERABLE (A DOCUMENT OR SET OF DOCUMENTS)
 - WRITTEN SO THE "CUSTOMER" CAN UNDERSTAND IT, REVIEW IT, AND APPROVE IT
 - USE AN APPROPRIATE LANGUAGE (DIAGRAMS, TEXT, PICTURES, ETC.)

INSTRUCTOR NOTES

OFTEN WHEN YOU ASK FOR REQUIREMENTS YOU GET SOLUTIONS. DON'T TAKE A SOLUTION METHOD FOR A PROBLEM DEFINITION (AND VICE VERSA). ALSO, THERE CAN BE MANY "HOWS" FOR THE SAME "WHAT." SOMETIMES DEALING ONLY WITH EXTERNAL BEHAVIOR WILL HELP SEPARATE THE "HOWS" FROM THE "WHATS."

GENERAL GUIDELINES FOR ANALYSIS - continued

- ALWAYS ASK WELL FOCUSED QUESTIONS
 - FOCUS ON THE USER'S PROBLEM FROM HIS VIEWPOINT
 - SAMPLE QUESTIONS
 - "WHAT ARE YOU GOING TO DO WITH _____?"
 - "HOW OFTEN MUST _____ BE DONE?"
 - "WHAT DATA IS REQUIRED TO MAKE DECISION _____?"
- EXPECT ERRORS, BECAUSE PEOPLE FORGET, MISCOMMUNICATE AND CHANGE THEIR MINDS
- FOCUS ON WHAT IS TO BE DONE, NOT HOW IT IS TO BE DONE

INSTRUCTOR NOTES

THESE NEXT FEW SLIDES PROVIDES CURRENT THINKING FROM DoD'S POINT OF VIEW ON THE ROLE OF ANALYSIS.

DoD-STD-SDS VIEW OF ANALYSIS

- THE DoD STANDARD FOR DEFENSE SYSTEM SOFTWARE DEVELOPMENT (MIL-STD-SDS) ESTABLISHES
 - UNIFORM SOFTWARE DEVELOPMENT PROCESS
 - PRACTICES DEMONSTRATED TO BE COST-EFFECTIVE FROM A SOFTWARE LIFE CYCLE PERSPECTIVE
 - DOCUMENTATION REQUIREMENTS
 - REVIEW REQUIREMENTS
- SDS ANALYSIS GOAL
 - ESTABLISH COMPLETE SET OF FUNCTIONAL PERFORMANCE AND INTERFACE REQUIREMENTS FOR EACH SOFTWARE ITEM
- SDS ANALYSIS ACTIVITIES
 - ANALYZE EXISTING NEEDS OR REQUIREMENTS DOCUMENTS FOR ADEQUACY, TESTABILITY, UNDERSTANDABILITY, AND COMPLETENESS
 - USE A STRUCTURED REQUIREMENTS ANALYSIS TOOL OR TECHNIQUES
 - DEVELOP SPECIFICATION SET

INSTRUCTOR NOTES

THE SOFTWARE SPECIFICATION REVIEW CONSISTS OF REVIEWING BOTH THE SOFTWARE REQUIREMENTS
AND INTERFACE REQUIREMENTS SPECIFICATION DOCUMENTS.

MIL-STD-SDS VIEW OF ANALYSIS

- SDS ANALYSIS PRODUCTS
 - RECORDS OF DOCUMENT REVIEWS, STUDIES, ACTION ITEMS AND OTHER ANALYSIS INTERMEDIATE INFORMATION
 - SOFTWARE REQUIREMENTS SPECIFICATION
 - INTERFACE REQUIREMENTS SPECIFICATION
- SDS ANALYSIS REVIEWS
 - SOFTWARE SPECIFICATION REVIEW

INSTRUCTOR NOTES

THE NEXT COUPLE OF SLIDES DETAILS WHAT A SOFTWARE REQUIREMENTS SPECIFICATION LOOKS LIKE AND ITS CONTENTS. POINT OUT THAT THE SPECIFICATION FULLY DESCRIBES THE SOFTWARE ITEM FROM MANY VIEWPOINTS AND LEVELS OF ABSTRACTION.

SOFTWARE REQUIREMENTS SPECIFICATION

(TOPICAL OUTLINE)

1. SCOPE - PROVIDES AN IDENTIFICATION AND SUMMARIZES THE PURPOSE OF THE SOFTWARE ITEM.
2. APPLICABLE DOCUMENTS - IDENTIFIES OTHER DOCUMENTS WHICH FORM PART OF THIS SPECIFICATION.
3. REQUIREMENTS - SPECIFIES IN FOLLOWING SUBSECTIONS BELOW ALL PHYSICAL, INTERFACE, FUNCTIONAL, PERFORMANCE, AND QUALITY REQUIREMENTS FOR THE SOFTWARE ITEM.
 - 3.1 PROGRAMMING REQUIREMENTS - IDENTIFIES PROGRAMMING LANGUAGE AND COMPILER/ASSEMBLER REQUIREMENTS.
 - 3.2 SIZING AND TIMING REQUIREMENTS - SPECIFIES MEMORY AND PROCESSING TIME ALLOCATED TO THIS SOFTWARE ITEM.
 - 3.3 INTERFACE REQUIREMENTS - TYPICALLY REFER TO THE INTERFACE REQUIREMENTS SPECIFICATION.

INSTRUCTOR NOTES

VG 778.1

5-12i

SOFTWARE REQUIREMENTS SPECIFICATION (TOPICAL OUTLINE) - continued

- 3.4 DETAILED FUNCTIONAL AND PERFORMANCE REQUIREMENTS
 - 3.4.X FUNCTION X - IDENTIFIES AN INDIVIDUAL FUNCTION INDICATING THE STATES IT IS USED IN, INPUTS, OUTPUTS, AND PROCESSING
 - 3.4.X.1 INPUTS - DESCRIBES THE INPUT INFORMATION REQUIRED BY FUNCTION X. IDENTIFIES SOURCES, FORMATS, UNITS OF MEASURE, LIMITS, ETC.
 - 3.4.X.2 PROCESSING - SPECIFIES THE OPERATIONS THAT MAKE UP FUNCTION X. IDENTIFIES LOGICAL ALGORITHM, SEQUENCING, TIMING OF EVENTS, PARAMETERS, RESTRICTIONS AND ALLOCATION OF PROCESSING REQUIREMENTS.
 - 3.4.X.3 OUTPUTS - DESCRIBES THE OUTPUT INFORMATION GENERATED BY FUNCTION X. IDENTIFIES RANGE OF VALUES, ACCURACY, PRECISION AND FREQUENCY OF OUTPUT INFORMATION.
 - 3.5 ADAPTATION REQUIREMENTS - IDENTIFIES DATA THAT CAN BE CENTRALLY MODIFIED TO DEFINE THE SCOPE OF THE OPERATIONAL FUNCTIONS WITHIN PRESCRIBED LIMITS. IDENTIFIES ENVIRONMENTAL DATA, SYSTEM PARAMETERS AND SYSTEM CAPACITIES.
 - 3.6 DATABASE REQUIREMENTS - DESCRIBES THE STRUCTURE OF ANY DATABASE WHICH IS PART OF THE SOFTWARE ITEM.

VG 778.1

5-13i

1. *Staphylococcus aureus* (100%)

SOFTWARE REQUIREMENTS SPECIFICATION

(TOPICAL OUTLINE) - continued

- 3.7 QUALITY FACTORS - SPECIFIES CORRECTNESS, RELIABILITY, EFFICIENCY, INTEGRITY, USABILITY, MAINTAINABILITY, TESTABILITY, FLEXIBILITY, PORTABILITY, REUSABILITY AND INTEROPERABILITY REQUIREMENTS.
- 3.8 TRACEABILITY REQUIREMENTS - PROVIDES A MAP FROM HIGHER LEVEL SPECIFICATIONS TO THESE SPECIFICATIONS ON A REQUIREMENT BY REQUIREMENT BASIS.
- 4. QUALIFICATION REQUIREMENTS - SPECIFIES THE METHODS AND LEVELS OF TESTING NECESSARY TO SATISFY THE REQUIREMENTS IN SECTION 3.
- 5. PREPARATION FOR DELIVERY - FORMAT OF TAPE, DISK, FLOPPY, ETC.
- 6. NOTES.

INSTRUCTOR NOTES

THIS SLIDE PROVIDES THE FORM AND CONTENT OF THE INTERFACE REQUIREMENTS SPECIFICATION DOCUMENT. QUALIFICATION REQUIREMENTS AND PREPARATION FOR DELIVERY ARE THE SAME AS FOR THE SOFTWARE REQUIREMENTS SPECIFICATION.

INTERFACE REQUIREMENTS SPECIFICATION (TOPICAL OUTLINE)

1. SCOPE - PROVIDES AN IDENTIFICATION AND SUMMARIZES THE PURPOSE OF THE INTERFACES TO/FROM THIS SOFTWARE ITEM.
2. APPLICABLE DOCUMENTS.
3. REQUIREMENTS.
 - 3.1 INTERFACE BLOCK DIAGRAM - IDENTIFIES THE RELATIONSHIP OF THIS SOFTWARE ITEM AND OTHER SOFTWARE AND/OR HARDWARE ITEMS.
 - 3.2 INTERFACE IDENTIFICATION/DOCUMENTATION TABLE - IDENTIFIES INTERFACES AND WHERE THEY ARE DOCUMENTED.
 - 3.3 DETAILED INTERFACE REQUIREMENTS - SPECIFIES FOR EACH INTERFACE DATA SOURCE/DESTINATION, TIMING, TRANSFER PROTOCOL, FORMAT, ETC.
4. QUALIFICATION REQUIREMENTS.
5. PREPARATION FOR DELIVERY.
6. NOTES.

INSTRUCTOR NOTES

THE NEXT TWO SLIDES ARE LEAD-INS TO THE ACTUAL TECHNIQUES TO BE STUDIED.

THE PERSPECTIVES PROVIDE THE DIFFERENT VIEWPOINTS FROM WHICH A SYSTEM CAN BE ANALYZED.

THE FORMATS PROVIDE THE DIFFERENT WAYS THE VIEWPOINTS CAN BE DISPLAYED BY THE ANALYST.

WE USE PERSPECTIVES AND FORMATS TO HELP CATEGORIZE THE METHODOLOGIES.

METHODOLOGY COURSE

ANALYSIS PERSPECTIVES AND FORMATS

- PRIMARY PERSPECTIVES CONSIDERED DURING ANALYSIS:
 - FUNCTIONAL: VIEWS FLOW AND TRANSFORMATION OF DATA
(INFORMATION)
 - BEHAVIORAL: VIEWS SEQUENCING AND CONCURRENCY OF
THE TRANSFORMATION OF DATA (INFORMATION)
 - INFORMATIONAL: VIEWS RELATIONSHIPS BETWEEN DATA
(INFORMATION)
- PRIMARY FORMATS:
 - GRAPHICAL MODEL
 - REQUIREMENTS LANGUAGE
 - DATA DICTIONARY
 - DOCUMENT STRUCTURE

INSTRUCTOR NOTES

DISCUSS THE ADVANTAGES AND DISADVANTAGES OF PROVIDING MULTIPLE PERSPECTIVES AND FORMATS. FOR EXAMPLE, USE THE FOLLOWING ANALOGIES:

CIRCUIT DIAGRAM LANGUAGE OF ELECTRICAL ENGINEERS - CONCISE, PRECISE

TEXT LANGUAGE WHICH MANY UNDERSTAND - NOT FORMAL, NO WAY OF TESTING COMPLETENESS

PICTURE PICTURE IS SIMPLER - SHOWS DIMENSIONS, RELATES TO A COMMON LEVEL OF UNDERSTANDING

NOTE: ONE IS NOT "BETTER" THAN THE OTHER. IT REALLY DEPENDS ON THE AUDIENCE AND ITS BACKGROUND.

ANALYSIS METHODOLOGIES TO BE PRESENTED

<u>METHODOLOGY</u>	<u>PERSPECTIVES</u>		<u>FORMAT</u>
• STRUCTURED ANALYSIS AND DESIGN TECHNIQUE (SADT)	•	FUNCTIONAL	• GRAPHICAL MODEL
• SOFTWARE REQUIREMENTS ENGINEERING METHODOLOGY (SREM)	• • •	FUNCTIONAL BEHAVIORAL INFORMATIONAL	• GRAPHICAL MODEL • REQUIREMENTS LANGUAGE • DOCUMENT STRUCTURE
• ENTITY DIAGRAMMING	•	INFORMATIONAL	• GRAPHICAL MODEL
• PSL/PSA	• • •	FUNCTIONAL INFORMATIONAL BEHAVIORAL	• REQUIREMENTS LANGUAGE • DOCUMENT STRUCTURE
• DEMARCO DATA FLOW	•	FUNCTIONAL	• GRAPHICAL MODEL • DATA DICTIONARY
• GANE AND SARSON DIAGRAMS	•	FUNCTIONAL	• GRAPHICAL MODEL • DATA DICTIONARY
• SOFTWARE COST REDUCTION PROJECT	• • •	FUNCTIONAL BEHAVIORAL INFORMATIONAL	• REQUIREMENTS LANGUAGE • DOCUMENT STRUCTURE

1
 2
 3
 4
 5
 6
 7
 8
 9
 10
 11
 12
 13
 14
 15
 16
 17
 18
 19
 20
 21
 22
 23
 24
 25
 26
 27
 28
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 40
 41
 42
 43
 44
 45
 46
 47
 48
 49
 50
 51
 52
 53
 54
 55
 56
 57
 58
 59
 60
 61
 62
 63
 64
 65
 66
 67
 68
 69
 70
 71
 72
 73
 74
 75
 76
 77
 78
 79
 80
 81
 82
 83
 84
 85
 86
 87
 88
 89
 90
 91
 92
 93
 94
 95
 96
 97
 98
 99
 100
 101
 102
 103
 104
 105
 106
 107
 108
 109
 110
 111
 112
 113
 114
 115
 116
 117
 118
 119
 120
 121
 122
 123
 124
 125
 126
 127
 128
 129
 130
 131
 132
 133
 134
 135
 136
 137
 138
 139
 140
 141
 142
 143
 144
 145
 146
 147
 148
 149
 150
 151
 152
 153
 154
 155
 156
 157
 158
 159
 160
 161
 162
 163
 164
 165
 166
 167
 168
 169
 170
 171
 172
 173
 174
 175
 176
 177
 178
 179
 180
 181
 182
 183
 184
 185
 186
 187
 188
 189
 190
 191
 192
 193
 194
 195
 196
 197
 198
 199
 200
 201
 202
 203
 204
 205
 206
 207
 208
 209
 210
 211
 212
 213
 214
 215
 216
 217
 218
 219
 220
 221
 222
 223
 224
 225
 226
 227
 228
 229
 230
 231
 232
 233
 234
 235
 236
 237
 238
 239
 240
 241
 242
 243
 244
 245
 246
 247
 248
 249
 250
 251
 252
 253
 254
 255
 256
 257
 258
 259
 260
 261
 262
 263
 264
 265
 266
 267
 268
 269
 270
 271
 272
 273
 274
 275
 276
 277
 278
 279
 280
 281
 282
 283
 284
 285
 286
 287
 288
 289
 290
 291
 292
 293
 294
 295
 296
 297
 298
 299
 300
 301
 302
 303
 304
 305
 306
 307
 308
 309
 310
 311
 312
 313
 314
 315
 316
 317
 318
 319
 320
 321
 322
 323
 324
 325
 326
 327
 328
 329
 330
 331
 332
 333
 334
 335
 336
 337
 338
 339
 340
 341
 342
 343
 344
 345
 346
 347
 348
 349
 350
 351
 352
 353
 354
 355
 356
 357
 358
 359
 360
 361
 362
 363
 364
 365
 366
 367
 368
 369
 370
 371
 372
 373
 374
 375
 376
 377
 378
 379
 380
 381
 382
 383
 384
 385
 386
 387
 388
 389
 390
 391
 392
 393
 394
 395
 396
 397
 398
 399
 400
 401
 402
 403
 404
 405
 406
 407
 408
 409
 410
 411
 412
 413
 414
 415
 416
 417
 418
 419
 420
 421
 422
 423
 424
 425
 426
 427
 428
 429
 430
 431
 432
 433
 434
 435
 436
 437
 438
 439
 440
 441
 442
 443
 444
 445
 446
 447
 448
 449
 450
 451
 452
 453
 454
 455
 456
 457
 458
 459
 460
 461
 462
 463
 464
 465
 466
 467
 468
 469
 470
 471
 472
 473
 474
 475
 476
 477
 478
 479
 480
 481
 482
 483
 484
 485
 486
 487
 488
 489
 490
 491
 492
 493
 494
 495
 496
 497
 498
 499
 500
 501
 502
 503
 504
 505
 506
 507
 508
 509
 510
 511
 512
 513
 514
 515
 516
 517
 518
 519
 520
 521
 522
 523
 524
 525

5-17i

- VG 778.1

SUMMARY

- REQUIREMENTS CAN BE VIEWED AS NEEDS OR DEMANDS
- WRONG REQUIREMENTS ARE COSTLY
- ANALYSTS ARE THE BRIDGE BETWEEN USERS AND DEVELOPERS
- DESIGNERS HELP ANALYSTS BY VERIFYING FEASIBILITY
- SCOPE OF ANALYSIS GOES BEYOND SIMPLE IDENTIFICATION OF THE ALGORITHMS NEEDED TO BE IMPLEMENTED

INSTRUCTOR NOTES

THEME: SADT IS A GRAPHICAL BASED METHOD THAT MANAGES COMPLEXITY AS ONE ACHIEVES AN UNDERSTANDING OF THE SOFTWARE OR SYSTEM REQUIREMENTS.

PURPOSE: SADT HAS MANY OF THE CHARACTERISTICS OF OTHER GRAPHICAL BASED ANALYSIS METHODS, THEREFORE FAMILIARIZING THE STUDENTS WITH A CLASS OF METHODS.

REFERENCES: 1. ROSS, D., BRACKETT, J. "AN APPROACH TO STRUCTURED ANALYSIS" COMPUTER DECISIONS; SEPTEMBER, 1976.

2. ROSS, D., SHOMAN, K. "STRUCTURED ANALYSIS FOR REQUIREMENTS DEFINITION" IEEE TRANSACTIONS ON SOFTWARE ENGINEERING VOL. 3, NO. 1; JANUARY 1977.

Section 6

SADT METHODOLOGY

INSTRUCTOR NOTES

SADT AIDS IN THE UNDERSTANDING OF THE FUNCTIONS OF COMPLICATED STRUCTURES.
UNDERSTANDING THE FUNCTIONS AIDS IN UNDERSTANDING WHAT THE SYSTEM IS SUPPOSED TO DO.
SADT DOES NOT ADDRESS PERFORMANCE OF A SYSTEM.

STRUCTURED ANALYSIS AND DESIGN TECHNIQUE

- IS AN INTEGRATED APPROACH TO ...
 - ANALYZING SYSTEMS
 - DOCUMENTING REQUIREMENTS
 - COMMUNICATING AMONG ANALYSTS AND USERS
 - REVIEWING, APPROVING, AND CONTROLLING DOCUMENTATION.

INSTRUCTOR NOTES

THE PURPOSE OF SADT IS TO PRODUCE MODELS TO HELP UNDERSTAND THE SYSTEM.

SADT STARTS OUT BY VIEWING THE SYSTEM AS A WHOLE. THEN DECOMPOSES IT INTO VIEWS OF SUBSYSTEMS.

THE LANGUAGE IN A MODEL IS A NATURAL LANGUAGE (E.G., GERMAN) WITH TERMS PERTINENT TO THE PROBLEM DOMAIN.

SADT OVERVIEW

- SADT ANALYSIS PROCEEDS IN A TOP-DOWN FASHION
 - PRESENTS AN OVERVIEW OF THE SYSTEM
 - EXPOSES DETAILS ONE STEP AT A TIME
 - EACH DETAIL IS RELATED TO HIGHER AND LOWER LEVEL DETAILS
- SADT ANALYSIS PRODUCES MODELS
 - ARE A SERIES OF "BLUE-PRINT" DRAWINGS
 - EACH DRAWING:
 - SHOWS A LIMITED AMOUNT OF DETAIL
 - IS AN EASY-TO-GRASP UNIT
 - FITS EXACTLY INTO THE WHOLE MODEL
 - USE LANGUAGE APPROPRIATE FOR DISCUSSING THE PROBLEM

INSTRUCTOR NOTES

"CURRENT UNDERSTANDING" EVOLVES DURING A PROJECT. SADT DOCUMENTS THIS UNDERSTANDING IMMEDIATELY, WHILE IT IS BEING CREATED. THERE IS NO LAG BETWEEN IDEA AND DOCUMENT.

TELL CLASS --

MODELING NOTATION WILL BE COVERED LATER IN THIS TOPIC

INTERVIEWING TECHNIQUES ARE PROPRIETARY

WE WILL NOW SUMMARIZE SADT MODELING PRINCIPLES ...

SADT OVERVIEW

- SADT DOCUMENTATION ...

- REFLECTS THE CURRENT UNDERSTANDING OF ANALYSTS AND USERS
- APPEARS IN A STANDARD FORMAT
- IS ORGANIZED FOR TOP-DOWN READING

- SADT CONSISTS OF

- A GRAPHIC NOTATION
- A SET OF MODELING PRINCIPLES
- A SET OF REVIEW PROCEDURES
- A SET OF INTERVIEWING TECHNIQUES

INSTRUCTOR NOTES

THE LEFT COLUMN REPRESENTS BASIC MODELING PRINCIPLES. THE RIGHT COLUMN INDICATES WHICH SADT FEATURES SUPPORT THAT PRINCIPLE.

- SUBJECT BOUNDARY - SEPARATES ITEM BEING CONSIDERED FROM ITS ENVIRONMENT.
- LIMITING INFORMATION - RULE THAT STATES 3 TO 6 BOXES PER PAGE.
- ETC.

OVERVIEW

<u>MODELING PRINCIPLE</u>		<u>SADT FEATURE</u>
• SUBJECT BOUNDARY	•	BOX
• LIMITING INFORMATION	•	DIAGRAM
• TOP-DOWN DECOMPOSITION	•	BOX DECOMPOSITION
• LEVELS OF DETAIL	•	PARENT AND CHILD DIAGRAMS
• FUNCTIONS AND DATA	•	BOXES AND ARROWS
• LEVELS OF ABSTRACTION	•	MULTIPLE MODELS

GRAPHIC NOTATION (SADT'S LANGUAGE)

- IN SADT BOXES DEPICT FUNCTIONS AND ARROWS DEPICT DATA

- THE SADT LANGUAGE GRAPHICALLY SHOWS MANY KINDS OF
SYSTEM CHARACTERISTICS:

- FEEDBACK
- BOUNDARIES
- INTERFACES
- DATA TRANSFORMS
- DOMINANCE
- REQUIRED SEQUENCING
- CONSTRAINTS

INSTRUCTOR NOTES

WE WILL ONLY COVER ACTIVITY DIAGRAMMING. NO MENTION WILL BE MADE OF DATA DIAGRAMMING.
SO, ALL DISCUSSION OF BOXES, ARROWS, DIAGRAMS, ETC. WILL BE DONE FROM AN ACTIVITY
DIAGRAMMING VIEWPOINT.

INPUT DATA IS TRANSFORMED, WHILE CONTROL DATA AFFECTS THE FUNCTIONALITY OF THE BOX WITHOUT ITSELF BEING TRANSFORMED.

POINT OUT THE PLACEMENT OF ARROWS WITH RESPECT TO DIFFERENT TYPES OF INFORMATION.

SADT LANGUAGE

- IN AN SADT ACTIVITY MODEL ...

INPUT DATA

- ARE CONVERTED INTO OUTPUT

CONTROL DATA

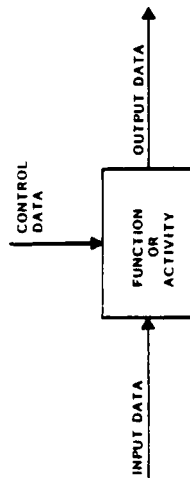
- CONSTRAIN HOW INPUT IS MODIFIED TO PRODUCE OUTPUT
- CAN BE USED TO PRODUCE OUTPUT

OUTPUT DATA

- ARE USED BY OTHER ACTIVITIES

FUNCTION OR ACTIVITY

- PRODUCES OUTPUT DATA BY MODIFYING INPUT DATA AS CONSTRAINED BY CONTROL DATA

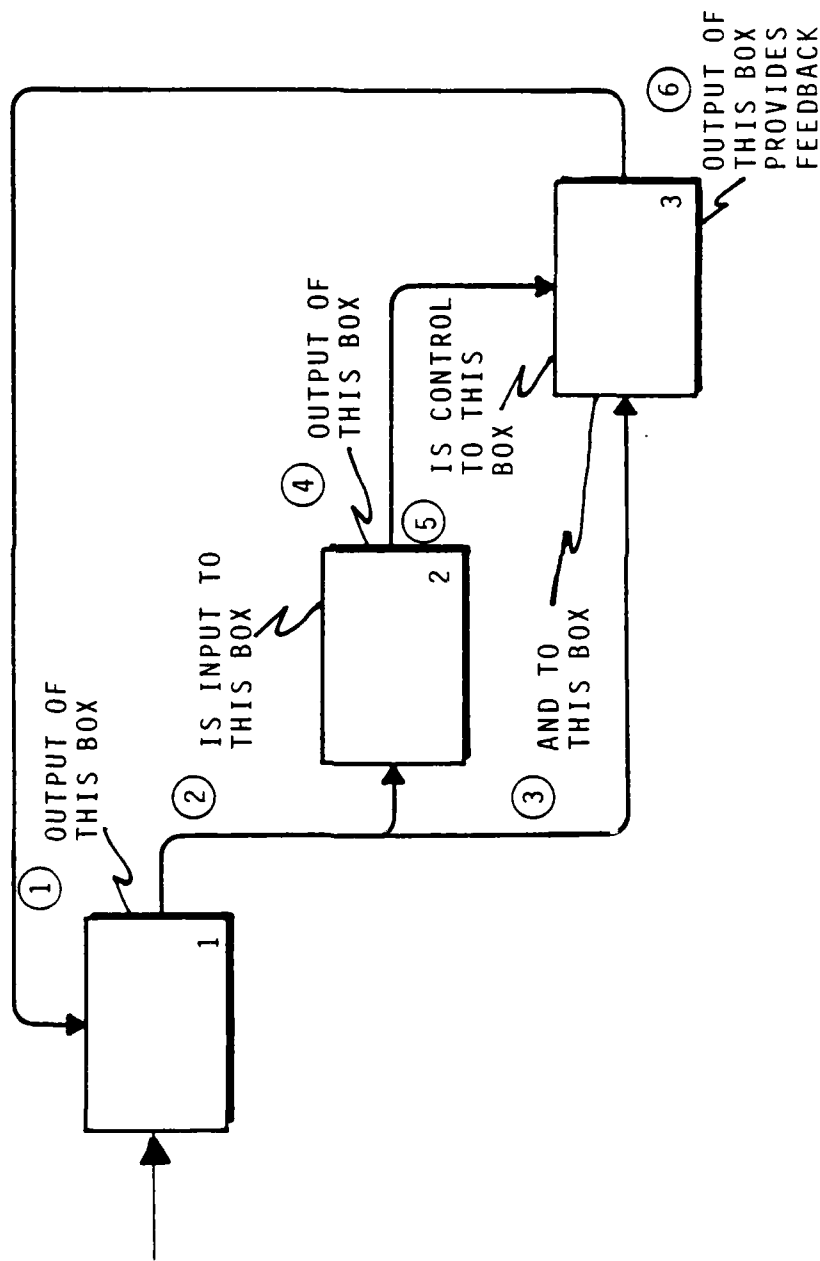


INSTRUCTOR NOTES

DIAGRAMS ARE COMPOSED OF CONNECTED BOXES AND ARROWS.

WALK THROUGH PICTURE, HIGHLIGHTING ARROWS WITH BOLD LINES AND HATCHING BOXES.

SADT DIAGRAM



• IDENTIFIES FUNCTIONS AND DATA FLOW

INSTRUCTOR NOTES

USE THESE SIMPLE RULES:

ACTIVITIES OR FUNCTIONS ARE BOXES AND ARE ALWAYS LABELED WITH VERBS.

ARROWS ARE DATA AND ARE ALWAYS LABELED WITH NOUNS.

THESE CONVENTIONS WILL HELP PROVIDE CLARITY.

SADT DIAGRAM

- DIAGRAM CONSISTS OF SET OF BOXES, ARROWS, LABELS AND ANNOTATION
- BOTH BOXES AND ARROWS HAVE LABELS
 - LABELS (CHOICE OF WORDS) AID COMMUNICATION.
 - LABELS REPRESENT CATEGORIES OF FUNCTIONS AND DATA.
 - FUNCTIONS AND DATA HAVE A RANGE (BREADTH) OF INTERPRETATION.

INSTRUCTOR NOTES

SOME WAYS TO INTERPRET LABELS

INTERPRETATION OF LABELS ON SADT DIAGRAM

FUNCTION LABELS

• A CONTINUOUS PROCESS

• A ONE-TIME OCCURRENCE

• A SERIES OF DISCRETE ACTIONS

• A SET OF SIMILAR ACTIONS
TAKING PLACE ASYNCHRONOUSLY

• A SET OF RELATED BUT DISSIMILAR
ACTIONS

DATA LABELS

• A CONTINUOUSLY CHANGING VARIABLE

• A (SERIES OF) DISCRETE OBJECT(S)

• THE VALUES OF A VARIABLE

• A SET OF SIMILAR OBJECTS OR
VARIABLES CHANGING ASYNCHRONOUSLY

• A SET OF RELATED BUT DISSIMILAR
OBJECTS OR VARIABLES

INSTRUCTOR NOTES

STRESS THAT ACTIVATIONS WILL BECOME CLEAR IN A MOMENT (I.E., IN THE
NEXT SLIDE)

ACTIVATIONS

- EACH DIAGRAM "TELLS A STORY"
- WHENEVER CERTAIN DATA BECOMES AVAILABLE, BOXES BECOME "ACTIVE" AND PERFORM THEIR FUNCTIONS
- AN ACTIVATION IS A WAY IN WHICH A BOX CAN OPERATE USING SOME OF ITS INPUTS AND CONTROLS TO PRODUCE SOME OF ITS OUTPUTS
- FOR ANY SPECIFIC ACTIVATION:
 - NOT ALL INPUTS AND CONTROLS ARE NECESSARILY USED
 - NOT ALL OUTPUTS ARE NECESSARILY PRODUCED

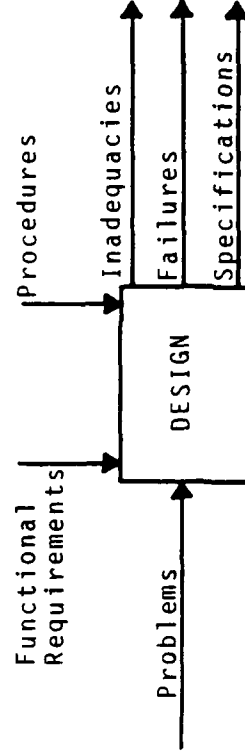
INSTRUCTOR NOTES

WALK THROUGH EACH ACTIVATION, GIVING A STORY OF EACH DESIGN PROCESS.

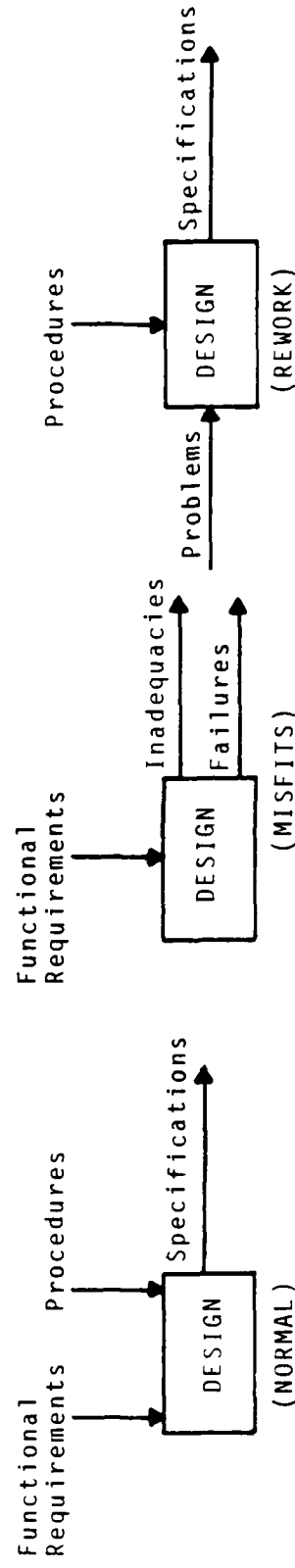
ACTIVATIONS

- THE NUMBER OF POSSIBLE ACTIVATIONS OF A BOX DEPENDS UPON THE INTERACTIONS OF ITS INTERFACE ARROWS, NOT ON THE NUMBER OF ARROWS

THE BOX:



HAS 3 ACTIVATIONS:



DECOMPOSITION

- A BOX IS A BOUNDED CONTEXT.
- THE BOX HELPS ...
 - DEFINE THE SUBJECT
 - FOCUS OUR ATTENTION
 - AVOID INTRODUCING EXTRANEIOUS IDEAS
- DECOMPOSITION IS BREAKING A SUBJECT (BOX) INTO
PIECES (SEVERAL BOXES IN A NEW DIAGRAM)

INSTRUCTOR NOTES

NO INFORMATION IS LOST WHEN A BOX IS DECOMPOSED. ALL INTERFACES MUST BE SATISFIED. DECOMPOSING A BOX INTO 2 PIECES IS TOO LITTLE, 7 IS TOO MUCH. LIMIT DECOMPOSITION TO 3-6 BOXES. (DISCUSS HORARE LIMIT (7 ± 2) ON HUMAN COMPREHENSION LIMITS.) REMEMBER THESE ARE GUIDELINES.

NOTE THAT IN ADDITION TO THE BOXES YOU HAVE TO DEAL WITH:

- CONTEXT OF THIS DIAGRAM
- INTERCONNECTIVITY OF THE BOXES
- PHYSICAL PLACEMENT OF THE BOXES

DISCUSS

EACH OF THESE ISSUES HAVE TO BE KEPT IN MIND WHEN VIEWING THE DECOMPOSITION

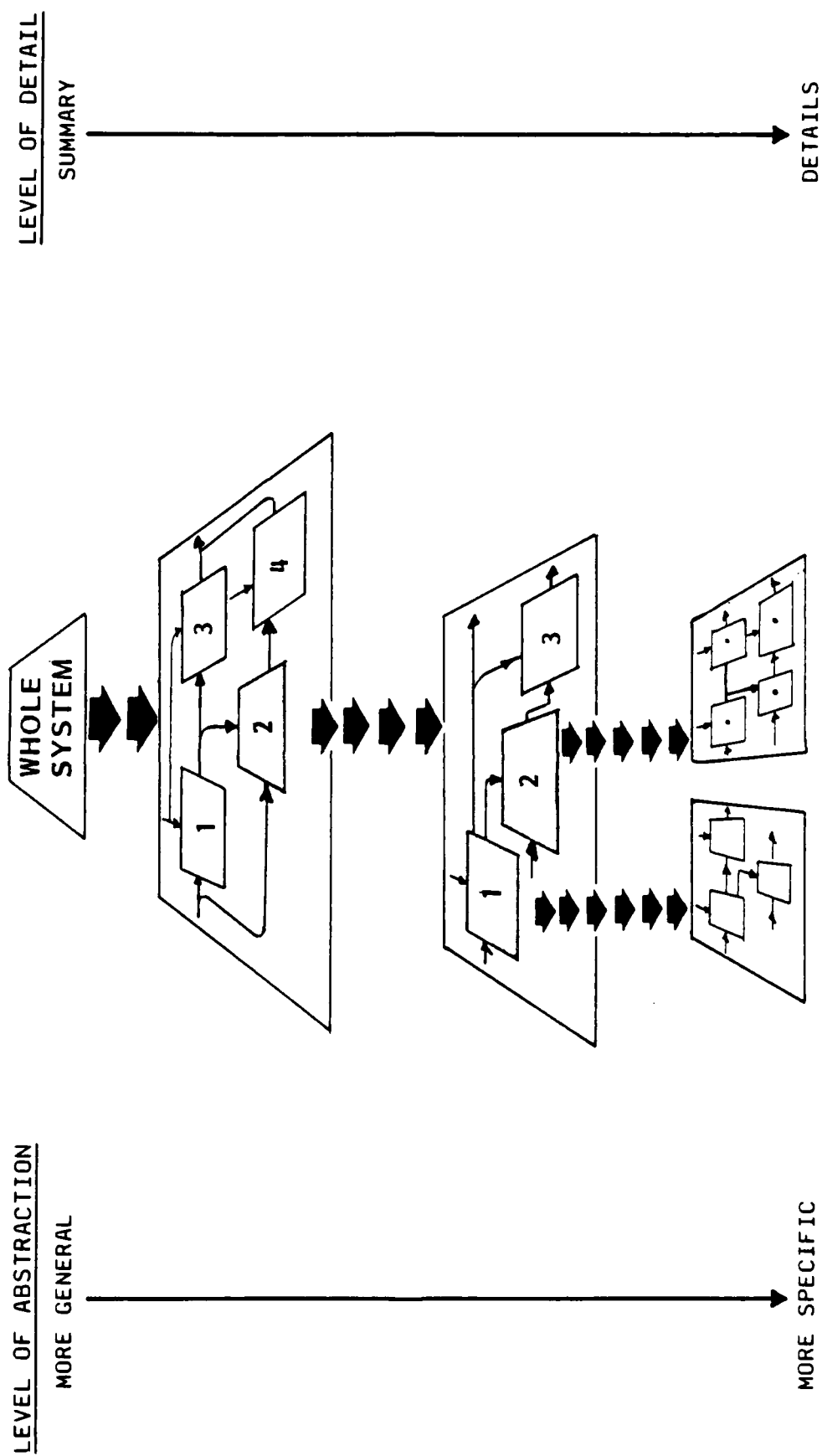
DECOMPOSITION

- WHENEVER A BOX IS DECOMPOSED ...
 - EVERYTHING INSIDE THE BOX IS INCLUDED
 - NOTHING OUTSIDE THE BOX IS INCLUDED
- DUE TO HUMAN LIMITS IN THEIR ABILITY TO UNDERSTANDING MULTIPLE IDEAS AT ONCE, SADT...
 - LIMITS THE NUMBER OF BOXES (3-6) PER DIAGRAM
 - ENSURES EACH DIAGRAM MATCHES EXACTLY WITH ITS PARENT BOX

INSTRUCTOR NOTES

DECOMPOSITION IS A RECURSIVE, AS WELL AS, A HIERARCHIC PROCESS.

DECOMPOSITION

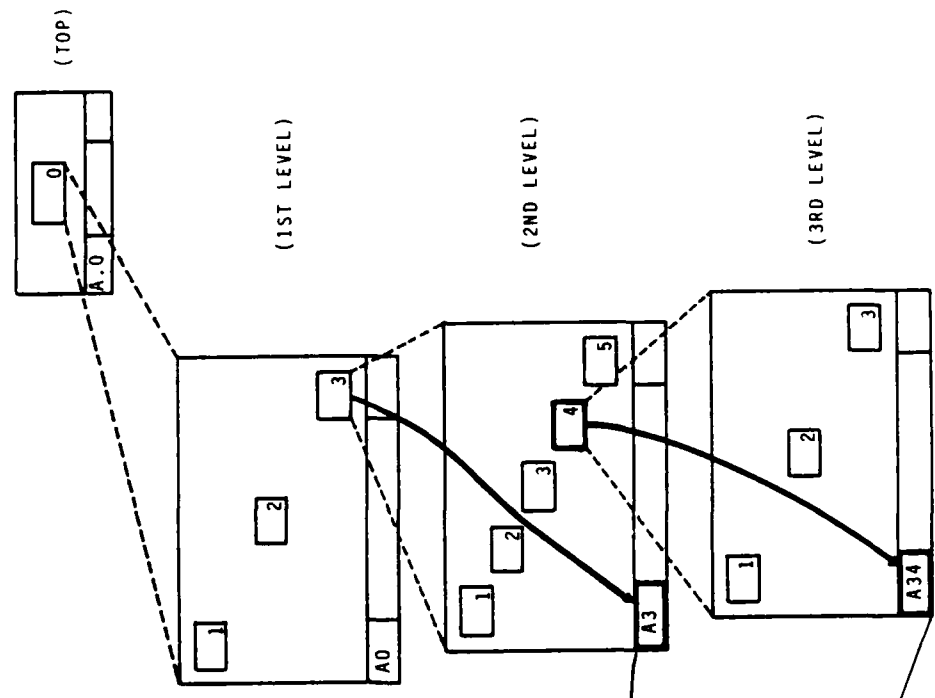


INSTRUCTOR NOTES

WALK THROUGH DIAGRAMS. POINT OUT NODE NUMBERS. SHOW HOW A NODE NUMBER IS CREATED.

DECOMPOSITION

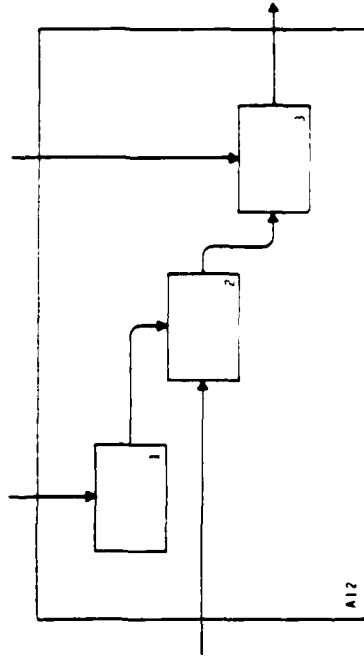
- HOW THE PIECES FIT TOGETHER



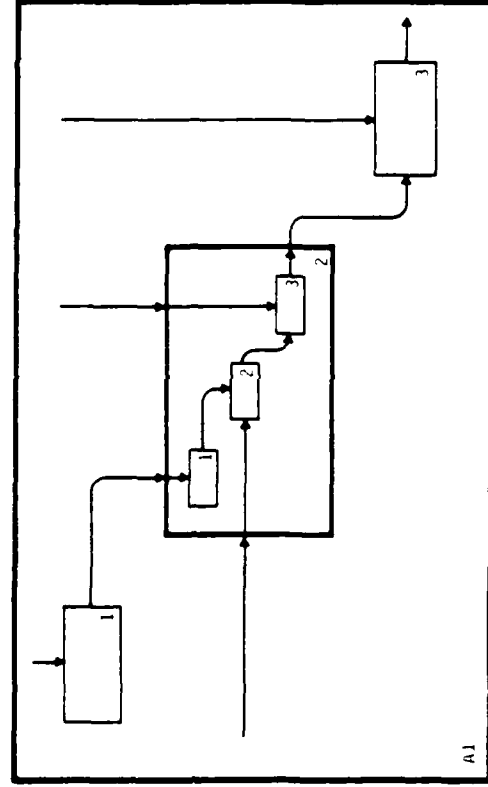
- DIAGRAM NODE NUMBER
PLUS BOX NUMBER
IS NODE NUMBER OF
DETAIL DIAGRAM

DECOMPOSITION

- AFTER DECOMPOSITION, SOME ARROWS REMAIN UNCONNECTED



- SO CONNECTIONS MUST BE MADE TO THE PARENT BOX



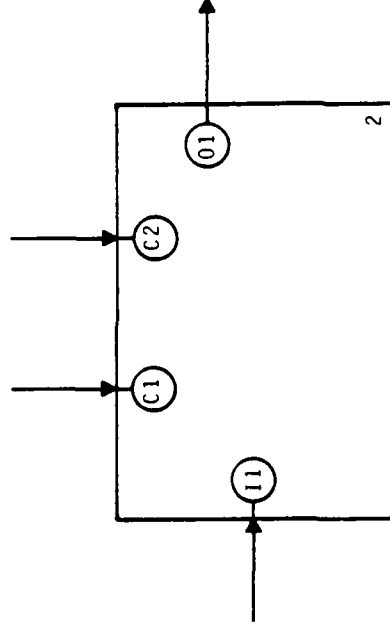
INSTRUCTOR NOTES

WALK AROUND BOX CLOCKWISE, FROM I TO C TO O. TELL HOW NUMBERS ARE GIVEN TOP-BOTTOM AND LEFT-RIGHT. THE LABELING CONVENTIONS FOR INPUTS, CONTROLS, AND OUTPUTS AID IN THE VISUALIZATION PROCESS.

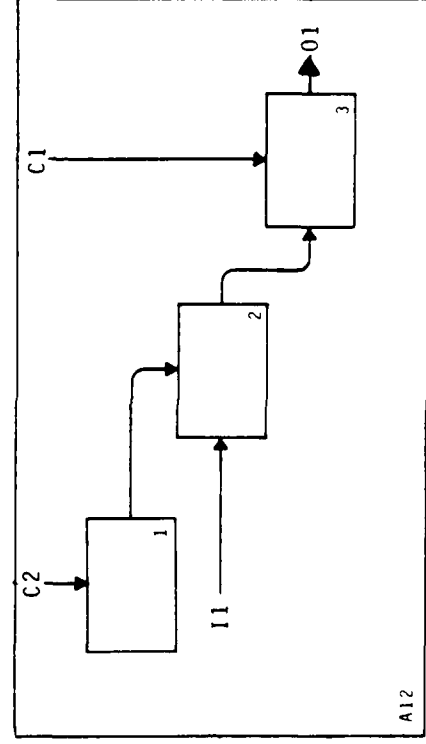
POINT OUT ARROW CORRESPONDENCE BETWEEN THE BOX AND DIAGRAM.

ICOMS

- SADT USES IMPLICIT OFFPAGE CONNECTORS, BASED ON THE POSITION OF THE ARROWS



- THESE CODES ARE USED EXPLICITLY AS LABELS ON BOUNDARY ARROWS



INSTRUCTOR NOTES

LET'S LEAVE THE DECOMPOSITION TOPIC AND TALK ABOUT A FULL MODEL.

A MODEL IN GENERAL LIMITS THE DETAIL OF A SUBJECT TO ENHANCE UNDERSTANDABILITY.

MODEL

- A MODEL IS A REPRESENTATION OF A SUBJECT



- ANSWERS QUESTIONS ABOUT THE SUBJECT

- A MODEL IS A SET OF DIAGRAMS THAT:

- SHOW LIMITED AMOUNTS OF DETAIL
- TALK ABOUT ONE SUBJECT (TOP BOX)

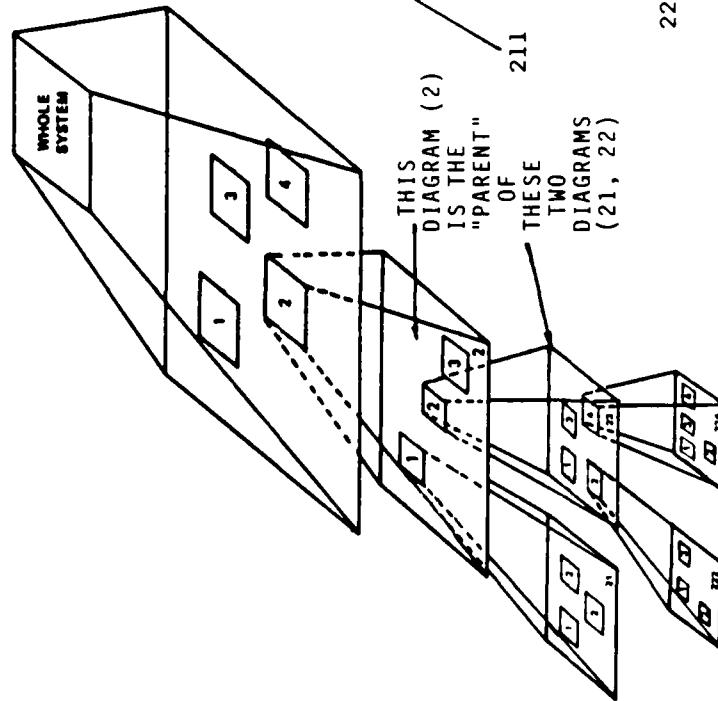
INSTRUCTOR NOTES

IN SADT WE USE SETS OF DIAGRAMS TO REPRESENT OUR MODEL.

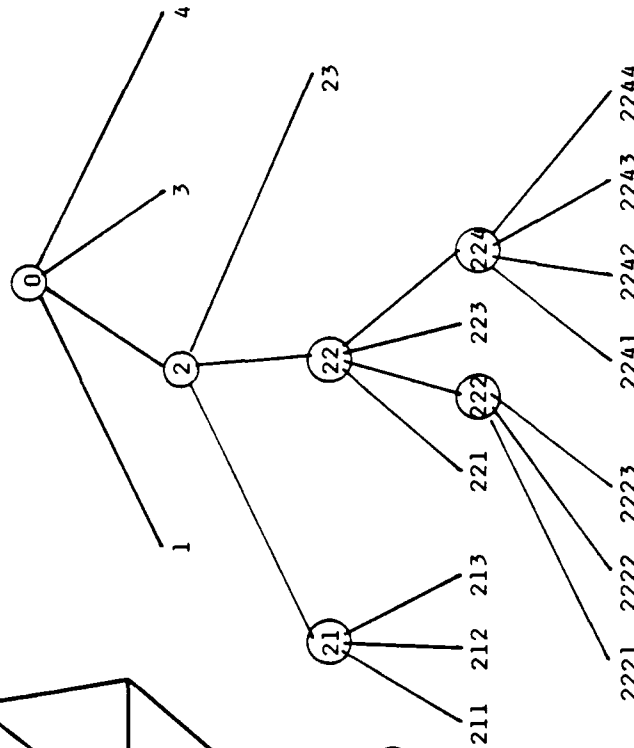
ALTHOUGH WE GO TOP-DOWN (HIERARCHAL) IN OUR ANALYSIS DOES NOT MEAN WE MUST IMPLEMENT OUR SYSTEM THAT WAY.

DIAGRAMS FIT TOGETHER TO FORM A MODEL

- EACH DIAGRAM....
- DETAILS ITS PARENT BOX
- MAY BE DETAILED BY OTHER DIAGRAMS



SADT MODEL

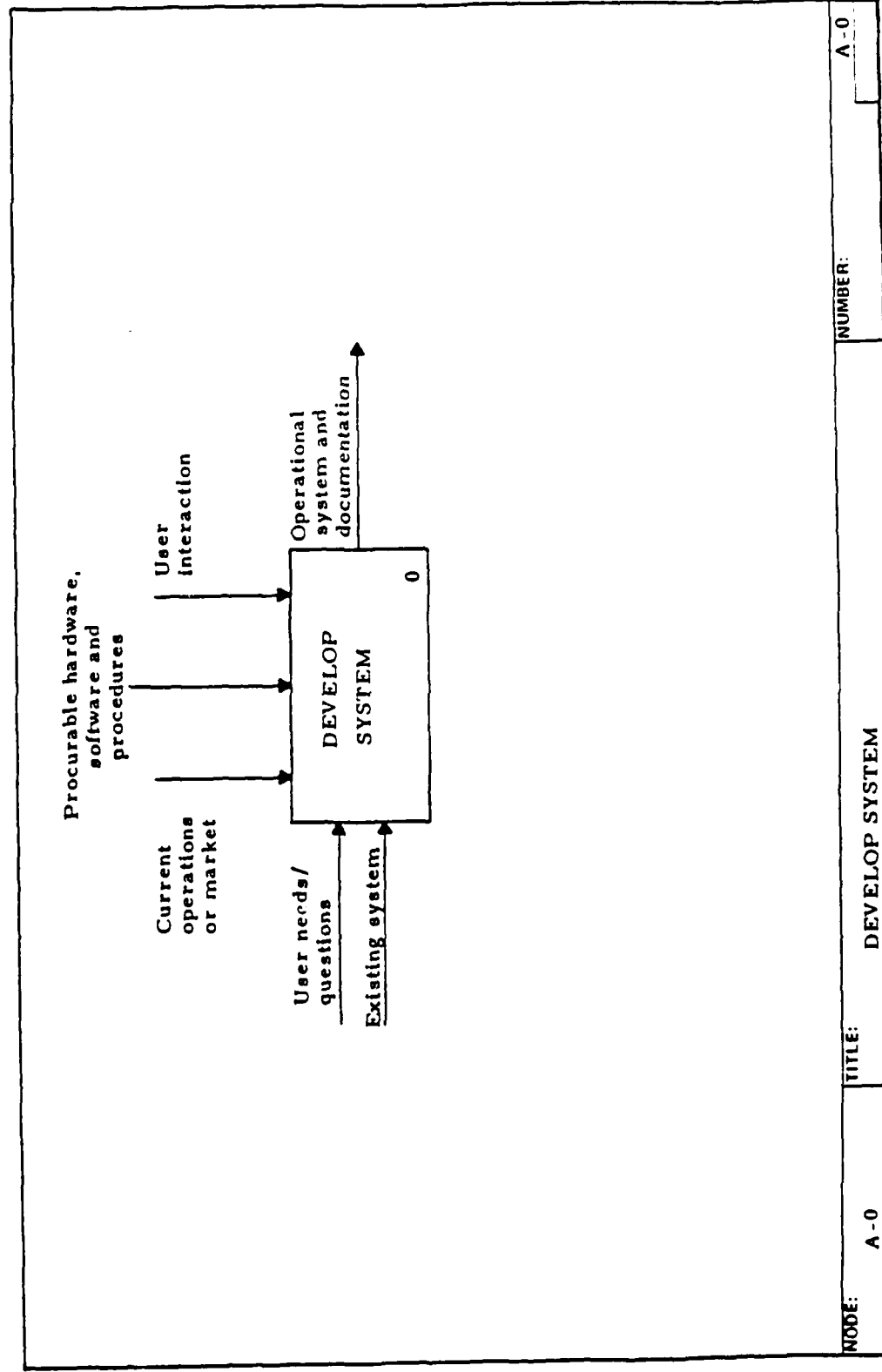


CORRESPONDING NODE TREE

INSTRUCTOR NOTES

FOR THE NEXT 3 SLIDES, WALK THROUGH THE DIAGRAMS AND HIGHLIGHT THE MAIN PATH. TELL STUDENTS THIS IS JUST AN EXAMPLE AND THE MODEL IS NOT NECESSARILY CORRECT.

MODEL

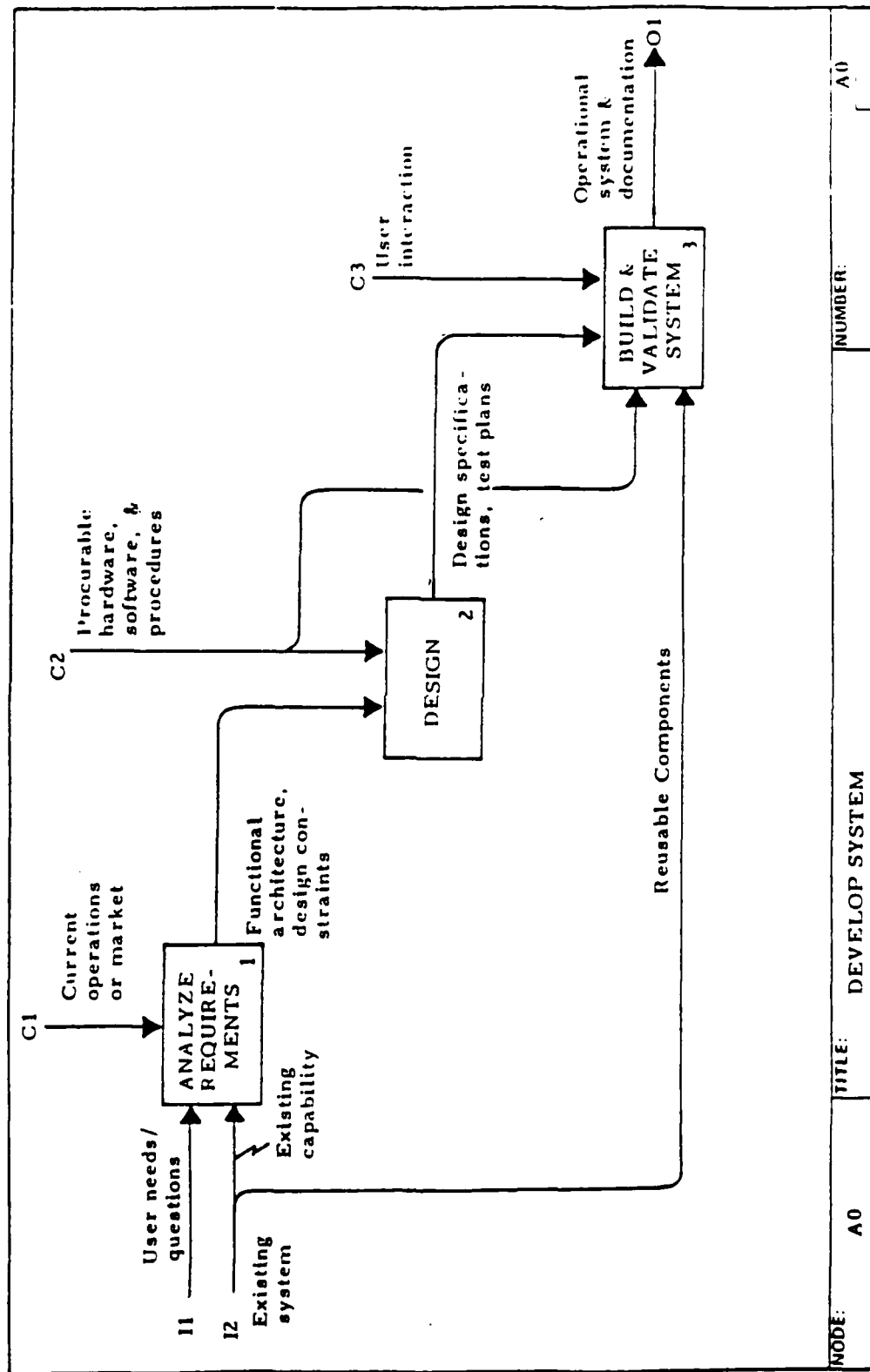


INSTRUCTOR NOTES

THE FIRST DECOMPOSITION YIELDS . . .

POINT OUT HOW C2 IS USED AS BOTH A CONTROL AND AS AN INPUT.

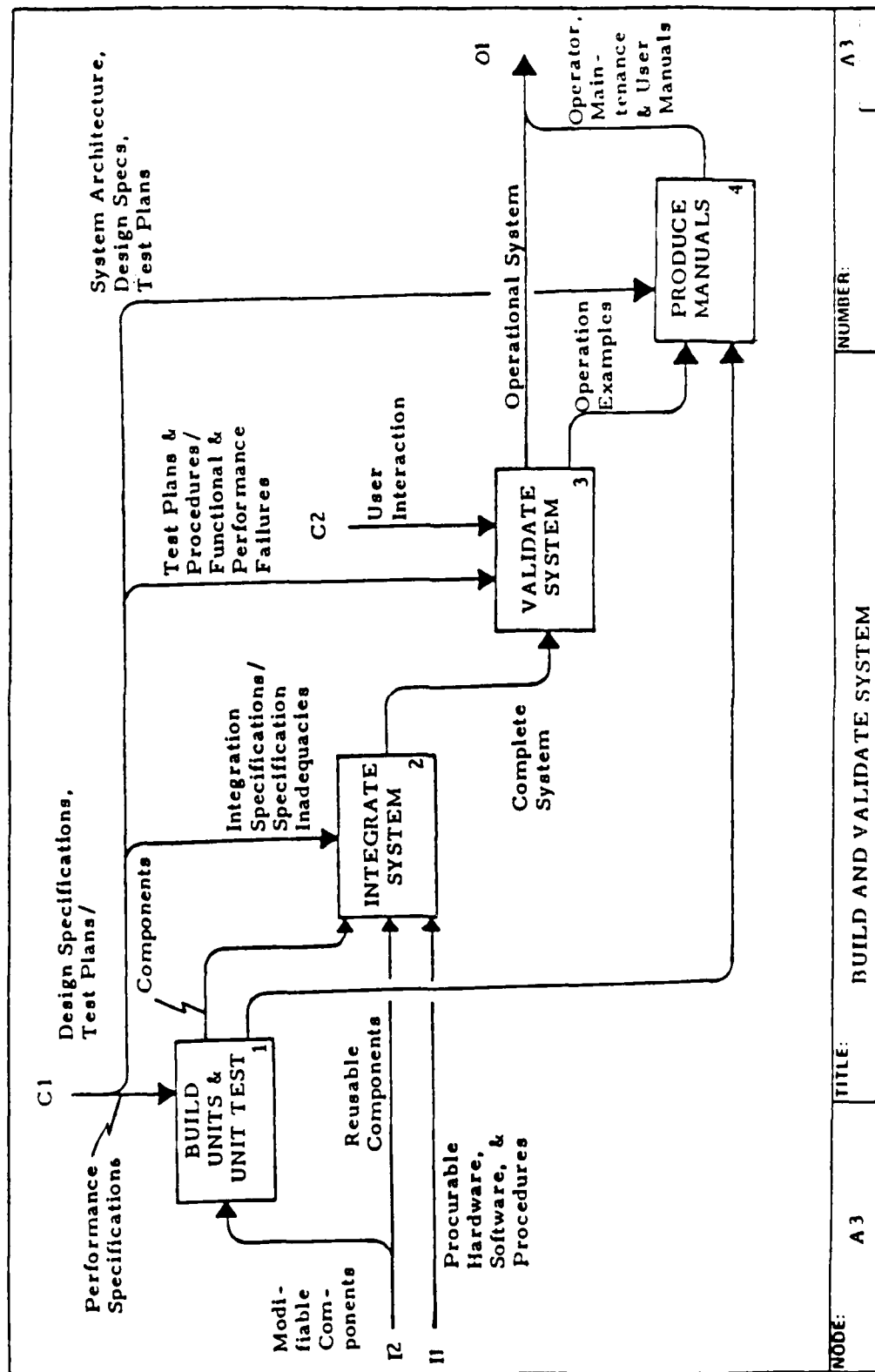
MODEL



INSTRUCTOR NOTES

POINT OUT THAT IN THE PREVIOUS DECOMPOSITION, WE NEEDED ONLY 3 BOXES TO SHOW THE ACTIVITIES AT THAT LEVEL. HERE WE NEED 4 BOXES.

MODEL



INSTRUCTOR NOTES

ANY QUESTION ABOUT THE SYSTEM SHOULD BE ANSWERABLE FROM EVERY VIEWPOINT, ALTHOUGH THE IMPORTANCE GIVEN BY A SINGLE VIEWPOINT MAY NOT MATCH THAT OF A DIFFERENT ONE.

A SINGLE MODEL THAT ADDRESSES ALL VIEWPOINTS SIMULTANEOUSLY IS AS COMPLEX TO BUILD AS THE SYSTEM ITSELF.

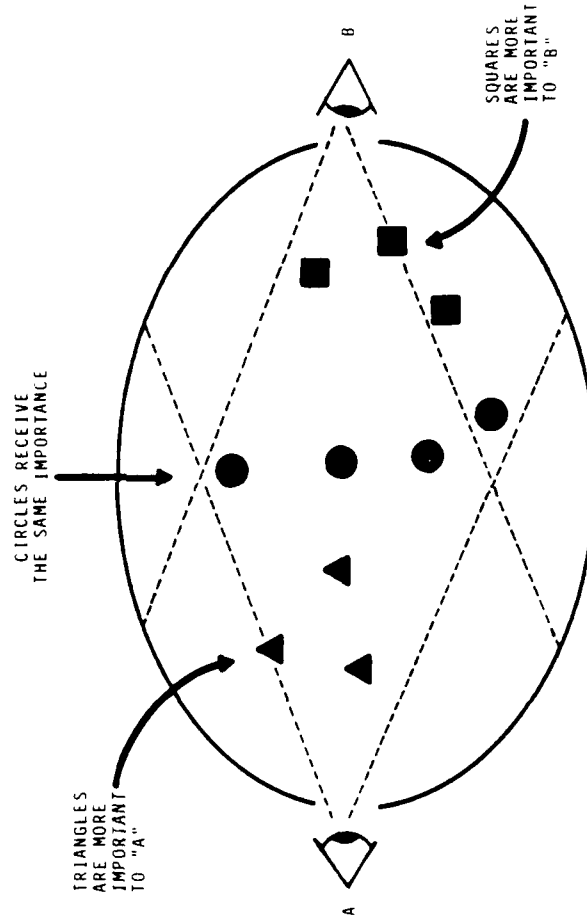
IT IS IMPORTANT TO UNDERSTAND THE VIEWPOINT OF THE AUTHOR OF THE SADT DIAGRAM.

LABEL DIFFERENT EYES AS MANAGER, OPERATOR, OR MAINTAINER, FOR EXAMPLE.

USE ANALOGY OF 2 PEOPLE LOOKING AT A GROUP OF OBJECTS FROM OPPOSITE ENDS OF A LARGE CONTAINER. NOTE THAT DISTANCE OF OBJECTS CORRESPONDS TO EMPHASIS: CLOSER OBJECTS ARE MORE IMPORTANT.

VIEWPOINT

- A MODEL IS MADE FROM A VIEWPOINT
- A SYSTEM CAN BE MODELED FROM SEVERAL VIEWPOINTS WHICH DIFFER IN
 - EMPHASIS
 - TERMINOLOGY
 - DETAIL
 BUT ALWAYS COVER THE SAME SUBJECT

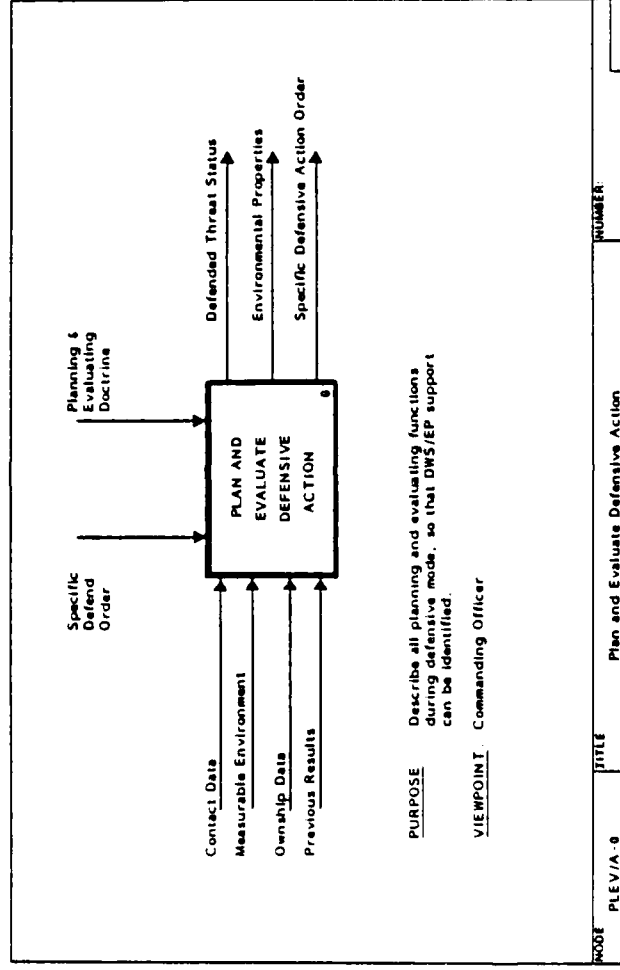


- VIEWPOINTS ARE SEPARATED IN SADT BY BUILDING SEPARATE MODELS

INSTRUCTOR NOTES

GO THROUGH THIS EXAMPLE CAREFULLY, POINTING OUT ALL THE KEY CONCEPTS THAT HAVE BEEN PREVIOUSLY EXPLAINED. DISCUSS THE A-0, OR HIGHEST LEVEL DIAGRAM, AND HOW EVERYTHING CAN BE RELATED TO THIS DIAGRAM.

SAMPLE APPLICATION



A-0

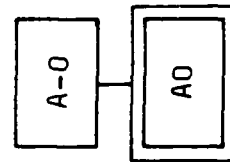
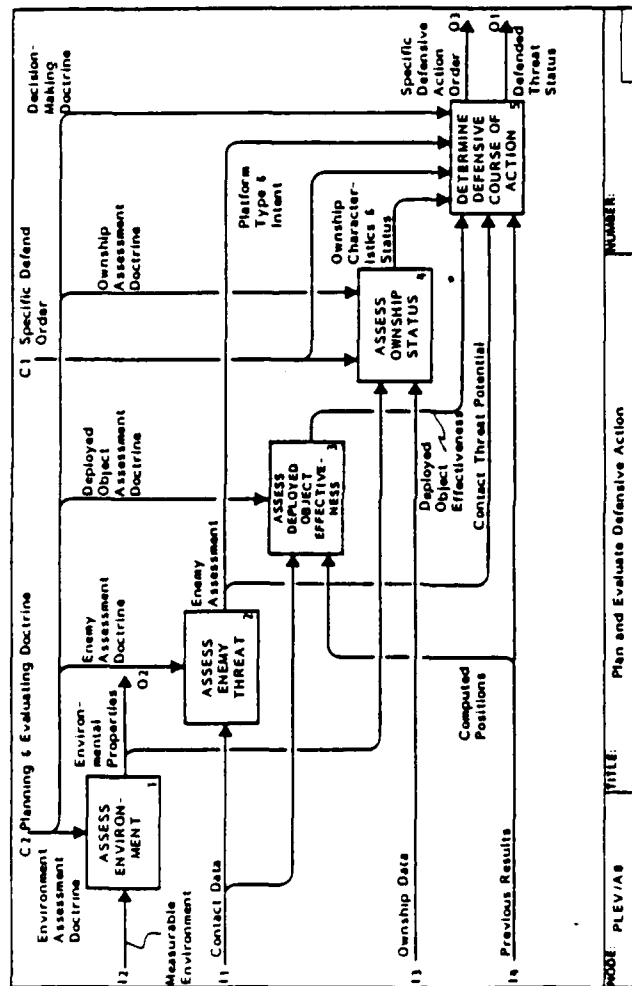
- MODEL OF PLANNING AND EVALUATION FUNCTION FOR A NAVAL PLATFORM DURING DEFENSIVE ACTIONS
- PURPOSE IS TO IDENTIFY MAJOR NEW SOFTWARE FUNCTIONS OF THE DEFENSIVE WEAPON SUBSYSTEM

INSTRUCTOR NOTES

POINT OUT THE DECOMPOSITION AND INPUTS AND OUTPUTS. RELATE A-0 TO A0.

NOTE ASSESS ENEMY THREAT WILL BE DECOMPOSED ON NEXT SLIDE.

SAMPLE APPLICATION



MODEL HIERARCHY

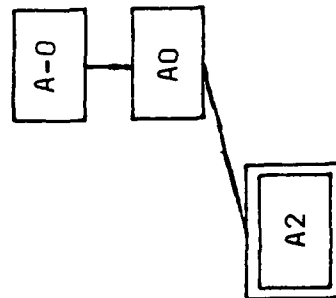
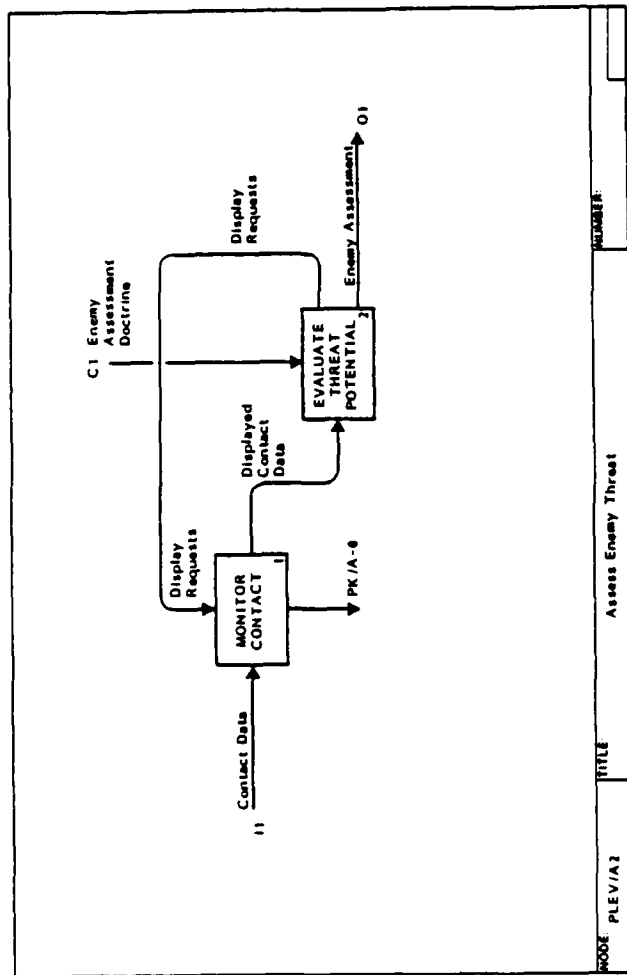
INSTRUCTOR NOTES

POINT OUT THAT THIS DIAGRAM IS A DECOMPOSITION OF ASSESS ENEMY THREAT.

VG 778.1

6-26i

SAMPLE APPLICATION



MODEL HIERARCHY

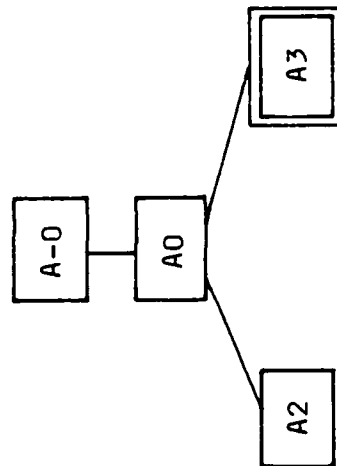
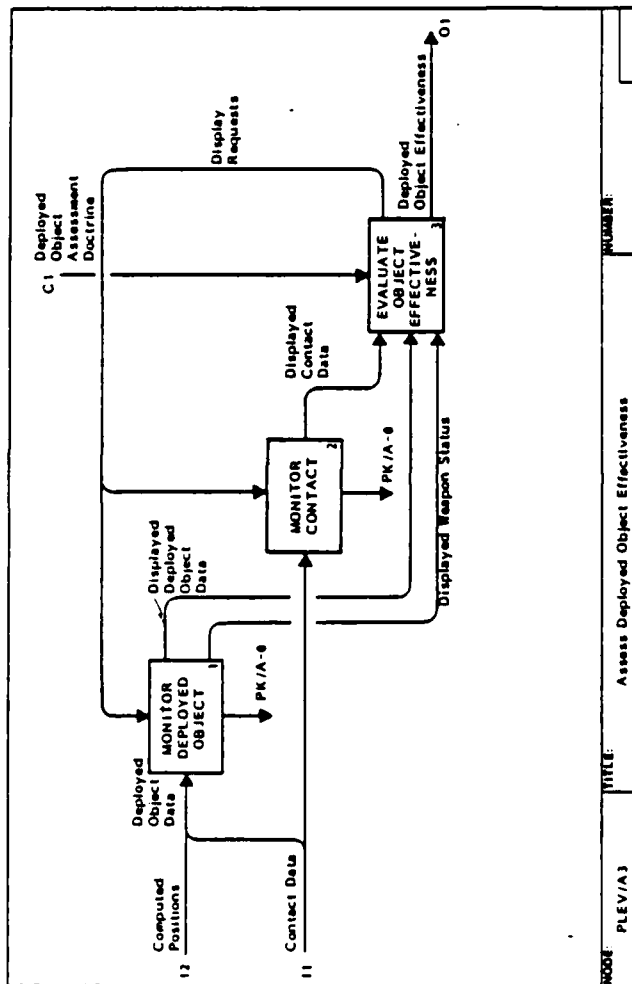
INSTRUCTOR NOTES

TRACE BACK AN INPUT OR OUTPUT TO THE A-0 LEVEL

VG 778.1

6-27i

SAMPLE APPLICATION



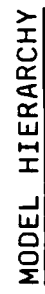
MODEL HIERARCHY

INSTRUCTOR NOTES

DISCUSS HOW INTUITIVE AND STRAIGHTFORWARD THE DECOMPOSITION IS ITSELF.

```

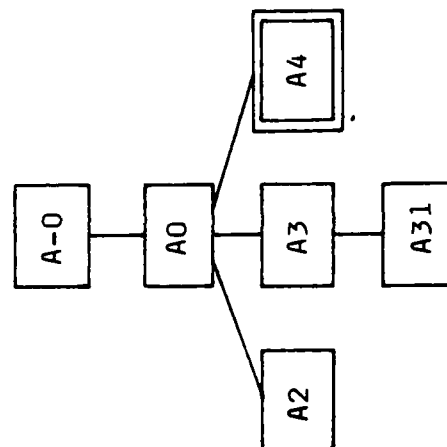
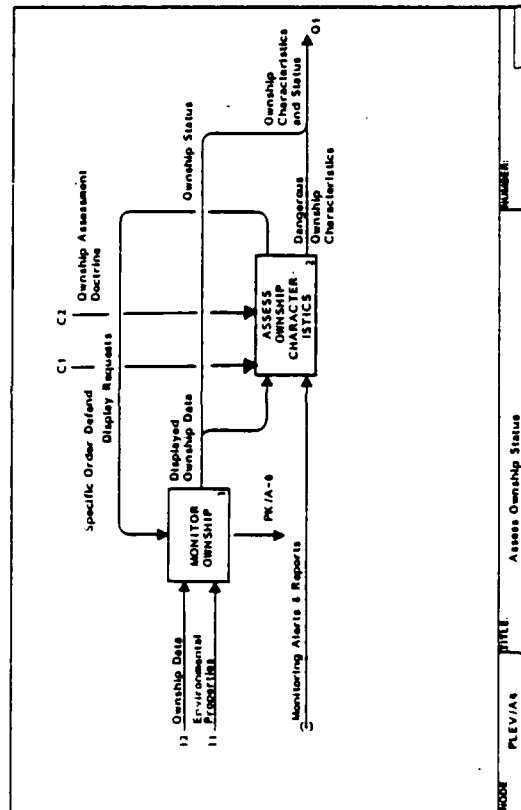
graph TD
    C1[C1 Display Requests] --> MC1[MONITOR CM 1]
    MC1 --> PKA8_1[PK/A-8] --> MM2[MONITOR MOSS 2]
    MC1 --> DCM[Displayed CM Data]
    MM2 --> PKA8_2[PK/A-8] --> MW3[MONITOR WEAPON 3]
    MM2 --> DMOSS[Displayed MOSS Data]
    MW3 --> PKA8_3[PK/A-8]
    MW3 --> DW[Displayed Weapon Data]
    DCM --> DOD[Displayed Object Data]
    DMOSS --> DOD
    DW --> DOD
    DOD --> DOS[Displayed Object Status]
  
```



INSTRUCTOR NOTES

THIS IS THE LAST EXAMPLE DIAGRAM. DISCUSS HOW FROM THIS SIMPLE MODEL WE CAN GAIN A BETTER UNDERSTANDING OF THE SYSTEM AND ARE ABLE TO QUANTIFY OUR KNOWLEDGE OF THE SYSTEM BY THE DEPTH OF THE DIAGRAM (E.G., WE ARE ABLE TO GO TO THE A31 LEVEL BUT NOT A41 OR EVEN A1).

SAMPLE APPLICATION



MODEL HIERARCHY

INSTRUCTOR NOTES

IDEF IS CURRENTLY USED TO MODEL U.S. AEROSPACE INDUSTRY.

SADT IS COMPATIBLE WITH MANY TECHNIQUES CURRENTLY USED OR DEVELOPED BY MANY GOVERNMENT ORGANIZATIONS.

SADT EXTENSIONS

- IDEF* METHODS HAVE EXTENDED AND AUTOMATED SADT
 - ADDED BEHAVIORAL AND INFORMATIONAL MODELS
 - AUTO IDEF AUTOMATES PRODUCTION AND MAINTENANCE OF IDEF MODELS
 - EXTENDED APPLICATION TO MANUFACTURING MODELING
- OTHER AUTOMATED TOOLS AND PAIRING OF METHODOLOGIES HAVE BEEN DEVELOPED
 - SADT AND PSL/PSA
 - SADT AND SIMULATION LANGUAGES
 - SADT AND SCRP

*INTEGRATED COMPUTER AIDED MANUFACTURING DEFINITION PROJECT - U.S. AIR FORCE

INSTRUCTOR NOTES

THEME:

SREM IS ONE OF THE ATTEMPTS TO AUTOMATE THE ANALYSIS PROCESS USING A MIX OF GRAPHICAL AND TEXTUAL NOTATIONS.

PURPOSE:

TO PROVIDE AN OVERVIEW OF THE METHOD AND CHARACTERISTICS OF THE NOTATION USED FOR DOCUMENTING REQUIREMENTS THE LEVEL OF PRESENTATION SHOULD BE LIGHT HIGHLIGHTING OF KEY POINTS NOT A DETAIL BY DETAIL EXPOSURE.

REFERENCES:

1. ALFORD, M. "SOFTWARE REQUIREMENTS ENGINEERING METHODOLOGY (SREM) AT THE AGE OF FOUR" TRW, A1; JUNE 1980.
2. LOSHBOUGH, R. "APPLICABILITY OF SREM TO THE VERIFICATION OF MANAGEMENT INFORMATION SYSTEM SOFTWARE REQUIREMENTS" TRW, A1; APRIL 1980.

Section 7

SREM METHODOLOGY

"AN AUTOMATED METHODOLOGY FOR REALTIME SYSTEMS"

AD-A165 300

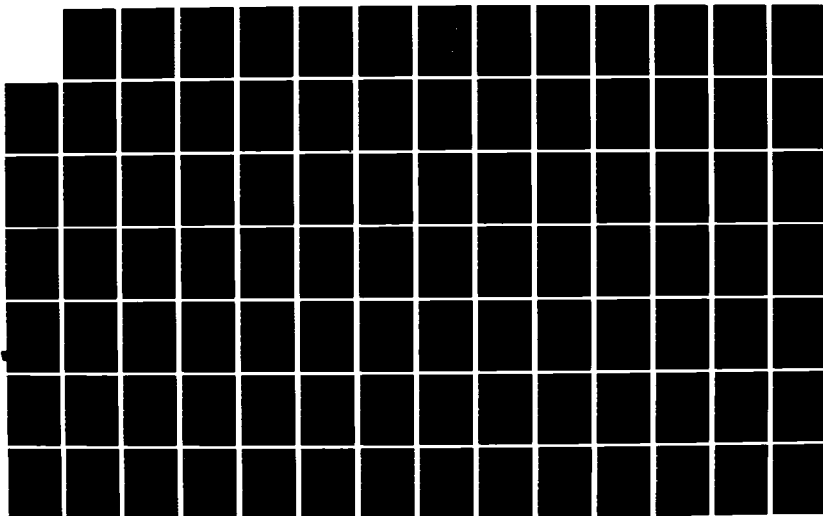
ADA (TRADEMARK) TRAINING CURRICULUM SOFTWARE
ENGINEERING METHODOLOGIES M201 TEACHER'S GUIDE VOLUME 1
(U) SOFTECH INC WALTHAM MA 1986 DAAB07-83-C-K506

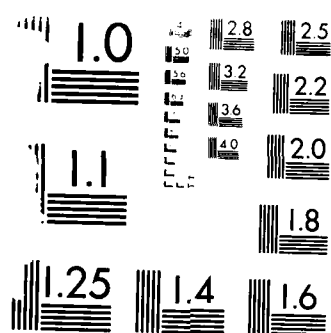
3/5

UNCLASSIFIED

F/G 5/9

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

INSTRUCTOR NOTES

THESE ARE THE FOUNDATION IDEAS OF SREM. MAKE SURE THE CLASS UNDERSTANDS THEM THOROUGHLY.

THE EMPHASIS IS ON INTERFACES - DATA AS WELL AS CONTROL.

SREM BEGINS WITH THE TRANSLATION AND DECOMPOSITION OF SYSTEM LEVEL REQUIREMENTS. AS PART OF THE METHODOLOGY, A SET OF SOFTWARE SUPPORT TOOLS WERE IMPLEMENTED TO AUTOMATE MANY OF THE PREVIOUSLY MANUAL ACTIVITIES ASSOCIATED WITH REQUIREMENTS ENGINEERING. THESE SOFTWARE TOOLS FORM THE REQUIREMENTS ENGINEERING AND VALIDATION SYSTEM (REVS); REVS PROCESSING IS ACCOMPLISHED BY EXPRESSION OF THE SOFTWARE REQUIREMENTS IN THE STRUCTURED, FORMAL REQUIREMENTS SPECIFICATION LANGUAGE (RSL). A KEY CONCEPT OF REVS IS THAT ALL REQUIREMENTS ARE TRANSLATED INTO A CENTRAL DATABASE CALLED THE ABSTRACT SYSTEM SEMANTIC MODEL (ASSM). THE RSL STATEMENTS THEMSELVES ARE NOT STORED IN THE ASSM. INSTEAD, THEY ARE TRANSLATED INTO REPRESENTATIONS OF THE INFORMATION CONTENT OF THE REQUIREMENTS STATEMENTS. THIS PROVIDES AN EFFICIENT AND FLEXIBLE MEANS OF MAINTAINING A LARGE SOFTWARE SPECIFICATION IN A RELATIVELY SMALL COMPUTER DATABASE.

SOFTWARE REQUIREMENTS ENGINEERING METHODOLOGY (SREM)

KEY CONCEPTS

- A SYSTEM IS COMPRISED OF MANY SUBSYSTEMS THAT COMMUNICATE VIA MESSAGES PASSED THROUGH COMMON INTERFACES.
- A SYSTEM OR SUBSYSTEM RESPONDS TO EACH (EXTERNAL) EVENT BY GENERATING A RESPONSE.
- EXPRESS REQUIREMENT THROUGH THE USE OF A LANGUAGE
 - REQUIREMENTS STATEMENT LANGUAGE (RSL)
 - DOCUMENTS SUBSYSTEMS, PROCESSES, MESSAGES AND INTERFACES
- EXPRESS RESPONSES TO EXTERNAL EVENTS USING RESPONSE-NETWORK GRAPHS (R-NETS)
- MANUAL ACTIVITIES ASSOCIATED WITH REQUIREMENTS SPECIFICATION AUTOMATED UNDER THE REQUIREMENTS ENGINEERING AND VALIDATION SYSTEM (REVS)

INSTRUCTOR NOTES

THE LANGUAGE, REQUIREMENTS STATEMENT LANGUAGE (RSL), IS FAIRLY SIMPLE IN APPEARANCE. IT IS VERY "ENGLISH-LIKE" IN STRUCTURE, HAS A RESTRICTED VOCABULARY AND GRAMMAR, AND IS VERY READABLE; IT IS ALSO A MACHINE READABLE LANGUAGE. THIS CHART IS AN EXAMPLE OF RSL. HERE DATA ARE DECLARED AND GIVEN A NAME. THERE ARE VARIOUS ADJECTIVES SUCH AS DESCRIPTION, MAXIMUM VALUE, MINIMUM VALUE, AND UNITS. THE RELATIONSHIPS BETWEEN THAT PARTICULAR PIECE OF DATA AND OTHER ITEMS IS ALSO SHOWN, E.G., "INPUT TO," "PASSED BY." FOR EXAMPLE, THE MESSAGE "OPERATOR_INQUIRY" IS PASSED BY THE "REMOTE_TERMINAL_INPUT_INTERFACE." IT IS MADE UP OF DATA "OPERATOR_ID," "QUERY_TYPE," AND "ITEM_NAME."

THE LANGUAGE ITSELF IS A PRIMITIVE LANGUAGE WHICH CONTAINS FOUR CONCEPTS: 1) ELEMENTS WHICH ARE THE NOUNS OF THE SYSTEM, SUCH AS DATA, MESSAGES, AND INTERFACES; 2) ATTRIBUTES OF ELEMENTS, SUCH AS DESCRIPTION, MAXIMUM VALUE, MINIMUM VALUE; 3) RELATIONSHIPS BETWEEN ELEMENTS SUCH AS INPUT TO, OUTPUT FROM; AND 4) STRUCTURES. RSL ALSO INCLUDES THE ABILITY TO EXTEND THE LANGUAGE AND ADD NEW CONCEPTS. FOR EXAMPLE, "DATA" CAN BE CHANGED TO "INFORMATION," "MAXIMUM VALUE" COULD BE CALLED "MAXIMUM_ALLOWED_VALUE." IF A NEW MEANINGFUL CONCEPT IS NEEDED IT CAN BE ADDED.

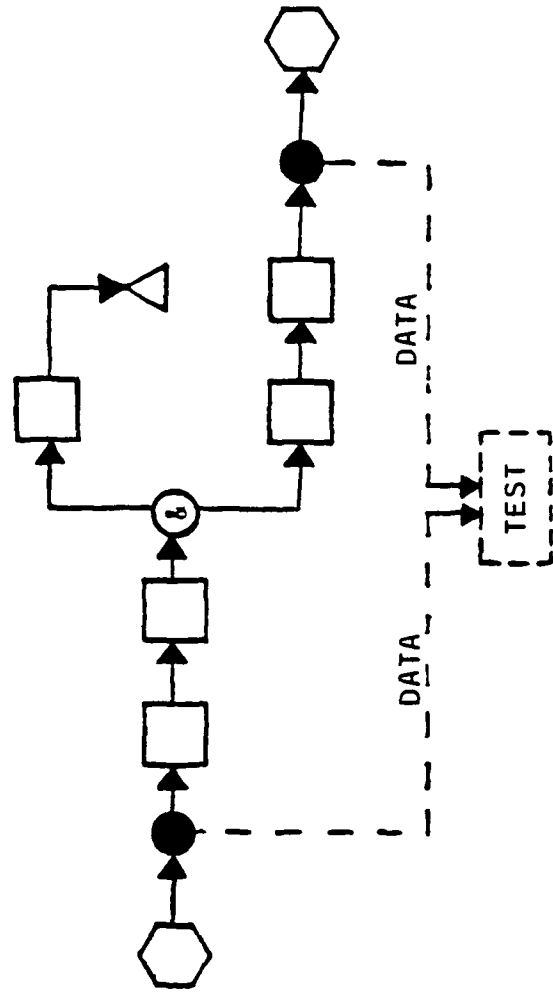
REQUIREMENT STATEMENT LANGUAGE

● RSL IS SIMPLE, PRECISE, AND NON-AMBIGUOUS:

DATA:	RANGE_ESTIMATE,
DESCRIPTION:	"THIS DATA ITEM REPRESENTS THE CURRENT ESTIMATE OF SLANT RANGE TO THE TARGET."
MAXIMUM_VALUE:	100000,
MINIMUM_VALUE:	100,
UNITS:	METERS,
INPUT TO:	PREDICT_INTERCEPT,
OUTPUT FROM:	ESTIMATE_TARGET_STATE,
MESSAGE:	OPERATOR_INQUIRY,
PASSED BY:	REMOTE_TERMINAL_INPUT_INTERFACE,
MADE BY:	OPERATOR_ID QUERY_TYPE ITEM_NAME,

RESPONSE-NETWORK

- DOCUMENTS HOW THE SYSTEM RESPONDS TO AN EXTERNAL EVENT
- STRUCTURING BY PATHS YIELDS TESTABILITY
- INTEGRATING PATHS INTO R-NETS YIELDS COMPREHENDABILITY
- TESTABLE PERFORMANCE REQUIREMENTS ARE EASILY EXPRESSED USING R-NETS



INSTRUCTOR NOTES

EXPERIENCE INDICATES THAT AN APPROACH, LANGUAGE, AND TOOLS ARE NOT ENOUGH TO ENSURE THE DEVELOPMENT OF GOOD SOFTWARE REQUIREMENTS. A METHODOLOGY IS NEEDED TO GUIDE THE EFFORT AND MAKE PROGRESS VISIBLE VIA EVALUABLE MILESTONES. THE RESULTING SOFTWARE REQUIREMENTS ENGINEERING METHODOLOGY TO GENERATE THESE REQUIREMENTS IS ILLUSTRATED ON THIS VIEWGRAPH. IT STARTS WITH A SYSTEM SPECIFICATION WHICH CAN BE A FORMAL SPECIFICATION OF A SYSTEM, A CONVERSATION WITH THE INTENDED USER, OR A MENTAL IMAGE OF THE SYSTEM. THIS SPECIFICATION IS TRANSLATED AND INTERPRETED TO DETERMINE THE INTERFACES TO THE OUTSIDE WORLD, THE MESSAGES ACROSS THESE INTERFACES, AND THE REQUIRED PROCESSING RELATIONSHIPS AND FLOWS. DURING THIS PROCESS, SPECIFIC PROBLEMS IN THE SYSTEM SPECIFICATION WILL BE FOUND, E.G., AMBIGUITIES AND INCONSISTENCIES. THESE PROBLEMS ARE CORRECTED BY AN ITERATIVE PROCESS UNTIL THE SPECIFICATION OF THE SYSTEM IS SATISFACTORY. NEXT, THE DETAILS OF THE FUNCTIONAL REQUIREMENTS, INCLUDING ALL OF THE INPUT/OUTPUT DATA RELATIONSHIPS, THE PROCESSING STEPS, THE ATTRIBUTES AND MAXIMUM VALUES, MINIMUM VALUES, AND THE ALLOWED DATA RANGES ARE COMPLETED. THE FORMAL LANGUAGE IS USED TO INPUT THESE REQUIREMENTS INTO REVS AND THE STATIC ANALYZER IS EXECUTED TO TEST FOR ERRORS IN CONSISTENCY AND COMPLETENESS.

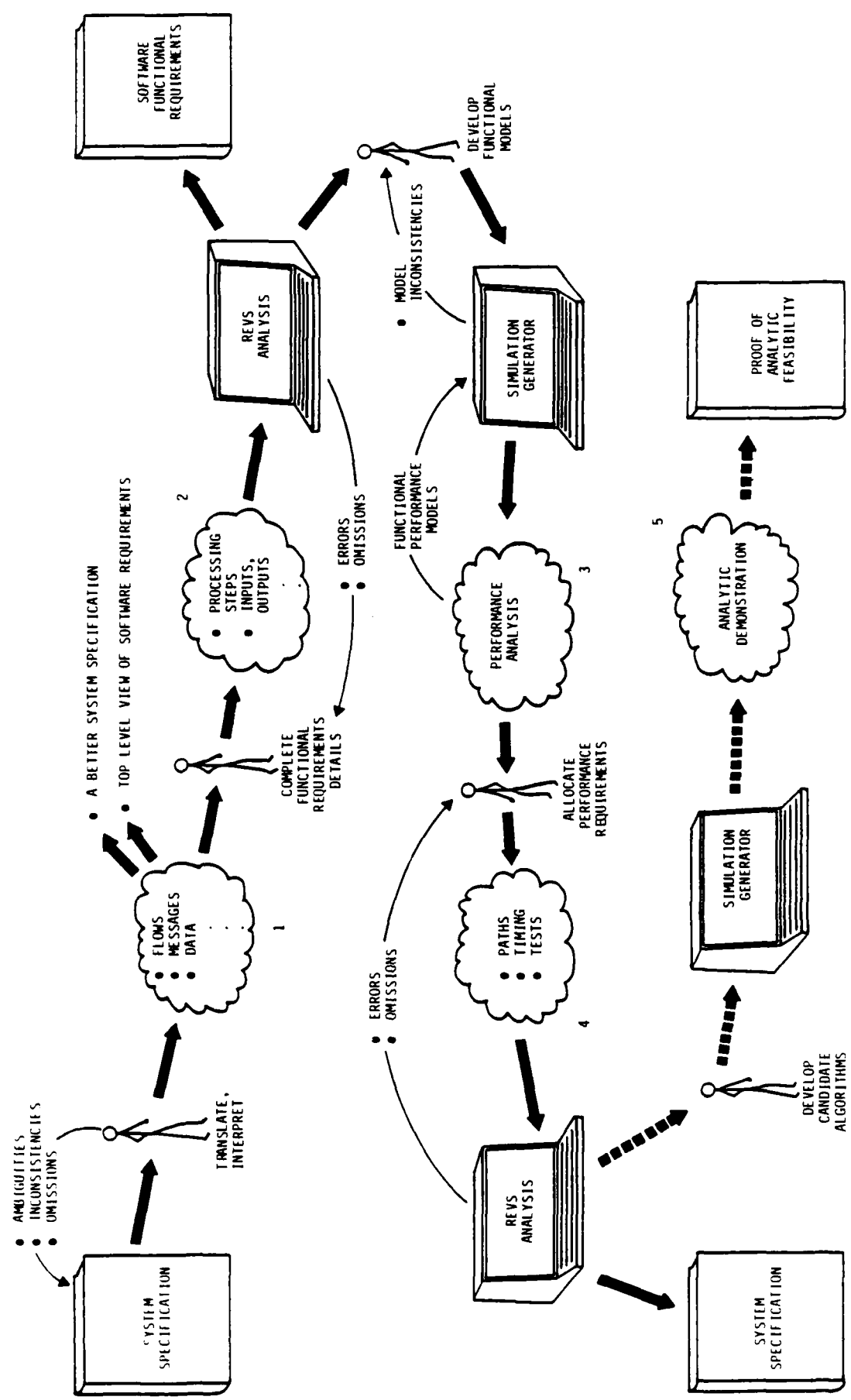
WHEN ALL INFORMATION HAS BEEN INPUT AND ALL ERRORS CORRECTED, THE RESULT IS A FUNCTIONAL SPECIFICATION. BEFORE THE FUNCTIONAL SPECIFICATION CAN BE VALIDATED, A SIMULATION OF THE SYSTEM IS NEEDED. FIRST, SIMPLE FUNCTIONAL MODELS ARE DEVELOPED FOR EACH OF THE PROCESSING STEPS AND PUT THROUGH THE SIMULATION GENERATION FUNCTION MENTIONED EARLIER. ONCE ALL THE MODELS ARE BUILT, THE SIMULATOR IS EXECUTED TO VERIFY THE ENTIRE PROCESS.

AFTER THE FUNCTIONAL REQUIREMENT SPECIFICATION IS VALIDATED, THE PERFORMANCE REQUIREMENTS ARE DEVELOPED. THE PERFORMANCE REQUIREMENTS USUALLY AREN'T WELL STRUCTURED; THEY ARE STATED IN SYSTEM TERMS SUCH AS "KILL PROBABILITY" AND "MISS DISTANCES" FOR WHICH THE SOFTWARE SHARES ONLY PARTIAL RESPONSIBILITY. THE PATHS OF THE DEFINED PROCESSING MUST BE MAPPED AND THE PATHS WHICH ARE BEING CONSTRAINED BY THESE PERFORMANCE REQUIREMENTS MUST BE IDENTIFIED. ANOTHER VERSION OF THE FUNCTIONAL SIMULATOR WHICH INCLUDES ACTUAL PERFORMANCE MODELS MAY BE NEEDED. TRADE STUDIES AT THIS LEVEL WILL BE PERFORMED UNTIL THE RIGHT PERFORMANCE REQUIREMENTS ARE ALLOCATED FOR EACH PATH.

ALL OF THE PATHS ARE NOW IDENTIFIED, THE TIMING AND ANALYTIC ACCURACY REQUIREMENTS ARE SPECIFIED FOR EACH OF THE PATHS, AND ALL OF THIS INFORMATION IS INPUT TO THE REVS DATABASE. COMPLETION OF THESE STEPS RESULTS IN A SOFTWARE SPECIFICATION.

BEFORE ATTEMPTING TO BUILD EXPENSIVE REALTIME CODE, IT MAY BE NECESSARY TO VERIFY THAT THE SYSTEM IS ANALYTICALLY FEASIBLE. TO DO THIS, ONE MORE SIMULATION STEP CALLED ANALYTICAL FEASIBILITY DEMONSTRATION WILL BE PERFORMED. THIS STEP WILL USE REAL ALGORITHMS INSTEAD OF FUNCTIONAL MODELS. IT MAY NOT RUN IN REAL TIME, BUT IT SHOULD CONSIST OF REAL TRACKING ALGORITHMS AND REAL GUIDANCE ALGORITHMS. IT SHOULD BE DRIVEN BY A DRIVER WITH ENOUGH FIDELITY TO REPRESENT THE BEST UNDERSTANDING OF THE ENVIRONMENT AND MEASUREMENTS OF THE RADAR, THE CONFIGURATION OF THE MISSILE, ETC., AND VERIFY THAT THE SOFTWARE WILL ACTUALLY WORK.

SREM PROVIDES AN ORDERLY, MANAGEABLE ENGINEERING OF REQUIREMENTS



INSTRUCTOR NOTES

BENEFITS OF USING SREM FOR THE ENGINE MONITORING SYSTEM EXAMPLE TO FOLLOW LATER ARE LISTED HERE. THESE PROBLEMS ARE DETECTED EARLY IN THE SYSTEM SPECIFICATION DEVELOPMENT. THEREFORE, THEY CAN BE CORRECTED WITH MUCH LESS COST IMPACT THAN IF DISCOVERED LATER IN THE DESIGN OR TESTING PHASES.

FOR EXAMPLE, THE ORIGINATING SYSTEM REQUIREMENTS DID NOT SPECIFY WHETHER ENGINES COULD BE MONITORED AT DIFFERENT RATES; THE RSL REQUIREMENTS REQUIRE THAT THE SOFTWARE CONTAIN A PARAMETER RATE FOR EACH ENGINE TO BE MONITORED, AND THAT THIS DATA BE USED TO SELECT THE MEASUREMENTS TO BE MADE. IF THIS IS NOT A DESIRED CAPABILITY OF THE SYSTEM, THE RSL CAN BE REVISED, BUT IT IS NOT AMBIGUOUS. THE OTHER QUESTIONS ARE ANSWERABLE FROM THE RSL IN SIMILAR FASHION.

THIS TYPE OF AUTOMATED APPROACH HAS DEFINITE ADVANTAGES OVER PURELY MANUAL ANALYSES.

SOME BENEFITS OF USING SREM

- THE LANGUAGE AND METHODS PRECLUDE TYPICAL AMBIGUITIES, SUCH AS:
 - CAN THE ENGINES BE MONITORED AT DIFFERENT RATES?
 - DOES "OUTPUT WARNING" MEAN EACH TIME OR JUST THE FIRST TIME?
 - ARE "PRESCRIBED LIMITS" ESTABLISHED FOR EACH PARAMETER OR ARE THERE ONLY ONE MINIMUM AND ONE MAXIMUM?
 - ARE THE "PRESCRIBED LIMITS" FOR "WARNING" DIFFERENT THAN THOSE FOR "ALARM"?
- AUTOMATED ERROR CHECKS DETECT ERRORS OF CONSISTENCY AND COMPLETENESS, SUCH AS:
 - HOW ARE "PRESCRIBED LIMITS" DEFINED?
 - HOW QUICKLY MUST THE SYSTEM OUTPUT A "WARNING" OR "ALARM"?
 - WHAT DOES A "WARNING MESSAGE" CONTAIN?
 - WHAT IS TO BE DONE WITH THE HISTORY DATA FOR EACH ENGINE?
- AUTOMATED DOCUMENTATION PAYS HIGH DIVIDENDS
 - ONE COMPUTER RUN YIELDS COMPLETE SPECIFICATION WITH ALL CURRENT REVISIONS.
 - SAME THOROUGHNESS IS APPLIED TO EVERY REVISION -- LAST MINUTE PANICS DO NOT INTRODUCE UNDETECTED ERRORS.

INSTRUCTOR NOTES

THE SREM APPLICATIONS ARE SUMMARIZED HERE. LISTED FIRST ARE THE AREAS FROM DEVELOPMENT OF THE ORIGINAL SPECIFICATION OF SOFTWARE REQUIREMENTS THROUGH TESTING AND ACCEPTANCE. SECOND ARE THE AREAS OF VERIFICATION AND VALIDATION.

RECALL THAT SREM WAS DEVELOPED SPECIFICALLY TO SUPPORT THE GENERATION OF REALTIME SOFTWARE REQUIREMENTS. IT IS THE ONLY TECHNIQUE CURRENTLY KNOWN TO ADDRESS THE ISSUES RAISED BY THE NEW DoD 5000.29 REGULATIONS FOR SOFTWARE SPECIFICATION PRIOR TO DSARC II APPROVAL.

WE HAVE ALSO OBSERVED THAT, WITH MINOR MODIFICATION, SREM, REVS, AND RSL CAN BE USED TO PERFORM AN INDEPENDENT VALIDATION OF EXISTING SPECIFICATIONS, INCLUDING V & V OF THE FUNCTIONAL REQUIREMENTS, PERFORMANCE REQUIREMENTS, AND INDEPENDENT CONFIRMATION OF SIMULATION RESULTS.

APPLICATIONS OF SREM

- WITHIN A DEVELOPMENT PROJECT

- ORIGINAL SPECIFICATION OF SOFTWARE REQUIREMENTS
- VALIDATION OF SOFTWARE REQUIREMENTS PRIOR TO COMMITTING DEVELOPMENT RESOURCES
- INSTRUMENTAL TO OBTAINING DSARC APPROVAL PER DOD 5000.29
- BASIS FOR SOFTWARE TESTING/ACCEPTANCE

- WITHIN AN INDEPENDENT V&V PROGRAM

- STRUCTURED, THOROUGH ANALYSIS OF SOFTWARE SPECIFICATIONS
- QUICKLY ISOLATES AREAS OF AMBIGUITY, INCONSISTENCY
- INDEPENDENT CONFIRMATION OF DEVELOPER'S SIMULATIONS

INSTRUCTOR NOTES

SREM ENCOMPASSES EIGHT MAJOR PHASES OR STEPS OF ENGINEERING ACTIVITY THAT BEGIN WITH RECEIPT OF THE SET OF INFORMATION WHICH DEFINES THE SYSTEM LEVEL REQUIREMENTS ON THE DATA PROCESSING SUBSYSTEM. THIS INPUT SET IS DENOTED AS THE DATA PROCESSING SYSTEM PERFORMANCE REQUIREMENTS (DPSPR) SPECIFICATION. THE DPSPR INCLUDES THE DATA PROCESSING SYSTEM INTERFACE REQUIREMENTS SPECIFICATIONS AND ANY EXTERNAL SUBSYSTEM PERFORMANCE REQUIREMENTS SPECIFICATIONS WHICH INFLUENCE THE DEFINITION OF THE PROCESS PERFORMANCE REQUIREMENTS. USING THESE SOURCE DOCUMENTS AS A STIMULUS, THE REQUIREMENTS ENGINEER BECOMES INVOLVED IN THE EIGHT MAJOR ENGINEERING ACTIVITIES DEFINED BY SREM TO DEVELOP THE PROCESS PERFORMANCE REQUIREMENTS SPECIFICATION. THESE ENGINEERING ACTIVITIES ARE:

- IDENTIFICATION, DEFINITION, AND DEVELOPMENT OF THE FUNCTIONAL REQUIREMENTS,
- IDENTIFICATION, DEFINITION, AND DEVELOPMENT OF THE PERFORMANCE REQUIREMENTS,
- DEVELOPMENT OF THE PROCESS PERFORMANCE REQUIREMENTS SPECIFICATION, AND
- DEVELOPMENT OF THE ANALYTIC FEASIBILITY DEMONSTRATIONS.

THE ACTIVITIES DEFINED BY SREM THAT LEAD TO SPECIFICATION AND VALIDATION OF THE FUNCTIONAL REQUIREMENTS (ACTIVITY ONE ABOVE) HAVE BEEN SEGMENTED INTO THE FIRST FIVE PHASES LISTED ON THIS CHART. PHASES 6 LEADS TO ACTIVITIES TWO AND THREE ABOVE; PHASE 7 TO ACTIVITY FOUR ABOVE.

STEP 0: ASSEMBLE DOCUMENTATION

- SOURCE DOCUMENTATION MAY INCLUDE
 - SYSTEM DEFINITION
 - SUBSYSTEMS SPECIFICATIONS
 - INTERFACES SPECIFICATIONS
 - HOW SYSTEM OPERATES [OPERATING RULES]
 - PERFORMANCE REQUIREMENTS [ALLOCATION TO SUBSYSTEMS]
 - FUNCTIONAL MODELS OF SUBSYSTEMS
 - ANALYTICAL MODELS OF SUBSYSTEMS

INSTRUCTOR NOTES

THE RSL REPRESENTATION OF THE STRUCTURING FROM THE SYSTEM LEVEL DOCUMENTS TO THE RSL ELEMENTS IS AS FOLLOWS:

THE SOURCE DOCUMENTS CONTAIN THE ORIGINATING_REQUIREMENTS. IF THE ORIGINATING_REQUIREMENT IS TOO GENERAL TO TRACE DIRECTLY TO AN RSL ELEMENT, THE ORIGINATING_REQUIREMENT SHOULD BE BROKEN DOWN SO THAT IT INCORPORATES MORE (LOWER LEVEL) ORIGINATING_REQUIREMENTS WHICH CAN BE TRACED TO A PARTICULAR RSL ELEMENT. ORIGINATING_REQUIREMENTS ALSO IMPLEMENT A VERSION. THE FIRST BASELINE COULD BE VERSION_BASELINE-1. THEN AS MORE ORIGINATING_REQUIREMENTS ARE RECEIVED AND INCORPORATED, A NEW BASELINE COULD BE VERSION_BASELINE-2, AND SO ON. THIS WOULD PROVIDE A CONTINUING PICTURE FOR MANAGEMENT ON THE STATUS DEVELOPMENT OF THE SOFTWARE REQUIREMENTS.

METHODOLOGY STEPS

- PHASE 1 - DEFINITION OF SUBSYSTEM ELEMENTS
- PHASE 2 - EVALUATION OF THE KERNEL
- PHASE 3 - COMPLETION OF THE FUNCTIONAL DEFINITION
- PHASE 4 - COMPLETION OF MANAGEMENT AND CONTROL INFORMATION
- PHASE 5 - DYNAMIC FUNCTIONAL VALIDATION
- PHASE 6 - PERFORMANCE REQUIREMENTS
- PHASE 7 - ANALYTIC FEASIBILITY DEMONSTRATION

INSTRUCTOR NOTES

THERE ARE TWO DIFFERENT "STRUCTURAL" ELEMENTS TO BE DEFINED IN THE FIRST STAGE OF FUNCTIONAL SPECIFICATION. ONE IS THE FLOW CONNECTIVITY. THE OTHER, CONTAINED IN THE CURRENT METHODOLOGY (SREM), DEFINES THE REQUIRED DATA HIERARCHIES. BY ADDING THE DATA HIERARCHY TO THE STRUCTURES, A STEP-BY-STEP APPROACH TO TOP-LEVEL REQUIREMENTS DEVELOPMENT HAS BEEN IDENTIFIED IN WHICH ONLY THOSE AREAS REQUIRING ENGINEERING CREATIVITY ARE LEFT UNCONSTRAINED.

THE REQUIREMENTS ENGINEERS AND MANAGEMENT SHOULD ALWAYS KEEP IN MIND THAT THEIR TASK IS TO DEVELOP REQUIREMENTS FOR SOFTWARE AND NOT TO DESIGN THE SOFTWARE ITSELF. THE PERTINENT QUESTION IS TO CONSTANTLY ASK, "IS THIS A PRECISE STATEMENT OF WHAT THE DPS IS REQUIRED TO DO IN THE MOST GENERAL WAY SUCH THAT THE PROCESS DESIGNER HAS A MAXIMUM RANGE OF CHOICES IN DERIVING A SOLUTION?"

A TYPICAL SPECIFICATION PROCESS USUALLY STARTS WITH A FORMULATION OF THE PROCESSING STEPS INVOLVED AND LATER CONSIDERS THE DATA NEEDED TO SUPPORT THAT PROCESSING. SREM PARTIALLY REVERSES THIS ORDER OF CONSIDERATION; THE ORDER OF CONCERN IS: 1) "WHAT DATA ARE PRESENT TO THE DPS FOR PROCESSING?"; AND 2) "WHAT DATA ARE EXPECTED FROM THE DPS AS OUTPUT?". FROM THE ANSWERS TO THESE QUESTIONS, AN INITIAL CONCEPT OF THE REQUIRED PROCESSING STEPS CAN BE DERIVED. THE BASIC STEPS IN DETERMINING THE ANSWERS TO THESE QUESTIONS ARE LISTED ON THIS CHART.

PHASE 1: DEFINITION OF SUBSYSTEM ELEMENTS

- STEP 0 - ASSEMBLE DOCUMENTATION
- STEP 1 - REPRESENT DOCUMENTATION IN RSL
- STEP 2 - DEFINE SUBSYSTEMS AND INTERFACES
- STEP 3 - DEFINE MESSAGES AND CONTENTS
- STEP 4 - DEFINE INITIAL R_NETS
- STEP 5 - PRELIMINARY ANALYSES
- STEP 6 - DEFINE ENTITIES AND ASSOCIATED DATA
- STEP 7 - REFINE R_NETS

INSTRUCTOR NOTES

BEFORE WE CAN ADDRESS THE GENERATION OF SOFTWARE REQUIREMENTS, WE MUST FIRST ASSEMBLE ALL THE AVAILABLE DOCUMENTATION ABOUT A SYSTEM FROM WHICH SUCH REQUIREMENTS ARE TO BE DERIVED.

ALL OF THE INFORMATION LISTED BELOW IS TYPICALLY AVAILABLE WHEN SOFTWARE REQUIREMENTS GENERATION IS STARTED. ALL OF IT IS RELEVANT:

- SYSTEM DEFINITION - DEFINES WHAT THE SYSTEM DOES AND HOW WELL
- SUBSYSTEM AND INTERFACE SPECIFICATION - DEFINES THE PIECES OF THE SYSTEMS, THEIR RESPECTIVE PERFORMANCE REQUIREMENTS, AND HOW THEY COMMUNICATE WITH EACH OTHER
- OPERATING RULES - DEFINE HOW THE SUBSYSTEMS ARE TO WORK TOGETHER TO ACHIEVE THE SYSTEM GOALS
- PERFORMANCE REQUIREMENTS - CONTAIN ANY OF THE ABOVE
- ANALYTICAL AND/OR FUNCTIONAL MODELS - ARE SUPPLIED BY THE CUSTOMER OR DEVELOPED BY ENGINEERS FOR SIMULATION ANALYSIS

THIS INFORMATION MAY BE CONTAINED IN A VARIETY OF DOCUMENTS; ALL OF THESE SHOULD BE OBTAINED AND BASELINED. EACH TYPE OF INFORMATION LISTED HERE IS DISCUSSED IN THE NEXT FEW SLIDES.

STEP 0: ASSEMBLE DOCUMENTATION

- SOURCE DOCUMENTATION MAY INCLUDE
 - SYSTEM DEFINITION
 - SUBSYSTEMS SPECIFICATIONS
 - INTERFACES SPECIFICATIONS
 - HOW SYSTEM OPERATES [OPERATING RULES]
 - PERFORMANCE REQUIREMENTS [ALLOCATION TO SUBSYSTEMS]
 - FUNCTIONAL MODELS OF SUBSYSTEMS
 - ANALYTICAL MODELS OF SUBSYSTEMS

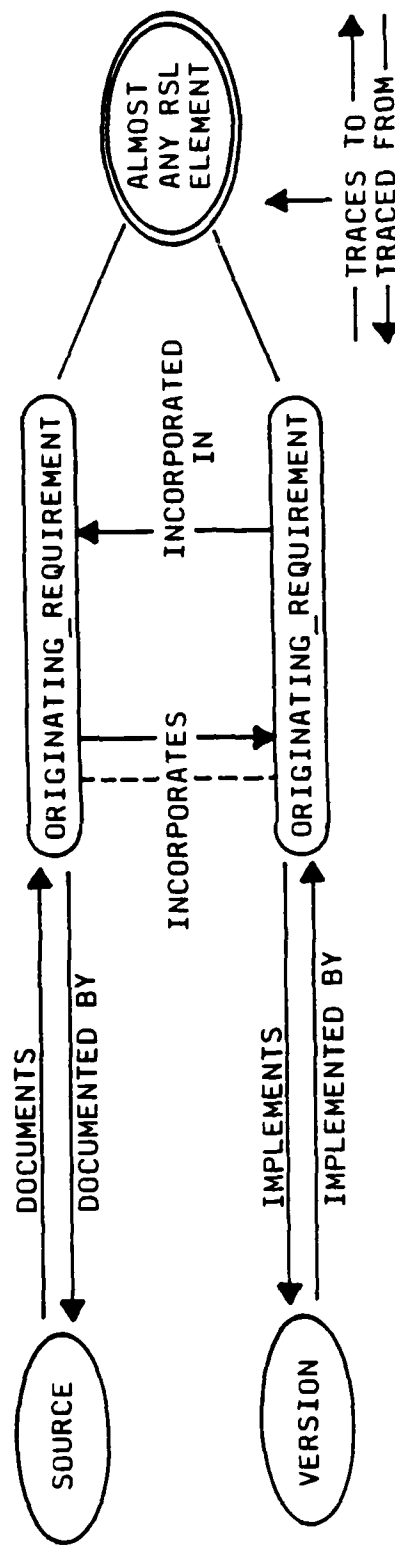
INSTRUCTOR NOTES

THE RSL REPRESENTATION OF THE STRUCTURING FROM THE SYSTEM LEVEL DOCUMENTS TO THE RSL ELEMENTS IS AS FOLLOWS:

THE SOURCE DOCUMENTS CONTAIN THE ORIGINATING_REQUIREMENTS. IF THE ORIGINATING_REQUIREMENT IS TOO GENERAL TO TRACE DIRECTLY TO AN RSL ELEMENT, THE ORIGINATING_REQUIREMENT SHOULD BE BROKEN DOWN SO THAT IT INCORPORATES MORE (LOWER LEVEL) ORIGINATING_REQUIREMENTS WHICH CAN BE TRACED TO A PARTICULAR RSL ELEMENT. ORIGINATING_REQUIREMENTS ALSO IMPLEMENT A VERSION. THE FIRST BASELINE COULD BE VERSION_BASELINE-1. THEN AS MORE ORIGINATING_REQUIREMENTS ARE RECEIVED AND INCORPORATED, A NEW BASELINE COULD BE VERSION_BASELINE-2, AND SO ON. THIS WOULD PROVIDE A CONTINUING PICTURE FOR MANAGEMENT ON THE STATUS DEVELOPMENT OF THE SOFTWARE REQUIREMENTS.

STEP 1: REPRESENT DOCUMENTATION

- TRANSLATE INTERFACE DEFINITIONS, SYSTEM FUNCTION AND SYSTEM OPERATING RULES TO RSL
- RSL CAPTURES DOCUMENTATION INFORMATION FOR TRACEABILITY ANALYSES LATER:



ATTRIBUTES
DESCRIPTION
ENTERED_BY

INSTRUCTOR NOTES

HERE'S AN EXAMPLE OF HOW "CONVENTIONAL" REQUIREMENTS ARE PUT INTO RSL FORM.

VG 778.1

7-111

STEP 1: REPRESENT DOCUMENTATION

"CONVENTIONAL"FORMAT

3.2.2.1 RADAR ORDERS BUFFER There shall be output interface from the data processing system called RADAR ORDERS BUFFER. The data processing system shall communicate through this interface with RADAR. Across this interface shall be passed RADAR ORDERS.

It is abbreviated by ROB. It was entered by Mike Richter.

3.2.2.2 RADAR ORDER. When transmitted across an interface, the software shall handle the message RADAR ORDER. This message is made up of RADAR COMMAND.

3.2.2.3 STARTUP. Information shall be maintained about STARTUP. This information shall include RO ORDER ID and STARTUP TIME.

3.2.2.4 SHUTDOWN. Information shall be maintained about SHUTDOWN. This information shall include RO ORDER ID.

3.2.2.5 TRANSMIT RECEIVE. Information shall be maintained about TRANSMIT RECEIVE. This information shall include;

RO ORDER ID
RO IMAGE ID
ALPHA PHASE TAPER
BETA PHASE TAPER
TRANSMIT INFORMATION
RECEIVE INFORMATION
NUMBER OF RANGE GATES
RANGE GATE INFORMATION

4.1.3.1 INITIALIZE TRACK ON IMAGE. Logical processing shall be done to INITIALIZE TRACK ON IMAGE. This shall have as input HANDOVER DATA. This shall have as output HOIQ, STATE DATA, and IMAGE ID. This logical processing shall, when appropriate, identify a new instance of IMAGE. This logical processing, when appropriate, shall identify the type of entity instance as being IMAGE IN TRACK.

NOTE: A request for pulses is made by entering a formal record into the HOIQ which feeds the pulse-sending procedures.

VG 778.1

7-11

RSL FORMAT

OUTPUT INTERFACE: RADAR_ORDERS_BUFFER.
ABBREVIATED BY: ROB.
CONNECTS TO: RADAR.
RECEIVES FROM: DPS.
PASSES: RADAR ORDER.
ENTERED BY: "MIKE RICHTER".

MESSAGE: RADAR ORDER.
MADE BY: RADAR_COMMAND.

DATA: STARTUP
INCLUDES: RO_ORDER_ID.
STARTUP_TIME.

DATA: SHUTDOWN
INCLUDES: RO_ORDER_ID.

DATA: TRANSMIT RECEIVE.
INCLUDES: RO_ORDER_ID
RO_IMAGE_ID
ALPHA PHASE TAPER
BETA PHASE TAPER
TRANSMIT INFORMATION
RECEIVE INFORMATION
NUMBER OF RANGE GATES
RANGE_GATE_INFORMATION

ALPHA: INITIALIZE STATE VAL DATA.
ARTIFICIALITY: VALIDATION
INPUTS: HANDOVER DATA, HANDOVER TIME.
OUTPUTS: UPDATE STATE, VALIDATION DATA.
DESCRIPTION: "THE REFERENCE ALGORITHM,
KALMAN FILTER, IS INITIALIZED.
HANDOVER DATA IS COPIED INTO
UPDATE STATE VALIDATION DATA
WITH PLACE IN TRACK TIME SET
TO HANDOVER_TIME."

ALPHA: INITIALIZE TRACK ON IMAGE.
INPUTS: HANDOVER DATA.
OUTPUTS: HOIQ, STATE_DATA, IMAGE_ID.
CREATES: IMAGE.
SETS: IMAGE IN TRACK.
DESCRIPTION: "A REQUEST FOR PULSE IS MADE
BY ENTERING A FORMAL RECORD
REQUEST INTO THE HOIQ, WHICH
FEEDS THE PULSE SENDING
PROCEDURES."

INSTRUCTOR NOTES

THE REQUIREMENTS STATEMENT LANGUAGE (RSL) IS A USER-ORIENTED MECHANISM FOR SPECIFYING REQUIREMENTS. IT IS ORIENTED HEAVILY TOWARD COLLOQUIAL ENGLISH. IT USES NOUNS FOR ELEMENTS AND ATTRIBUTES, TRANSITIVE VERBS FOR RELATIONSHIPS, AND A COMPLEMENTARY RELATIONSHIP USES THE PASSIVE FORM OF THE VERB. BOTH SYNTAX AND SEMANTICS ECHO ENGLISH USAGE, SO THAT MANY SIMPLE RSL SENTENCES MAY BE READ AS ENGLISH WITH THE SAME MEANING. HOWEVER, THE PRECISION OF RSL, ENFORCED THROUGH MACHINE TRANSLATION, IS NOT TYPICAL OF COLLOQUIAL SPEECH; AS A RESULT, MOST COMPLEX RSL SENTENCES ARE A SOMEWHAT STYLIZED FORM OF ENGLISH. THE BASIC COMPONENTS OF RSL ARE DESCRIBED AS FOLLOWS:

ELEMENTS. ELEMENTS IN RSL CORRESPOND ROUGHLY TO NOUNS IN ENGLISH. THEY ARE THOSE OBJECTS AND IDEAS WHICH THE REQUIREMENTS ANALYST USES AS BUILDING BLOCKS FOR HIS DESCRIPTION OF THE SYSTEM REQUIREMENTS. EACH ELEMENT HAS A UNIQUE NAME AND BELONGS TO ONE OF A NUMBER OF CLASSES CALLED ELEMENT TYPES. SOME EXAMPLES OF STANDARD ELEMENT TYPES IN RSL ARE ALPHA (THE CLASS OF FUNCTIONAL PROCESSING STEPS), DATA (THE CLASS OF CONCEPTUAL PIECES OF DATA NECESSARY IN THE SYSTEM), AND R_NET (THE CLASS OF PROCESSING FLOW SPECIFICATIONS).

ATTRIBUTES. ATTRIBUTES ARE MODIFIERS OF ELEMENTS SOMEWHAT IN THE MANNER OF ADJECTIVES IN ENGLISH; THEY FORMALIZE IMPORTANT PROPERTIES OF THE ELEMENTS. EACH ATTRIBUTE HAS ASSOCIATED WITH IT A SET OF VALUES WHICH MAY BE MNEMONIC NAMES, NUMBERS, OR TEXT STRINGS. EACH PARTICULAR ELEMENT MAY HAVE ONLY ONE OF THESE VALUES FOR ANY ATTRIBUTE. AN EXAMPLE OF AN ATTRIBUTE IS INITIAL VALUE, WHICH IS APPLICABLE TO ELEMENTS OF TYPE DATA. IT HAS VALUES WHICH SPECIFY WHAT THE INITIAL VALUE FOR THE DATA ITEM MUST BE IN THE IMPLEMENTED SOFTWARE AND FOR SIMULATIONS.

RELATIONSHIPS. THE RELATIONSHIP (OR RELATION) IN RSL MAY BE COMPARED WITH AN ENGLISH VERB. MORE PROPERLY, IT CORRESPONDS TO THE MATHEMATICAL DEFINITION OF A BINARY RELATION, A STATEMENT OF AN ASSOCIATION OF SOME TYPE BETWEEN TWO ELEMENTS. THE RSL RELATIONSHIP IS NON-COMMUTATIVE; IT HAS A SUBJECT ELEMENT AND AN OBJECT ELEMENT WHICH ARE DISTINCT. HOWEVER, THERE EXISTS A COMPLEMENTARY RELATIONSHIP FOR EACH SPECIFIED RELATIONSHIP WHICH IS THE CONVERSE OF THAT SPECIFIED RELATIONSHIP. ALPHA INPUTS DATA IS ONE OF THE RELATIONSHIPS IN RSL; THE COMPLEMENTARY RELATIONSHIP SAYS THAT DATA IS INPUT TO AN ALPHA.

STRUCTURES. THE FINAL RSL PRIMITIVE IS THE STRUCTURE, THE RSL REPRESENTATION OF THE FLOW GRAPH. TWO DISTINCT TYPES OF STRUCTURES HAVE BEEN IDENTIFIED. THE FIRST IS THE R NET (OR SUBNET) STRUCTURE. IT IDENTIFIES THE FLOW THROUGH THE FUNCTIONAL PROCESSING STEPS (ALPHAS) AND IS THUS USED TO SPECIFY THE SYSTEM RESPONSE TO VARIOUS STIMULI. THE SECOND STRUCTURE TYPE IS THE VALIDATION_PATH, WHICH IS USED TO SPECIFY PERFORMANCE OF THE SYSTEM.

RSL COMPONENTS

- ELEMENTS (NOUNS)
 - IDENTIFY CLASSES OF "THINGS" WITH NAMES
- ATTRIBUTES (ADJECTIVES)
 - PROPERTIES OF THE CLASSES OF "THINGS"
 - ATTACHED TO THE ELEMENTS
- RELATIONSHIPS (VERBS)
 - IDENTIFIES RELATIONS BETWEEN ELEMENT CLASSES
 - GENERAL FORM
 - <SUBJECT> RELATIONSHIP [OPTIONAL WORD] <OBJECT>
 - <OBJECT> COMPL RELATIONSHIP [OPTIONAL WORD] <SUBJECT>
- STRUCTURES (SEQUENCE)
 - PROVIDE MECHANISM FOR DEFINING SEQUENCES
 - R_NETS
 - SUBNETS
 - VALIDATION PATHS

{
 SOURCE
 ORIGINATING_REQUIREMENTS
 DATA
 ALPHA
 MESSAGE
 FILE
 }

{
 DESCRIPTION
 UNITS
 }

{
 INPUTS
 OUTPUTS
 PASSES
 MAKES
 }

INSTRUCTOR NOTES

THIS LIST IDENTIFIES THE POSSIBLE TYPES FOR EACH OF THE FOUR PRIMITIVE LANGUAGE CONCEPTS. THERE ARE 21 TYPES OF ELEMENTS, 23 RELATIONSHIPS, 21 ATTRIBUTES, AND 3 STRUCTURES. THIS IS NOT TO INFER THAT ALL TYPES OF ONE CONCEPT CAN BE USED FOR ANY OR ALL OF THE OTHER TYPES. THE LEGITIMATE CORRESPONDENCE CAN BE FOUND IN APPENDIX D OF THE USERS MANUAL.

THE "KERNEL" IS THE NUCLEUS OF INFORMATION CONSISTING OF THE REQUIREMENTS EXPRESSED IN RSL AND USING THE TYPES OF CONCEPTS LISTED ON THIS CHART.

RSL KEYWORDS

<u>21 ELEMENTS</u>	<u>23 RELATIONSHIPS</u>	<u>21 ATTRIBUTES</u>	<u>3 STRUCTURES</u>
ALPHA	ASSOCIATES	ALTERNATIVES	R_NETS
DATA	COMPOSES	ARTIFICIALITY	SUBNETS
DECISION	CONNECTS TO	BETA	VALIDATION_PATH
ENTITY_CLASS	CONSTRAINS	CHOICE	
ENTITY_TYPE	CONTAINS	COMPLETENESS	
EVENT	CREATES	DESCRIPTION	
FILE	DELAYS	ENTERED BY	
INPUT_INTERFACE	DESTROYS	GAMMA	
MESSAGE	DOCUMENTS	INITIAL_VALUE	
ORIGINATING_REQUIREMENT	ENABLES	LOCALITY	
OUTPUT_INTERFACE	EQUATES TO	MAXIMUM_TIME	
PERFORMANCE_REQUIREMENT	FORMS	MAXIMUM_VALUE	
R_NET	IMPLEMENTS	MINIMUM_TIME	
SOURCE	INCLUDES	MINIMUM_VALUE	
SUBNET	INCORPORATES	PROBLEM	
SUBSYSTEM	INPUTS	RANGE	
SYNONYM	MAKES	RESOLUTION	
UNSTRUCTURED_REQUIREMENT	ORDERS	TEST	
VALIDATION_PATH	OUTPUTS	TYPE	
VALIDATION_POINT	PASSES	UNITS	
VERSION	RECORDS	USE	
	SETS		
	TRACES TO		
	(PLUS THEIR COMPLEMENTS)		

INSTRUCTOR NOTES

HERE'S AN EXAMPLE OF HOW INTERFACES ARE DEFINED.

SREM IS HEAVILY ORIENTED TOWARD AN ORDERLY ANALYSIS OF THE INTERFACE DATA HIERARCHIES, IN A "TOP-DOWN" DIRECTION, AS THE FIRST STEP IN DEFINING DPS REQUIREMENTS. THIS IS A NATURAL DIRECTION, AS THE INTERFACES AND THE MESSAGES CROSSING THEM ARE USUALLY THE MOST CLEARLY DEFINED ELEMENTS OF THE ORIGINATING SPECIFICATIONS.

A SUBSYSTEM IS CONNECTED TO EITHER AN INPUT_INTERFACE OR AN OUTPUT_INTERFACE WHICH PASSES BLOCKS OF DATA CALLED MESSAGES. A MESSAGE IS MADE BY INDIVIDUAL DATA ITEMS, AND/OR BY FILES. IN TURN, A FILE CONTAINS INDIVIDUAL DATA ITEMS.

STEP 2: DEFINE SUBSYSTEMS AND INTERFACES

"CONVENTIONAL"FORMAT

3.2.2.1 RADAR ORDERS BUFFER There shall be output interface from the data processing system called RADAR ORDERS BUFFER. The data processing system shall communicate through this interface with RADAR. Across this interface shall be passed RADAR ORDERS.

It is abbreviated by ROB. It was entered by Mike Richter.

3.2.2.2 RADAR ORDER. When transmitted across an interface, the software shall handle the message RADAR ORDER. This message is made up of RADAR COMMAND.

3.2.2.3 STARTUP. Information shall be maintained about STARTUP. This information shall include RO ORDER ID and STARTUP TIME.

3.2.2.4 SHUTDOWN. Information shall be maintained about SHUTDOWN. This information shall include RO ORDER ID.

3.2.2.5 TRANSMIT RECEIVE. Information shall be maintained about TRANSMIT RECEIVE. This information shall include;

RO ORDER ID
RO IMAGE ID
ALPHA PHASE TAPER
BETA PHASE TAPER
TRANSMIT INFORMATION
RECEIVE INFORMATION
NUMBER OF RANGE GATES
RANGE GATE INFORMATION

4.1.3.1 INITIALIZE TRACK ON IMAGE. Logical processing shall be done to INITIATE TRACK ON IMAGE. This shall have as input HANDOVER DATA. This shall have as output HOIQ, STATE DATA, and IMAGE ID. This logical processing shall, when appropriate, identify a new instance of IMAGE. This logical processing, when appropriate, shall identify the type of entity instance as being IMAGE IN TRACK.

NOTE: A request for pulses is made by entering a formal record into the HOIQ which feeds the pulse-sending procedures.

VG 778.1

RSL FORMAT

OUTPUT INTERFACE: RADAR ORDERS BUFFER.
 ABBREVIATED BY: ROB.
 CONNECTS TO: RADAR.
 RECEIVES FROM: DPS.
 PASSES: RADAR ORDER.
 ENTERED BY: "MIKE RICHTER".

MESSAGE: RADAR ORDER.
 MADE BY: RADAR COMMAND.

DATA: STARTUP
 INCLUDES: RO ORDER ID.
STARTUP TIME.

DATA: SHUTDOWN
 INCLUDES: RO ORDER ID.

DATA: TRANSMIT RECEIVE.
 INCLUDES: RO ORDER ID
RO IMAGE ID
ALPHA PHASE TAPER
BETA PHASE TAPER
TRANSMIT INFORMATION
RECEIVE INFORMATION
NUMBER OF RANGE GATES
RANGE GATE INFORMATION

ALPHA: INITIATE STATE VAL DATA.
 ARTIFICIALITY: VALIDATION
 INPUTS: HANDOVER DATA, HANDOVER TIME.
 OUTPUTS: UPDATE STATE VALIDATION DATA.
 DESCRIPTION: "THE REFERENCE ALGORITHM, KALMAN FILTER, IS INITIALIZED. HANDOVER DATA IS COPIED INTO UPDATE STATE VALIDATION DATA WITH PLACE IN TRACK TIME SET TO HANDOVER TIME."

ALPHA: INITIATE TRACK ON IMAGE.
 INPUTS: HANDOVER DATA.
 OUTPUTS: HOIQ, STATE DATA, IMAGE ID.
 CREATES: IMAGE.
 SETS: IMAGE IN TRACK.
 DESCRIPTION: "A REQUEST FOR PULSE IS MADE BY ENTERING A FORMAL RECORD REQUEST INTO THE HOIQ, WHICH FEEDS THE PULSE SENDING PROCEDURES."

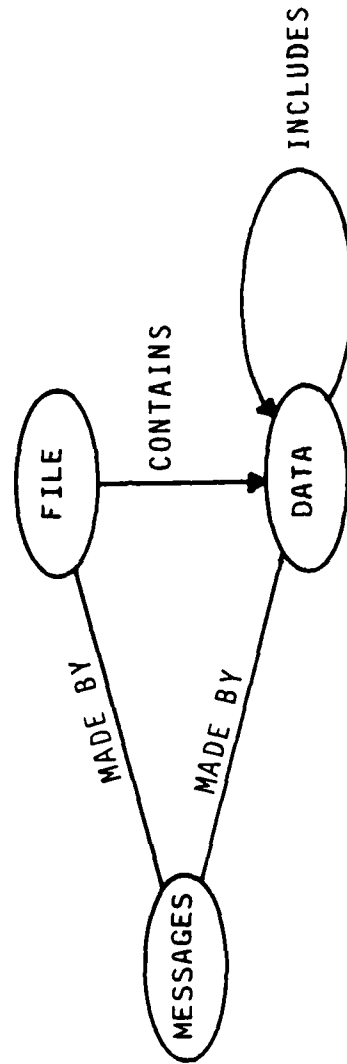
7-14

INSTRUCTOR NOTES

MESSAGES ARE THE AGGREGATION OF DATA AND FILES THAT ARE COMMUNICATED AS LOGICAL UNITS ACROSS INTERFACES. DATA AND FILES MAKE UP MESSAGES; A SINGLE DATA OR FILE MAY MAKE SEVERAL MESSAGES. A SINGLE INPUT_INTERFACE OR OUTPUT_INTERFACE MAY PASS SEVERAL MESSAGES TO OR FROM THE DATA PROCESSING SUBSYSTEM. A GIVEN MESSAGE MAY BE PASSED THROUGH ONLY ONE INTERFACE. THE DATA AND FILES THAT AN INPUT_INTERFACE OR OUTPUT_INTERFACE COMMUNICATES CAN BE ASCERTAINED BY REFERENCE TO THE DEFINITION OF ALL MESSAGES THAT PASS THROUGH THE INTERFACE. THESE RELATIONSHIPS, COMBINED WITH THE FILE AND DATA HIERARCHY RELATIONSHIPS, FORM AN ADDITIONAL HIERARCHY FOR ASSOCIATING INFORMATION.

STEP 3: DEFINE MESSAGES AND CONTENTS

• MESSAGE AND CONTENT RELATIONSHIPS



• RSL FORMAT

MESSAGE:

FULL_BEAM

MADE BY

DATA: SONIC_FREQUENCY

DATA: BANDSHIFT_FREQUENCY

DATA: BASELINE_COEFFICIENTS

DATA: DIGITAL_FILTER_SOURCE

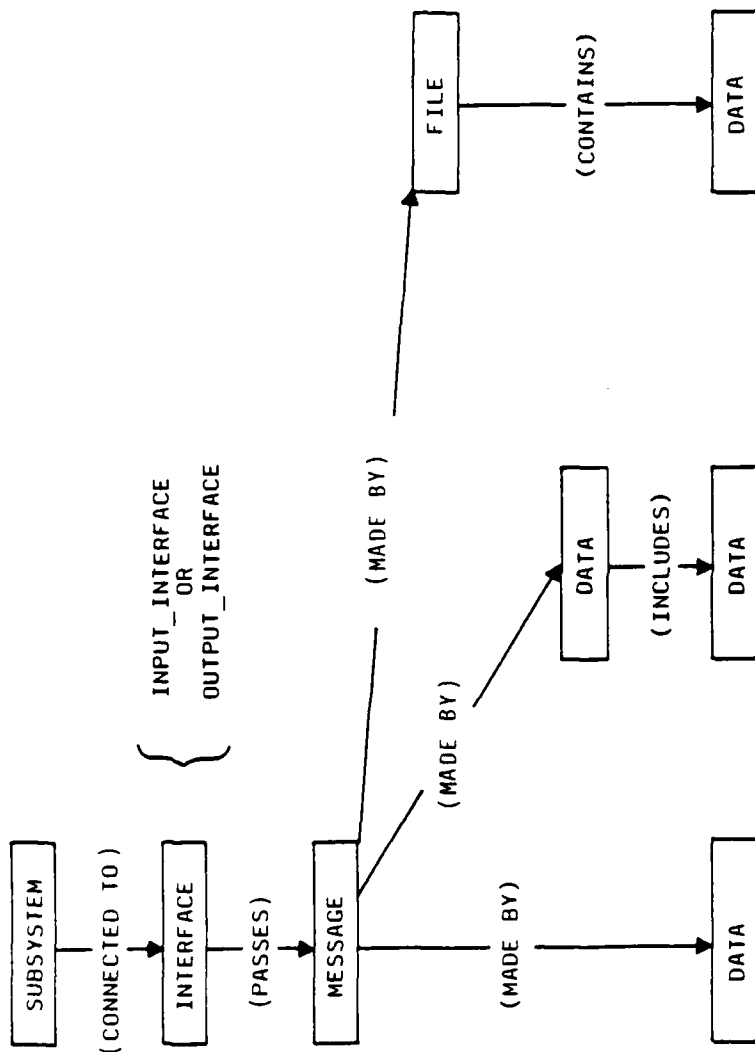
INSTRUCTOR NOTES

LET'S STEP BACK A MOMENT AND SUMMARIZE WHAT'S BEEN DONE SO FAR: A SUBSYSTEM IS CONNECTED TO AN INTERFACE. THE INTERFACE IS EITHER AN INPUT_INTERFACE OR AN OUTPUT_INTERFACE. THE INTERFACE PASSES MESSAGES. THE MESSAGES ARE MADE BY EITHER A FILE THAT CONTAINS DATA, DATA THAT INCLUDES OTHER DATA, AND/OR JUST BY DATA.

DURING STEPS 2 AND 3, PROGRESS CAN BE MEASURED BY THE NUMBER OF SUBSYSTEMS, INTERFACES, AND MESSAGES WHICH HAVE BEEN DEFINED IN RSL. IF INTERFACE SPECIFICATIONS EXIST WHICH CONTAIN THIS INFORMATION, THEN THE NUMBER OF EACH CAN BE DEFINED BEFORE THE JOB IS STARTED AND ABSOLUTE PROGRESS CAN BE MEASURED. IF SUCH DOCUMENTS DO NOT EXIST, THEN THE SYSTEM AND SUBSYSTEM DESIGNERS SHALL REVIEW THE RADX DOCUMENTATION FOR COMPLETENESS.

INTEGRATED VIEW OF STEP 2 AND 3

- IF INTERFACE SPECIFICATIONS EXIST ENCODE IT IN RSL
- IF NO INTERFACE SPECIFICATIONS EXIST CREATE RSL INTERFACE DEFINITIONS
- MUST BE REVIEWED BY SYSTEMS ENGINEERS FOR COMPLETENESS
- ELEMENT RELATIONSHIPS



INSTRUCTOR NOTES

IN STEP 4, ONE R_NET IS GENERATED FOR EACH INPUT_INTERFACE WITH APPROPRIATE SUBNETS SPECIFIED AS NEEDED. ADDITIONAL R_NETS ARE DEFINED TO REPRESENT PERIODIC OR TIME-DRIVEN PROCESSING. THESE R_NETS ARE THEN TRANSLATED TO RSL, INPUT TO THE RSL TRANSLATOR, LISTED, AND PLOTTED. THESE ARE CLEAR MEASURES FOR ASSESSING PROGRESS.

DURING STEP 4, AT LEAST ONE R_NET WILL BE CONSTRUCTED FOR EACH INPUT_INTERFACE, TO DESCRIBE THE PROCESSING OF ALL MESSAGES PASSED BY THAT INPUT_INTERFACE. MOREOVER, THERE WILL BE AT LEAST ONE DISTINCT PATH OF PROCESSING FOR EACH SUCH MESSAGE. THE R_NET PROGRESSES THROUGH THE DISTINCT PHASES OF BEING DRAWN, WRITTEN IN RSL, SUCCESSFULLY TRANSLATED BY REVS, LISTED AND PLOTTED BY RADX. STEP 4 IS NOT COMPLETE UNTIL ALL OF THE R_NETS ARE SUCCESSFULLY ENTERED INTO THE AUTOMATED DATABASE, AND REVIEWED WITH THOSE MOST FAMILIAR WITH THE SYSTEM. THIS IS EQUIVALENT TO THE EARLY ENGAGEMENT LOGIC PHASE OF STP.

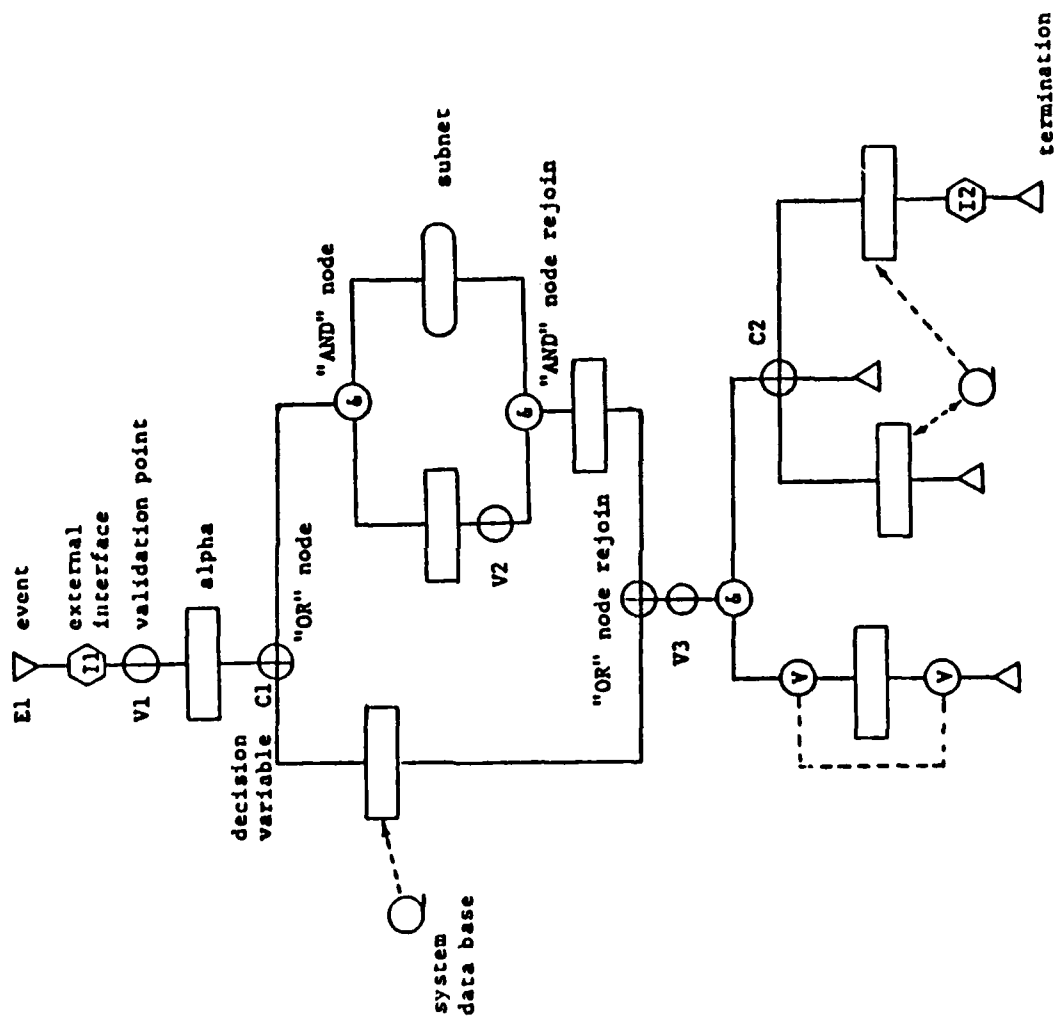
STEP 4: DEFINE INITIAL R-NETS

- AT LEAST ONE R_NET PER INPUT INTERFACE IS:
 - GENERATED BY HAND
 - TRANSLATED TO RSL
 - INPUT TO REVS
 - LISTED
 - PLOTTED
- THESE ARE REVIEWED BY OTHERS FOR COMPLETENESS OF FUNCTION

INSTRUCTOR NOTES

HERE'S THE STANDARD R-NET SYNTAX ...

R-NET NOTATION



INSTRUCTOR NOTES

DURING STEP 5, THE MESSAGE OUTPUT BY A BRANCH OF PROCESSING IS IDENTIFIED, AND WRITTEN IN RSL. RADX CAN THEN BE USED TO ASSURE THAT ALL PREVIOUSLY IDENTIFIED OUTPUT MESSAGES HAVE BEEN GENERATED. THIS IS A FIRST-LEVEL COMPLETENESS CHECK OF THE PROCESSING DEFINED TO DATE.

A SECOND KIND OF ANALYSIS PERFORMED AT THIS STEP ADDRESSES THE TRACEABILITY OF THE ORIGINATING REQUIREMENTS TO THE RSL REQUIREMENTS.

DURING STEP 5, ONE CAN MEASURE PROGRESS BY THE COMPLETION OF THE DEFINITION OF THE FORMS RELATIONSHIP BETWEEN THE ALPHAS ON THE R_NETS, AND THE MESSAGES PASSED BY THE OUTPUT_INTERFACES. WHEN ALL OF THE R_NETS HAVE BEEN SO ANALYZED, RADX IS THEN USED TO ASSURE THAT ALL OF THE OUTPUT MESSAGES PREVIOUSLY DEFINED IN THE INTERFACE DEFINITIONS HAVE BEEN FORMED. THUS THE TECHNICAL ANALYSIS OF RADX SERVES AS THE BASIS FOR MEASURING PROGRESS FOR MANAGEMENT AS WELL.

STEP 5: PRELIMINARY ANALYSES

- MESSAGE COMPLETENESS ANALYSIS
- TRACEABILITY ANALYSES
- REVS PRINTS OUTPUT MESSAGES NOT FORMED OR PASSED
- REVS PRINTS PROBLEM REPORTS NOT YET SOLVED

INSTRUCTOR NOTES

THIS SAMPLE RADX PRINTOUT ILLUSTRATES THE RADX COMMANDS REQUIRED TO LIST ALL MESSAGES NOT FORMED BY AN ALPHA AND AN EXAMPLE OF SUCH A MESSAGE DETECTED BY RADX. USING RADX FOR THIS PURPOSE ELIMINATES THE HUMAN ERROR IF DONE BY HAND, AND DECREASES THE AMOUNT OF TIME REQUIRED.

RADX PROCESSING OF THESE COMMANDS IDENTIFIES ERRORS

[RADX COMMAND =
SET OUT_MSG = MESSAGE THAT PASSED OUTPUT_INTERFACE,

SET COUNT = 3

[RADX COMMAND =
SET OUT_MSG_NOT_FORMED = OUT_MSG THAT IS NOT FORMED.

SET COUNT = 1
[RADX COMMAND =
LIST OUT_MSG_NOT_FORMED.

MESSAGE: REQUEST_NEXT_ENGINE.

MADE BY:

DATA: NEXT_CHANNEL_NO

DATA: NEXT_TIME,

PASSED THROUGH:

OUTPUT_INTERFACE: TO_MULTI,

[RADX COMMAND =
SET MSG_NOT_PASSED = MESSAGE THAT IS NOT PASSED.

SET COUNT = 0

[RADX COMMAND =
LIST MSG_NOT_PASSED.

[RADX COMMAND =
SET MULTI_PASSED_MESSAGE = MESSAGE THAT IS MULTIPLE PASSED.

SET COUNT = 0

INSTRUCTOR NOTES

IF ALTERNATIVES AND THE CHOICE ALTERNATIVE ARE NOT KNOWN AT THE TIME THE DECISION IS RECORDED, THE RECORD OF THE DECISION WITHOUT A CHOICE IS A PROBLEM REPORT. RADX CAN BE USED TO LIST THESE, THUS MAINTAINING A STATUS AND ASSURING THAT ALL DECISIONS ARE EVENTUALLY RESOLVED.

USE DECISIONS AS A PROBLEM REPORT AND TO RECORD DECISIONS

- AS R-NETS ARE DEFINED, YOU WILL DISCOVER THAT
 - DON'T KNOW HOW CERTAIN MESSAGES GET PROCESSES UNDER PARTICULAR CONDITIONS (E.G., ERROR PROCESSING)
 - DON'T KNOW CONDITIONS UNDER WHICH OUTPUT MESSAGES ARE TO BE PRODUCED (E.G., PERIODIC STATUS REPORTS)
 - MESSAGE CONTENTS NOT DEFINED, ETC.

- THESE CAN BE RECORDED IN DECISIONS WITHOUT CHOICE AND RADX CAN BE USED TO LIST THESE PERIODICALLY.

DECISION: XYZ.

PROBLEM: " _____ "

TRACES TO: MESSAGE _____
OUTPUT INTERFACE _____

R_NET _____
TRACES FROM: ORIGINATING_REQUIREMENT _____.

INSTRUCTOR NOTES

EXPERIENCE WITH THE ENGAGEMENT LOGIC IN THE SYSTEM TECHNOLOGY PROGRAM CONTRIBUTED TO THE REALIZATION THAT SOME CONCEPT WAS NEEDED TO REPRESENT "OBJECTS" IN THE OUTSIDE WORLD AND THE "PHASES" OF THAT OBJECT AS IT PASSED THROUGH THE SYSTEM. FOR EXAMPLE, AN OBJECT IN THE ENGAGEMENT LOGIC DURING THE "DETECTED" PHASE REQUIRED CERTAIN DATA, E.G., OBJECT ID, POSITION UPON DETECTION. AS THE OBJECT PASSED THROUGH DIFFERENT PHASES, E.G., VERIFIED, TRACK, ETC., DIFFERENT DATA WAS REQUIRED.

FOR EXAMPLE, OBJECTS IN A RADAR SYSTEM REQUIRE DIFFERENT TYPES OF DATA BASED ON THEIR STATUS. IF AN OBJECT IS IN THE DETECTED PHASE, THE SYSTEM RECOGNIZES IT AND SAVES ITS ID AND POSITION. LATER, WHEN THE OBJECT IS PLACED IN TRACK, THE SYSTEM NOT ONLY KNOWS ITS ID AND POSITION, BUT HAS COMPUTED ITS VELOCITY, RCS, BETA, AND IMPACT POINT.

THE EVALUATION CRITERIA FOR STEP 6 ARE THE UPDATING OF THE REQUIREMENTS DATABASE WITH THE ENTITY_CLASS AND ENTITY_TYPE INFORMATION, INCLUDING PARTIAL DEFINITION OF THE DATA AND FILES ASSOCIATED WITH THE ENTITIES.

STEP 6: DEFINE ENTITIES AND ASSOCIATED DATA

- ENTITY - A THING OR CATEGORY OF THINGS IN THE EXTERNAL WORLD, ABOUT WHICH THE SYSTEM MUST COLLECT, PROCESS AND MAINTAIN DATA
- MOST DATA IS ASSOCIATED WITH "OBJECTS" IN OUTSIDE WORLD
- OBJECTS MOVE THROUGH "PHASES" WHICH ASSOCIATE DIFFERENT DATA

OBJECT "PHASE"

DETECTED
VERIFIED
TRACK INITIATE
TRACK
DISCRIMINATION
INTERCEPT
DROPPED

ASSOCIATED DATA

ID, DETECTED -- POSITION
ID, POSITION/VELOCITY/RCS
ID, POSITION/VELOCITY/RCS/BETA
ABOVE + IMPACT POINT
ABOVE + DISCRIMINANTS
ABOVE + PROJECTED INTERCEPT POINT
POSITION/VELOCITY/BETA/RCS

ENTITIES AND DATA

"CONVENTIONAL"FORMAT

3.2.2.1 RADAR ORDERS BUFFER. There shall be output interface from the data processing system called RADAR ORDERS BUFFER. The data processing system shall communicate through this interface with RADAR. Across this interface shall be passed RADAR ORDERS.

It is abbreviated by ROB. It was entered by Mike Richter.

3.2.2.2 RADAR ORDER. When transmitted across an interface, the software shall handle the message RADAR ORDER. This message is made up of RADAR COMMAND.

3.2.2.3 STARTUP. Information shall be maintained about STARTUP. This information shall include RO ORDER ID and STARTUP TIME.

3.2.2.4 SHUTDOWN. Information shall be maintained about SHUTDOWN. This information shall include RO ORDER ID.

3.2.2.5 TRANSMIT RECEIVE. Information shall be maintained about TRANSMIT RECEIVE. This information shall include:

RO ORDER ID
RO IMAGE ID
ALPHA PHASE TAPER
BETA PHASE TAPER
TRANSMIT INFORMATION
RECEIVE INFORMATION
NUMBER OF RANGE GATES
RANGE GATE INFORMATION

4.1.3.1 INITIALIZE TRACK ON IMAGE. Logical processing shall be done to INITIATE TRACK ON IMAGE. This shall have as input HANDOVER DATA. This shall have as output H0IQ, STATE DATA, and IMAGE ID. This logical processing shall, when appropriate, identify a new instance of IMAGE. This logical processing, when appropriate, shall identify the type of entity instance as being IMAGE IN TRACK.

NOTE: A request for pulses is made by entering a formal record into the H0IQ which feeds the pulse-sending procedures.

VG 778.1

RSL FORMAT

OUTPUT INTERFACE: RADAR ORDERS_BUFFER.
ABBREVIATED BY: ROB.
CONNECTS TO: RADAR.
RECEIVES FROM: DPS.
PASSES: RADAR ORDER.
ENTERED BY: "MIKE RICHTER".

MESSAGE: RADAR ORDER.
MADE BY: RADAR_COMMAND.

DATA: STARTUP
INCLUDES: RO ORDER ID.
STARTUP TIME.

DATA: SHUTDOWN
INCLUDES: RO_ORDER_ID.

DATA: TRANSMIT RECEIVE.
INCLUDES: RO_ORDER_ID
RO_IMAGE_ID
ALPHA PHASE TAPER
BETA PHASE TAPER
TRANSMIT INFORMATION
RECEIVE INFORMATION
NUMBER OF RANGE GATES
RANGE_GATE_INFORMATION

ALPHA: INITIATE STATE VAL DATA.

ARTIFICIALITY: VALIDATION

INPUTS: HANDOVER DATA, HANDOVER TIME.

OUTPUTS: UPDATE STATE_VALIDATION DATA.

DESCRIPTION: "THE REFERENCE ALGORITHM,

KALMAN FILTER, IS INITIALIZED.

HANDOVER DATA IS COPIED INTO

UPDATE STATE VALIDATION DATA

WITH PLACE IN TRACK TIME SET

TO HANDOVER_TIME."

ALPHA: INITIATE TRACK ON IMAGE.

INPUTS: HANDOVER DATA.

OUTPUTS: H0IQ, STATE_DATA, IMAGE_ID.

CREATES: IMAGE.

SETS: IMAGE IN TRACK.

DESCRIPTION: "A REQUEST FOR PULSE IS MADE

BY ENTERING A FORMAL RECORD

REQUEST INTO THE H0IQ, WHICH

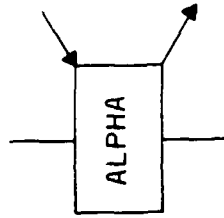
FEEDS THE PULSE SENDING

PROCEDURES."

7-23

INSTRUCTOR NOTES

THREE OF THE RELATIONSHIPS ASSOCIATED WITH AN ALPHA CONCERN THE HANDLING OF ENTITIES. THESE THREE ARE INCLUDED WITHIN THE DOTTED LINES ON THIS CHART. THESE RELATIONSHIPS ARE USED TO ESTABLISH THE OBJECT (SETS AND CREATES) AND REMOVES THE OBJECT FROM THE SYSTEM (DESTROYS). THE DATA PERTAINING TO THE OBJECT AND ITS STATE CAN BE UPDATED USING THE RELATIONSHIPS INPUTS AND OUTPUTS.



RSL -- ALPHA DEFINITION

- CAN BE USED TO CREATE, DESTROY OR SET ENTITIES
- SUPPORTS THE FOLLOWING

ATTRIBUTES

- DESCRIPTION
- ENTERED_BY
- COMPLETENESS
- ARTIFICIALITY

RELATIONSHIPS

- CREATES ENTITY_CLASS
- DESTROYS ENTITY_CLASS
- SETS ENTITY_TYPE
- INPUTS DATA, FILE
- OUTPUTS DATA, FILE
- FORMS MESSAGE
- DOCUMENTED BY SOURCE
- TRACED FROM DECISION
- EQUATED TO SYNONYM
- IMPLEMENTS VERSION

INSTRUCTOR NOTES

HERE'S HOW ALPHAS (PROCESSES THAT WORK ON ENTITIES) ARE REPRESENTED IN RSL.

"CONVENTIONAL" FORMAT

3.2.2.1 RADAR ORDERS BUFFER There shall be output interface from the data processing system called RADAR ORDERS BUFFER. The data processing system shall communicate through this interface with RADAR. Across this interface shall be passed RADAR ORDERS.

It is abbreviated by ROB. It was entered by Mike Richter.

3.2.2.2 RADAR ORDER. When transmitted across an interface, the software shall handle the message RADAR ORDER. This message is made up of RADAR COMMAND.

3.2.2.3 STARTUP. Information shall be maintained about STARTUP. This information shall include RO ORDER ID and STARTUP TIME.

3.2.2.4 SHUTDOWN. Information shall be maintained about SHUTDOWN. This information shall include RO ORDER ID.

3.2.2.5 TRANSMIT RECEIVE. Information shall be maintained about TRANSMIT RECEIVE. This information shall include;

RO ORDER ID
RO IMAGE ID
ALPHA PHASE TAPER
BETA PHASE TAPER
TRANSMIT INFORMATION
RECEIVE INFORMATION
NUMBER OF RANGE GATES
RANGE GATE INFORMATION

4.1.3.1 INITIALIZE TRACK ON IMAGE. Logical processing shall be done to INITIATE TRACK ON IMAGE. This shall have as input HANDOVER DATA. This shall have as output H0IQ, STATE DATA, and IMAGE ID. This logical processing shall, when appropriate, identify a new instance of IMAGE. This logical processing, when appropriate, shall identify the type of entity instance as being IMAGE IN TRACK.

NOTE: A request for pulses is made by entering a formal record into the H0IQ which feeds the pulse-sending procedures.

RSL FORMAT

OUTPUT INTERFACE: RADAR ORDERS_BUFFER.
 ABBREVIATED BY: ROB.
 CONNECTS TO: RADAR.
 RECEIVES FROM: DPS.
 PASSES: RADAR ORDER.
 ENTERED BY: "MIKE RICHTER".

MESSAGE: RADAR ORDER.
 MADE BY: RADAR_COMMAND.

DATA: STARTUP
 INCLUDES: RO ORDER ID.
STARTUP_TIME.

DATA: SHUTDOWN
 INCLUDES: RO_ORDER_ID.

DATA: TRANSMIT RECEIVE.
 INCLUDES: RO_ORDER_ID
RO_IMAGE_ID
ALPHA PHASE TAPER
BETA PHASE TAPER
TRANSMIT INFORMATION
RECEIVE INFORMATION
NUMBER OF RANGE GATES
RANGE_GATE_INFORMATION

ALPHA: INITIATE STATE VAL DATA.

ARTIFICIALITY: VALIDATION

INPUTS: HANDOVER DATA, HANDOVER TIME.

OUTPUTS: UPDATE STATE VALIDATION DATA.

DESCRIPTION: "THE REFERENCE ALGORITHM,

KALMAN FILTER, IS INITIALIZED.

HANDOVER DATA IS COPIED INTO

UPDATE STATE VALIDATION DATA

WITH PLACE IN TRACK TIME SET

TO HANDOVER_TIME."

ALPHA: INITIATE TRACK ON IMAGE.

INPUTS: HANDOVER DATA.

OUTPUTS: H0IQ, STATE_DATA, IMAGE_ID.

CREATES: IMAGE.

SETS: IMAGE IN TRACK.

DESCRIPTION: "A REQUEST FOR PULSE IS MADE

BY ENTERING A FORMAL RECORD

REQUEST INTO THE H0IQ, WHICH

FEEDS THE PULSE SENDING

PROCEDURES."

INSTRUCTOR NOTES

THE DETAILED REQUIREMENTS ALLOW THE R-NETS TO BE REFINED. INFORMATION LIKE:

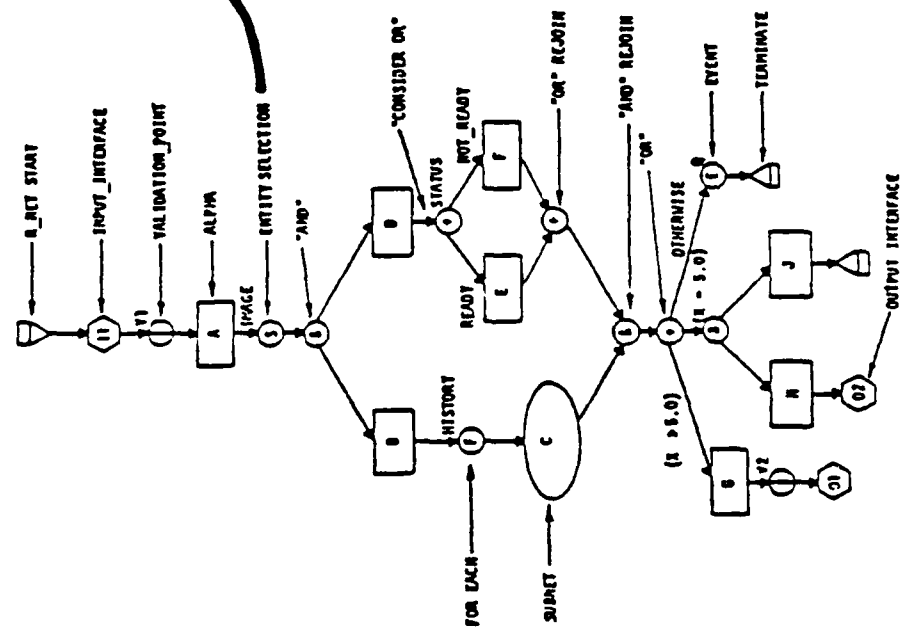
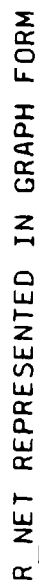
- CRITERIA FOR SELECTING AN ENTITY FOR PROCESSING (LIKE THE ONE SHOWN IN THIS SLIDE)

- CRITERIA FOR VALIDATING AN ENTITY

CAN BE ADDED TO THE R-NET PICTURES.

THE EVALUATION CRITERIA FOR THE COMPLETION OF STEP 7 ARE THE COMPLETION OF THE REVISION OF THE R_NETS TO REFLECT THE ENTITY INFORMATION, AND THE DEFINITION OF THE CREATES, DESTROYS, AND SETS RELATIONSHIP BETWEEN THE ALPHAS ON THE R_NETS, AND THE ENTITIES. PROGRESS CAN BE MEASURED BY THE NUMBER OF R_NETS AND THEIR ALPHAS SUCCESSFULLY REVISED.

STEP 7: REFINE R-NET



R NET REPRESENTED IN RSL

```

R NET: SAMPLE
STRUCTURE:
  INPUT_INTERFACE II
  VALIDATION_POINT VI
  ALPHA A
  SELECT ENTITY_CLASS IMAGE SUCH THAT (Y + Z)
DO
  ALPHA B
  FOR EACH FILE HISTORY RECORD
    DO SUBNET C END
  AND
  ALPHA D
  CONSIDER DATA STATUS
  IF (READY)
    ALPHA E
  OR (NOT READY)
    ALPHA F
  END
  IF (X > 5.0)
    ALPHA G
    VALIDATION_POINT V2
    OUTPUT_INTERFACE O1
  OR (X = 5.0)
    DO
      ALPHA M
      OUTPUT_INTERFACE O2
    AND
      ALPHA J
      TERMINATE
    OTHERWISE
      EVENT Q
      TERMINATE
    END
  END.

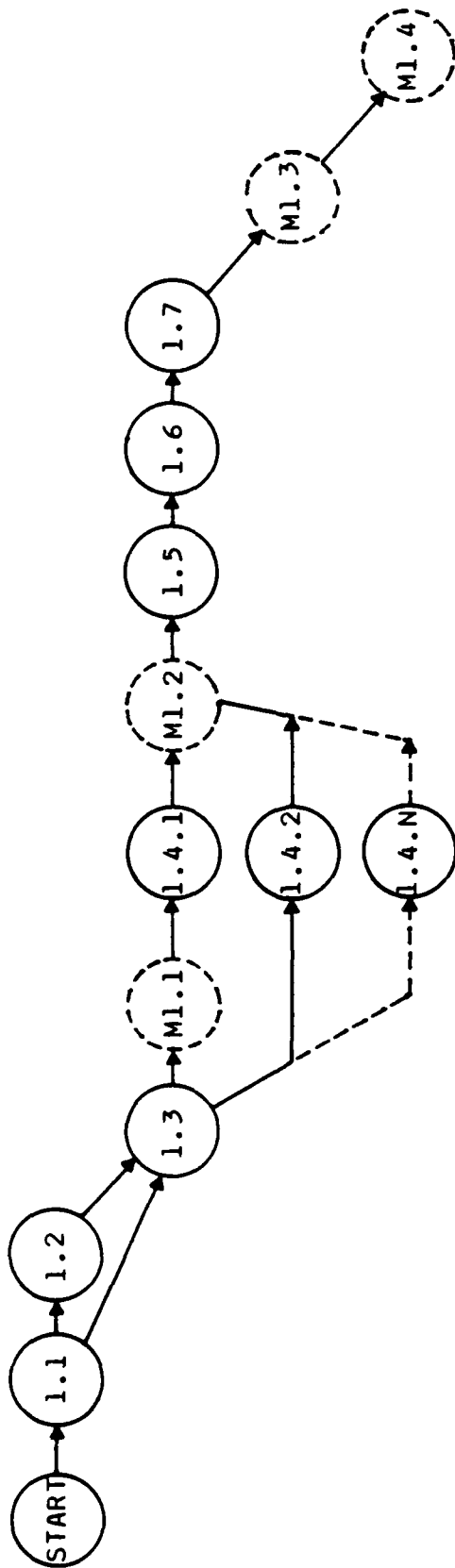
```


INSTRUCTOR NOTES

THE PREVIOUS SET OF TECHNICAL AND MANAGEMENT MILESTONES ARE PRESENTED HERE; FOR DIFFERENT APPLICATIONS, THE SET OF TECHNICAL MILESTONES COULD BE ADDED TO OR MODIFIED TO EMPHASIZE SPECIFIC PROGRAMMATIC OBJECTIVES. IT IS NOT NECESSARY FOR ONE STEP TO BE COMPLETELY FINISHED IN ALL RESPECTS BEFORE THE NEXT CAN BE UNDERTAKEN. IN MOST CASES, THE INTERFACES AND MESSAGES OF SOME SUBSYSTEMS OF THE SYSTEM MAY BE DEFINED BEFORE THOSE OF OTHERS; IN THIS CASE, WORK COULD BE INITIATED ON THE R_NET DEFINITION WHILE INTERFACES WERE BEING DEFINED FOR THE REMAINDER.

WE ALSO HAVE THE ADDITIONAL MANAGEMENT MILESTONES (HIGHLIGHTED). NOTE THAT THE INTERACTIONS WITH THE SYSTEMS ENGINEERS AND THE REQUIREMENTS ENGINEERING CONFIGURATION CONTROL BOARD BECOME THE FINAL AUTHORITY ON THE TECHNICAL CONTENT OF THE REQUIREMENTS DATABASE AS WORK PROGRESSES. SIMILAR DIAGRAMS COULD BE CONSTRUCTED FOR THE REMAINING PHASES, BUT WILL NOT BE PRESENTED HERE.

SAMPLE ACTIVITY NETWORK FOR SREM PHASE 1



MANAGEMENT MILESTONES

M1.1	M1.2	M1.3	M1.4
INTERFACE DEFINITION REVIEW WITH SYSTEMS ENGINEERING	DATABASE APPROVAL AT ALPHA LEVEL	DECISIONS APPROVED BY SRE CCB	SOFTWARE REQUIREMENT INTERNALLY BASELINED BY SRE CCB

TECHNICAL MILESTONES

START	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8
SOURCE SPECIFICA- TIONS BASELINED BY SYSTEMS ENGINEERING CCB	INTERFACE DEFINITIONS COMPLETED	MESSAGES DEFINED	INTERFACE DATA HIERARCHY SPECIFIED LEVEL	R NET #1 R NET #2, ETC., COMPLETE TO ALPHA DEFINED	ENTITIES AND ASSOCI- ATED DATA	FILE DEFINI- TIONS COMPLETED	ENGINEER- ING DE- CISIONS DOCUMENTED	R NETS COMPLETED (REFINED AS NECESSARY)

INSTRUCTOR NOTES

IN PHASE 1, THE PRESENTATION DEALT WITH HOW TO EXPRESS REQUIREMENTS IN RSL. IN STEP 7, THE PRESENTATION DESCRIBED HOW TO ANALYZE THE R_NETS AND ENTITIES USING RADX TO REFINE AND FILL IN THE MISSING INFORMATION. IN PHASE 2, RADX IS USED TO ANALYZE OTHER AREAS OF THE DATABASE CREATED THUS FAR.

PHASE 2 PERFORMS CONSOLIDATION AND EVALUATION OF THE RSL STATEMENTS PREPARED IN PHASE 1. AT THIS STAGE MANY OF THE EARLY MISTAKES AND INCONSISTENCIES HAVE BEEN DETECTED BY THE RSL TRANSLATOR DURING THE ENTRY PROCESS AND THIS NUCLEUS OR KERNEL HAS BEEN ENTERED INTO THE ABSTRACT SYSTEMS SEMANTIC MODEL (ASSM) TO CREATE A DATABASE. THIS KERNEL HAS BEEN CHECKED MANUALLY AND WITH THE AID OF THE TRANSLATOR. THE MAJOR PURPOSE OF PHASE 2 IS TO EVALUATE THE DATABASE USING RADX. THUS THE EVALUATION OF PHASE 2 IS AN ITERATIVE PROCESS OF MODIFYING THE DATABASE AND THEN USING RADX AGAIN TO CHECK FOR COMPLETENESS AND CONSISTENCY.

PHASE 2: EVALUATION OF THE KERNEL

- PHASE 2 IS AN EVALUATION OF PHASE 1 RESULTS
 - USE RADX TO IDENTIFY:
 - INCOMPLETENESS
 - INCONSISTENCIES
 - MODIFY DATABASE AND ITERATE
 - RADX OUTPUT CAN BE USED TO DETERMINE PHASE 3 IS COMPLETE
- PHASE 2 IS COMPLETE WHEN
 - RADX ANALYSES YIELD NO ERRORS NOT ADDRESSED BY PROBLEM REPORTS
 - PROBLEM REPORTS ARE GIVEN DUE DATES FOR RESOLUTION
 - DECISIONS ARE REVIEWED AND BASELINED

PHASE 2

- TO PERFORM THE DATA ANALYSIS, USE RADX TO OBTAIN THE FOLLOWING ...
 - FREE-STANDING DATA AND FILES, TO DETERMINE IF THEY SHOULD BE PART OF OTHER HIERARCHIES, OR EXTRANEOUS
 - EXTRANEOUS ALPHAS, INTERFACES, SUBNETS
 - ENTITIES AND THEIR HIERARCHIES TO DETERMINE IF ALL OUTSIDE "OBJECTS" OR "THINGS" AND THEIR PHASES ARE REPRESENTED
 - INPUT_INTERFACES AND THEIR HIERARCHIES TO DETERMINE IF ALL INPUTS TO THE COMPUTER ARE COMPLETE
 - OUTPUT_INTERFACES AND THEIR HIERARCHIES TO DETERMINE IF ALL OUTPUTS FROM THE COMPUTER ARE DONE
 - DOCUMENTATIONS

INSTRUCTOR NOTES

IN ORDER TO PERFORM THE ANALYSES AND OBTAIN DESIRED LISTINGS USING RADX, THE APPROPRIATE COMMANDS MUST BE INPUT. THE DEFINE SET FACILITY OF RADX PROVIDES AN ASSM QUERY AND INTERROGATION CAPABILITY AND PROVIDES A TECHNIQUE FOR IDENTIFYING SUBSETS OF INFORMATION TO BE PROCESSED BY OTHER RADX COMMANDS. THE INFORMATION IDENTIFIED BY THE SET COMMANDS WILL BE PRINTED BASED ON THE LIST COMMANDS. THUS, THERE ARE TWO BASIC CAPABILITIES:

- 1) IDENTIFICATION OF DESIRED INFORMATION USING SET COMMANDS, AND 2) PRINTOUT OF THIS INFORMATION USING LIST COMMANDS.

THERE ARE TWO BASIC TYPES OF THESE RADX COMMANDS:

- PRESPECIFIED
- USER SPECIFIED

THIS CHART LISTS SOME OF THE PRESPECIFIED SET COMMANDS AVAILABLE. A MORE DETAILED LIST CAN BE FOUND IN THE REVS USERS MANUAL, PAGES 6-17 THROUGH 6-23. A DECK OF THESE PRESPECIFIED COMMANDS IS ALSO AVAILABLE TO THE USER.

EXAMPLES OF PRESPECIFIED RADX COMMANDS FOR REQUIREMENTS ANALYSIS

COMMANDS TO TEST STRUCTURE SPECIFICATIONS (CONTINUED)

```

SET MULTI_CONTAINED = DATA THAT IS MULTIPLE CONTAINED
                        (*A DATA ITEM CAN ONLY BE CONTAINED IN ONE FILE.*).
SET INFO = DATA OR FILE.
SET MULTI_ASSOCIATES_CLASS = INFO WHICH IS MULTIPLE ASSOCIATED WITH ENTITY CLASS
                        (*A SINGLE DATA OR FILE CAN ONLY BE ASSOCIATED WITH ONE
                        ENTITY CLASS.*).
HIER ENTITY_TYPE_TRACE = ENTITY_TYPE ASSOCIATES FILE,
                        ENTITY_TYPE ASSOCIATES DATA,
                        DATA INCLUDES DATA.
HIER ENTITY_CLASS_TRACE = ENTITY_CLASS ASSOCIATES FILE,
                        ENTITY_CLASS ASSOCIATES DATA,
                        DATA INCLUDES DATA.
HIER MESSAGE_TRACE = MESSAGE MADE BY FILE,
                        MESSAGE MADE BY DATA,
                        DATA INCLUDES DATA.
HIER FILE_TRACE = FILE CONTAINS DATA,
                        DATA INCLUDES DATA.
SET TYPE_INFO = ALL IN HIER ENTITY TYPE TRACE.
SET TYPE_INFO = TYPE_INFO MINUS ENTITY_TYPE.
SET CLASS_INFO = CLASS_INFO MINUS ENTITY_CLASS.
SET FILE_INFO = ALL IN HIER FILE TRACE.
SET TYPE_AND_CLASS_INFO = TYPE_INFO AND CLASS_INFO
                        (*A DATA OR FILE CANNOT BE ASSOCIATED WITH BOTH AN
                        ENTITY_CLASS AND AN ENTITY_TYPE.*).
SET ENTITY_INFO = TYPE_INFO OR CLASS_INFO.
SET ENTITY_AND_FILE_INFO = ENTITY_INFO AND FILE_INFO
                        (*DATA CANNOT BE BOTH CONTAINED IN A FILE AND ASSOCIATED
                        WITH AN ENTITY.*).
SET STRUCTURE_ELEMENTS = R_NET, SUBNET, VALIDATION_PATH.
SET MISSING_STRUCTURE = STRUCTURE_ELEMENTS WITHOUT REFERS
                        (*THE REQUIREMENTS ARE NOT COMPLETE UNTIL ALL R_NET, SUBNET,
                        AND VALIDATION_PATH ELEMENTS HAVE BEEN GIVEN A STRUCTURE.*).
SET NON_ENABLING_EVENT = EVENT WITHOUT ENABLES
                        (*AN EVENT MUST ENABLE AT LEAST ONE R_NET.*)
SET COMPLEX_DATA = DATA THAT INCLUDES DATA.
SET BAD_DELAYED_EVENT = EVENT THAT IS DELAYED BY COMPLEX DATA
                        (*AN EVENT CAN ONLY BE DELAYED BY LOWEST LEVEL DATA.*)
SET NON_ENABLING_INPUT_INF = INPUT_INTERFACE WITHOUT ENABLES
                        (*AN INPUT_INTERFACE MUST ENABLE AN R_NET.*)
SET MULTI_DELAYED = EVENT THAT IS MULTIPLE DELAYED
                        (*AN EVENT CANNOT BE DELAYED BY MORE THAN ONE DATA_ELEMENT.*).

```


INSTRUCTOR NOTES

THIS IS AN EXAMPLE OF A USER SPECIFIED COMMAND. IT DEFINES A SET (SET X) THAT IS A COMBINATION OF TWO SETS (SET A AND SET B) BUT SUBTRACTING WHAT IS COMMON BETWEEN THEM.

FIRST, DEFINE A SET EQUAL TO THE INTERSECTION.

NEXT, DEFINE A SET Y EQUAL TO EVERYTHING IN BOTH SETS.

FINALLY, SUBTRACT THE INTERSECTION (Z) FROM THE COMPLETE SET (Y), LEAVING THE DESIRED SET X.

THE EXAMPLES ON THIS CHART ARE USER SPECIFIED SETS. THESE SETS AND LIST COMMANDS CAUSE RADX TO IDENTIFY AND PRINT:

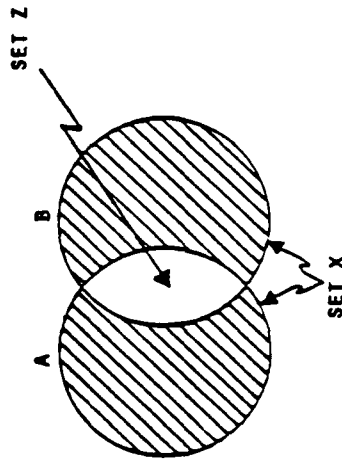
- INTERFACES WHICH DO NOT PASS A MESSAGE
- MESSAGES WHICH ARE NOT FORMED BY AN ALPHA
- MESSAGES WHICH ARE NOT PASSED BY AN INPUT_INTERFACE OR AN OUTPUT_INTERFACE

RADX USES SET MANIPULATION FACILITIES TO PERFORM THESE ANALYSES

SET Z = SET A AND SET B

SET Y = SET A OR SET B

SET X = SET Y MINUS SET Z



RADX REPORT

[RADX COMMAND =
LIST INTERFACE_NO_MESSAGE.

[RADX COMMAND =
SET OUT_MSG = MESSAGE THAT PASSED OUTPUT_INTERFACE,

SET COUNT = 3

[RADX COMMAND =
SET OUT_MSG_NOT_FORMED = OUT_MSG THAT IS NOT FORMED.

MESSAGE: REQUEST_NEXT_ENGINE.
MADE BY:

DATA: NEXT_CHANNEL_NO

DATA: NEXT_TIME,

PASSED THROUGH:

OUTPUT_INTERFACE: TO_MULTI,

[RADX COMMAND =
SET MSG_NOT_PASSED = MESSAGE THAT IS NOT PASSED.

INSTRUCTOR NOTES

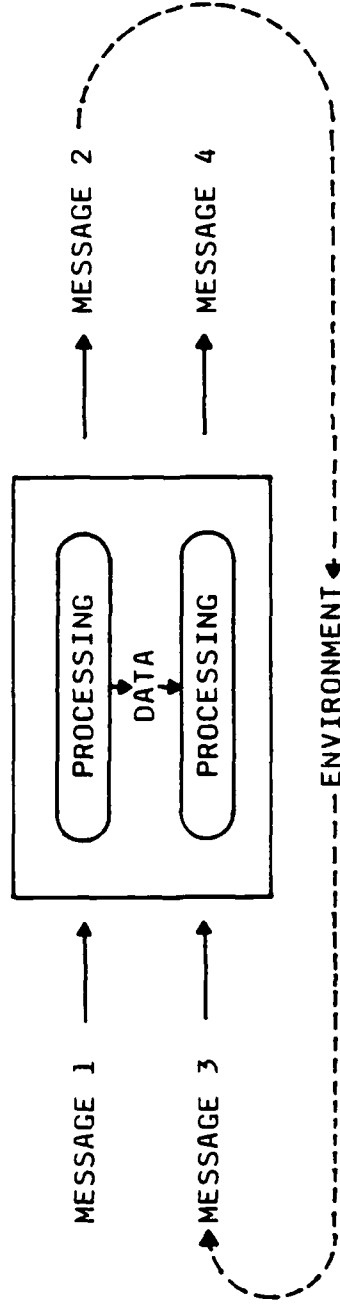
THE WORK THROUGH PHASE 2 PROVIDED DEFINITION OF THE HIGHER-LEVEL ELEMENTS OF DATA HIERARCHIES: DEFINITION OF R_NETS, THEIR STRUCTURES, AND THEIR ENABLING EVENTS, AND DECLARATION OF THE EXISTENCE OF ALPHAS AND THEIR PLACE IN THE R_NETS. THIS INFORMATION HAS BEEN ENTERED INTO THE ASSM AND HAS BEEN SUBJECTED TO VARIOUS FORMS OF AUTOMATED ANALYSIS AND REVIEW.

IT NOW REMAINS TO COMPLETE THE DEFINITION OF THESE ELEMENTS TO PROVIDE A BASIS FOR CONSTRUCTION OF AN EXECUTABLE SIMULATION. EACH ALPHA MUST BE DEFINED IN TERMS OF ITS DATA TRANSACTIONS, AND DEFINITION OF ALL DATA WHICH ARE KNOWN TO BE INPUT TO OR OUTPUT FROM AN ALPHA MUST BE COMPLETED. IN GENERAL, THE DATA WHICH CAN BE DESCRIBED AT THIS STAGE ARE EITHER THOSE GLOBAL TO MULTIPLE R_NETS, OR THOSE LOCAL DATA WHICH MADE MESSAGES. THE NEED FOR ADDITIONAL LOCAL DATA INTERNAL TO THE R_NETS WILL BE DISCOVERED IN THE DEVELOPMENT OF EXECUTABLE DESCRIPTIONS. AT THAT TIME THE NEED FOR DEFINITION OF LOWER LEVELS OF THE EXISTING DATA HIERARCHIES MAY BE APPARENT.

DURING THIS PROCESS, THE ORIGINAL ALPHAS WILL OFTEN NEED REDEFINITION. ADDITIONAL ALPHAS CAN BE ADDED, OR THE ORIGINAL ALPHAS CAN BE REDEFINED AS SUBNETS, WHICH PRESERVES TRACEABILITY AND MINIMIZES MODIFICATION OF THE R_NETS. OCCASIONALLY, THE STRUCTURE OF THE NET MAY CHANGE, OR NEW NETS MAY BE ADDED, AS THE PROCESSING REQUIREMENTS EVOLVE.

PHASE 3 IS THE FINAL DATA DEFINITION AND ANALYSIS. AS SHOWN ON THIS CHART, DATA GENERATED IN PROCESSING MESSAGE 1 AND OUTPUTTING MESSAGE 2 IS TO BE USED BY A LATER STEP WHICH PROCESSES MESSAGE 3 AND OUTPUTS MESSAGE 4. THIS DATA SHOULD BE ANALYZED TO DETERMINE HOW THIS DATA IS GENERATED, SAVED, AND USED.

PHASE 3: COMPLETION OF THE FUNCTIONAL DEFINITION



- PHASE 3 STEPS
 - ANALYZE THE DATA DEFINITION AND ITS USE
 - COMPLETE DATA DEFINITION WHERE NEEDED
 - EVALUATE USING RADX
- NEED TO DESCRIBE
 - WHAT INTERNAL DATA -- HOW DERIVED
 - ASSOCIATED WITH SPECIFIC "ENTITY"
 - HOW DATA IS USED FOR FURTHER PROCESSING

INSTRUCTOR NOTES

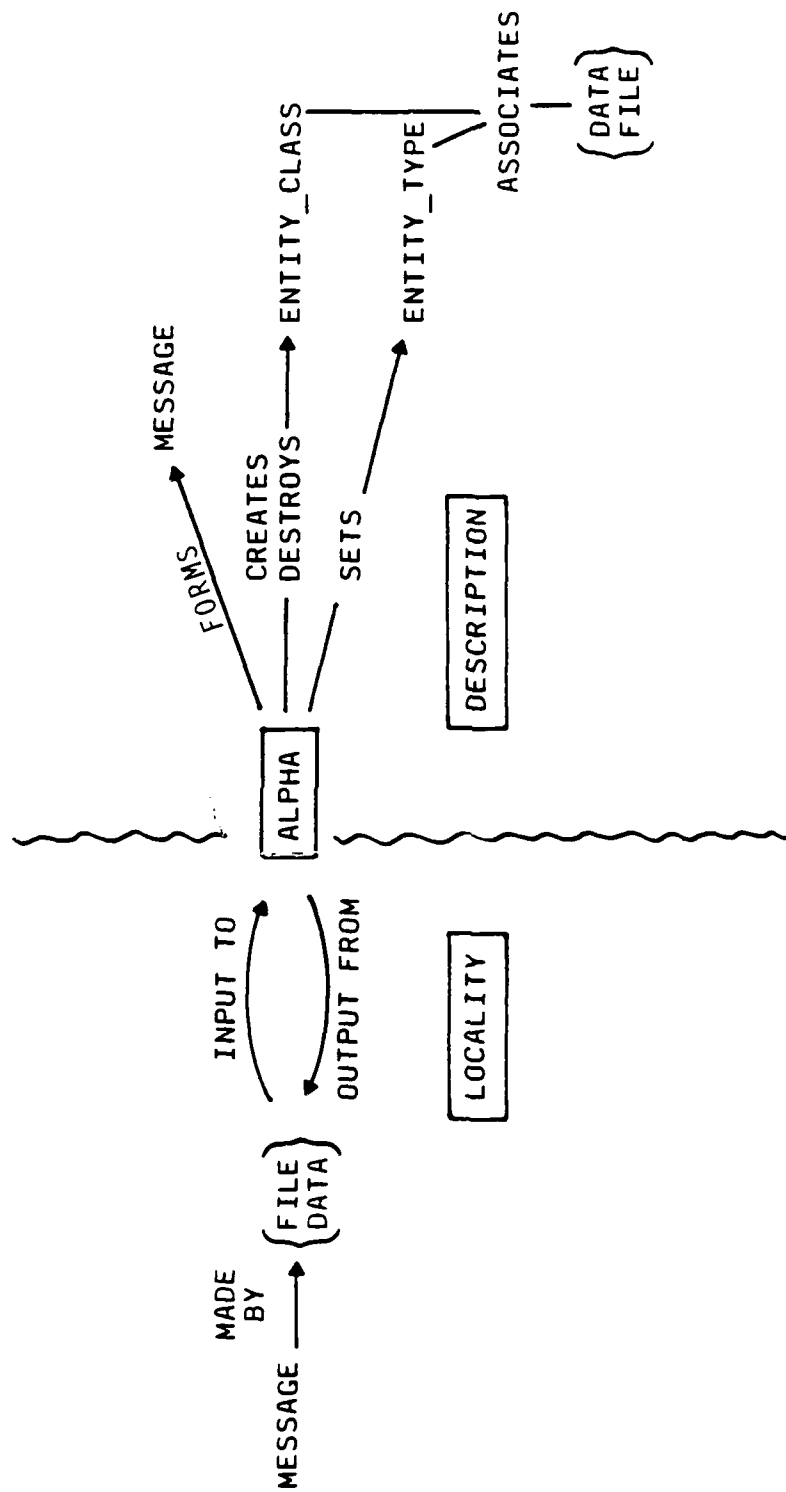
PHASE 3 OCCURS IN TWO PARTS: THE DATA ARE FIRST DEFINED, AND THEN RADX IS USED TO ANALYZE IT.

PROGRESS DURING THE FIRST PART OF PHASE 3 CAN BE MEASURED BY THE NUMBER OF R_NETS FOR WHICH THE DATA OF ALL TYPES HAS BEEN DEFINED IN RSL. SINCE THE R_NETS AND THEIR ALPHAS HAVE BEEN PREVIOUSLY DEFINED, ONE CAN HAVE AN ALMOST ABSOLUTE MEASURE OF SUCH PROGRESS. THIS MEASURE IS NOT ABSOLUTE BECAUSE, IN ATTEMPTING TO DEFINE THE INPUT AND OUTPUT RELATIONSHIPS OF THE ALPHAS, INCONSISTENCIES AND AMBIGUITIES WILL BE SURFACED WHICH MAY REQUIRE REVISION OF THE R_NETS, THE ADDITION AND MODIFICATION OF THE ALPHAS, THE REVISION OF THE ENTITIES, AND SO FORTH.

DURING THE SECOND HALF OF PHASE 3, THE TECHNICAL AND MANAGEMENT CRITERIA AGAIN OVERLAP. PHASE 3 IS COMPLETE WHEN RADX DETECTS NO MORE ERRORS OF THE KINDS IDENTIFIED HERE. PROGRESS CAN BE MEASURED BY THE RATE AT WHICH THESE ERRORS ARE CORRECTED.

FURTHER DATA ANALYSIS CAN BE PERFORMED BY CONSIDERING HOW DATA FITS INTO THIS DIAGRAM. A MESSAGE IS MADE BY FILES AND/OR DATA. FILES AND/OR DATA CAN BE INPUT TO AND OUTPUT FROM ALPHAS BUT MESSAGES CANNOT. AN ALPHA CAN "FORM" A MESSAGE, CREATE OR DESTROY AN ENTITY_CLASS, AND/OR SET AN ENTITY_TYPE.

THE DATA RELATIONSHIPS



INSTRUCTOR NOTES

TRACEABILITY IS A FEATURE OF THE SPECIFICATION SUPPORTING ITS MANAGEMENT, RATHER THAN ONE REQUIRED FOR ITS TECHNICAL QUALITY PER SE. THE REQUIREMENTS FOR TRACEABILITY ARE ALSO ENTERED INTO THE ASSM.

THE ORIGINATING REQUIREMENTS FOR A SOFTWARE SPECIFICATION ARE MOST COMMONLY CONTAINED IN HIGHER-LEVEL SPECIFICATIONS. IN GENERAL (DEPENDING ON MANAGEMENT DECISION) EACH IDENTIFIABLE SOFTWARE REQUIREMENT IN EACH HIGHER-LEVEL SPECIFICATION WILL BE CALLED OUT AS AN ORIGINATING REQUIREMENT IN THE ASSM. THE DESCRIPTION ATTRIBUTED TO AN ORIGINATING REQUIREMENT MAY BE A LITERAL EXCERPT OF THE SOURCE, OR MAY BE AN INTERPRETATION, AGAIN AT MANAGEMENT DISCRETION.

THERE MAY BE SOURCES OF INFORMATION WHICH DEFINE SPECIFICS OF THE SOFTWARE REQUIREMENTS, BUT WHICH ARE NOT SPECIFICATIONS IN THEMSELVES. FOR EXAMPLE, A BOOK DEFINING A STANDARD ATMOSPHERE MODEL MAY BE REFERENCED IN THE SOURCE SPECIFICATIONS. NOTE THAT A SOURCE MAY BE CHANGED DURING DEVELOPMENT OF EITHER THE SPECIFICATION OR THE SOFTWARE; WHEN SUCH A CHANGE OCCURS, ITS IMPACT ON THE SPECIFICATION MAY BE FOLLOWED THROUGH USE OF THE DOCUMENT'S RELATIONSHIP.

THE PROCESS OF DEVELOPING THE DPS REQUIREMENTS MAY EXPOSE SYSTEM ISSUES WHICH CANNOT BE RESOLVED IMMEDIATELY IN A FORMAL MANNER, BUT WHICH MAY HAVE SIGNIFICANT IMPACT ON THE DIRECTION OF FURTHER WORK. EACH SUCH ISSUE IS RECORDED IN THE ASSM AS A DECISION WHICH IS TRACED FROM THE ORIGINATING REQUIREMENT(S), EITHER DIRECTLY OR THROUGH OTHER DECISIONS. EACH DECISION INCLUDES A STATEMENT OF THE PROBLEM. AT SOME TIME THE RECOGNIZED ALTERNATIVES SHOULD BE ADDED AND WHEN THE ALTERNATIVE TO BE FOLLOWED IS SELECTED, IT SHOULD BE INDICATED USING CHOICE.

A SYNONYM MAY BE DECLARED AND EQUATED TO ANY OTHER ELEMENT FOR CLARITY AND CONVENIENCE. THERE IS AN ATTRIBUTE ENTERED BY WHICH PERMITS THE AUTHOR OF INFORMATION TO RECORD HIS NAME, THUS PROVIDING A LOG OF CHANGES.

PHASE 4: COMPLETION OF MANAGEMENT AND CONTROL INFORMATION

- COMPLETE AND ANALYZE TRACEABILITY INFORMATION
- TRACEABILITY INFORMATION
 - SOURCE
 - ORIGINATOR_REQUIREMENTS
 - DECISIONS
 - DESCRIPTIONS
 - AUTHORSHIP
- OBJECTIVE IS TO IDENTIFY AND EVALUATE THAT
 - ALL ORIGINATING_REQUIREMENTS (FUNCTIONAL) REPRESENTED IN RSL
 - ALL ORIGINATING_REQUIREMENTS ARE TRACED TO RSL ELEMENTS WHICH SATISFY THEM

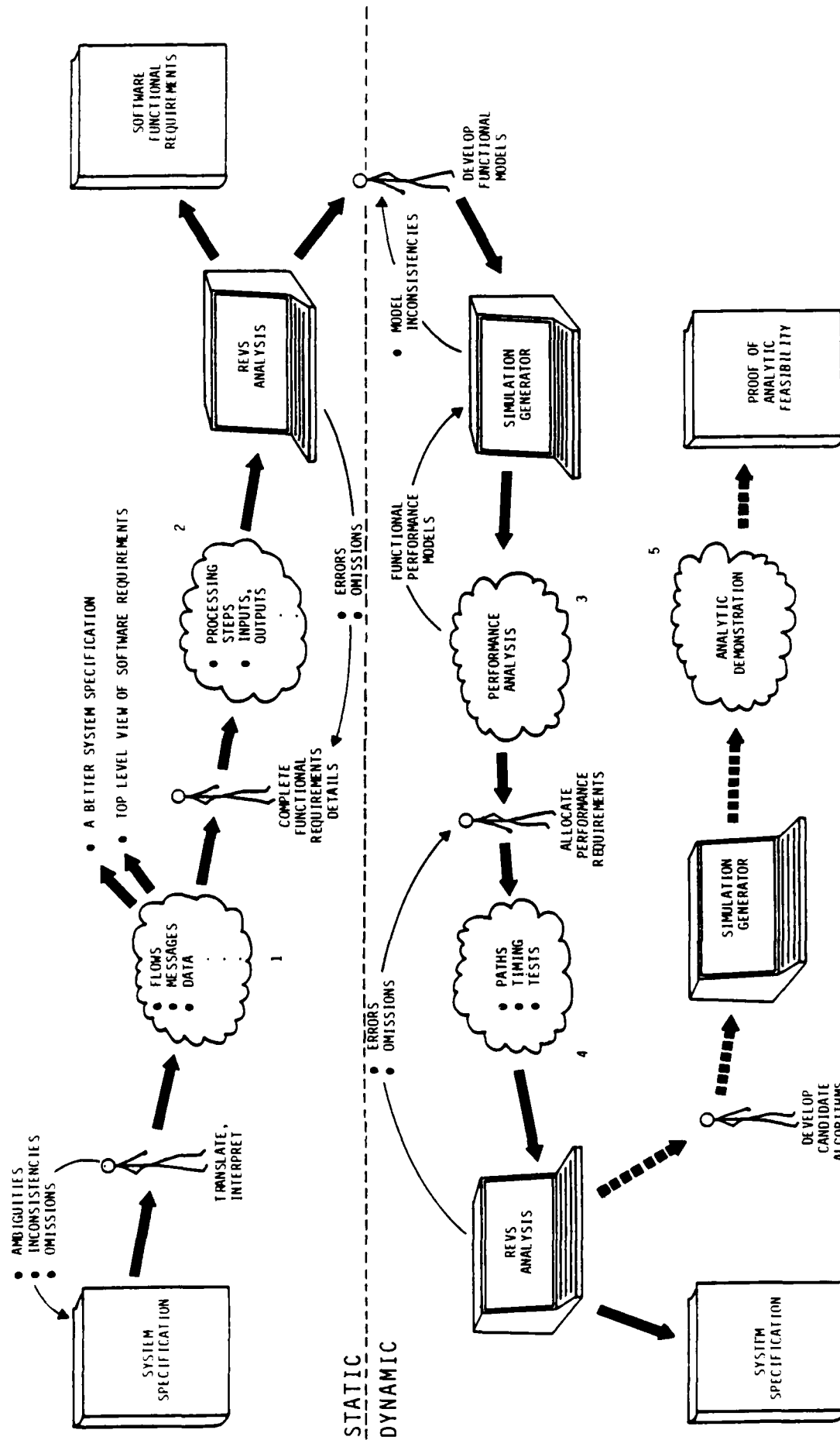
INSTRUCTOR NOTES

AT THIS POINT, THE FUNCTIONAL REQUIREMENTS FOR THE PROCESSING ARE COMPLETE. THE REMAINDER OF THE SRE METHODOLOGY ADDRESSES THE VALIDATION OF THE DYNAMIC PROPERTIES OF THESE FUNCTIONAL REQUIREMENTS VIA SIMULATION, AND THE DEFINITION AND VALIDATION OF THE PERFORMANCE REQUIREMENTS.

THE PURPOSE OF THE DYNAMIC VALIDATION IS TO VERIFY THE COMPLETENESS OF THE FUNCTIONAL REQUIREMENTS, E.G., AN OBJECT IS DETECTED, TRACKED, ENGAGED, AND THEN ITS DATA IS PURGED FROM THE SYSTEM. FOR THIS REASON, THE INTERACTION OF THE DATA PROCESSOR AND ITS ENVIRONMENT (THE SYSTEM ENVIRONMENT, THE THREAT, AND THE ACTIONS OF THE OTHER SUBSYSTEMS OF THE SYSTEM) ARE MODELED TO SIMULATE THE SEQUENCES OF MESSAGES IN AND OUT OF THE PROCESSOR.

THE PURPOSE OF THE PERFORMANCE REQUIREMENTS IS TO DEFINE HOW MUCH AND HOW WELL THE FUNCTIONAL REQUIREMENTS ARE TO BE SATISFIED, IN A MANNER WHICH IS TESTABLE, TRACEABLE TO THE SYSTEM PERFORMANCE REQUIREMENTS, AND SATISFIES OTHER DESIRABLE PROPERTIES. THE VALIDATION OF THESE PERFORMANCE REQUIREMENTS IS THEN ACCOMPLISHED AS NECESSARY VIA SIMULATION AT THE ANALYTICAL LEVEL, I.E., THE ACTUAL MESSAGE CONTENTS ARE SIMULATED, AND CANDIDATE ALGORITHMS ARE PROVIDED FOR EACH ALPHA.

PHASE 5: DYNAMIC FUNCTIONAL VALIDATION



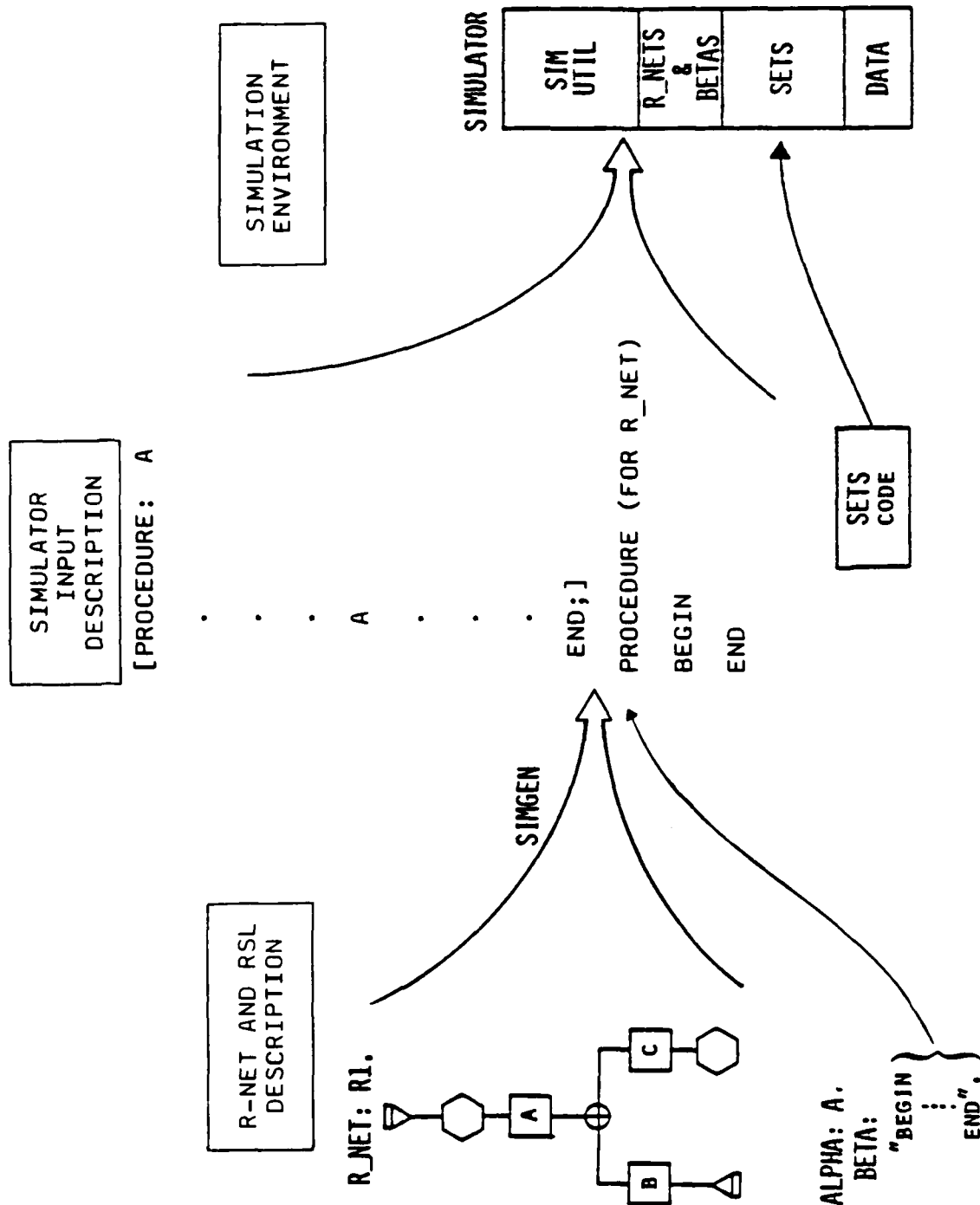
INSTRUCTOR NOTES

THE APPROACH TO FUNCTIONAL SIMULATION IS SHOWN. THE FIRST STEP IS TO DEFINE A BETA FOR EACH ALPHA ON THE R_NET STRUCTURE IN ORDER OF EXECUTION (FIRST TO LAST). THIS SHOULD BE DONE FOR EACH R_NET IN THE SAME ORDER AS THE R_NETS APPEAR, E.G., INITIALIZATION WOULD PROBABLY BE FIRST.

A BETA IS A PASCAL PROCEDURE WHICH MODELS THE TRANSFORMATION OF THE DATA AND FILES INPUT TO AN ALPHA INTO THE DATA AND FILES OUTPUT FROM IT. THE PURPOSE OF A BETA IS TO IMPLEMENT THE DATA FLOW REQUIRED FOR FUNCTIONAL SIMULATION IN ORDER TO DETERMINE THE SUFFICIENCY OF A FUNCTIONAL SPECIFICATION. THUS, IF THE PURPOSE OF AN ALPHA IS TO CHECK MESSAGE VALIDITY, THE BETA IS THE FUNCTIONAL SIMULATION OF HOW THIS SHOULD BE ACCOMPLISHED. HOW BETAS ARE WRITTEN IN PASCAL CODE WILL BE PRESENTED IN SUBSEQUENT CHARTS.

THE R_NETS, BETAS, AND DATA DEFINITIONS ARE INPUT INTO SIMGEN TO GENERATE PASCAL PROCEDURES AND DECLARATIONS. SETS IS USER SUPPLIED AND FUNCTIONALLY DEFINES THE OPERATION OF THE SUBSYSTEMS. THE PASCAL PROCEDURES AND DECLARATIONS ARE COMBINED WITH SIMULATION UTILITIES AND SETS TO FORM A SIMULATION PACKAGE.

DYNAMIC FUNCTIONAL VALIDATION APPROACH



INSTRUCTOR NOTES

THE ADVANTAGES OF THIS TYPE OF FUNCTIONAL SIMULATOR ARE LISTED HERE. THIS SIMULATOR IS AVAILABLE AT AN EARLIER DATE AND IS PRODUCED MORE EASILY.

DYNAMIC FUNCTIONAL VALIDATION ADVANTAGES

- SIMULATOR IS DIRECTLY TRACEABLE TO REQUIREMENTS
- STATIC VERIFICATION THAT ALPHA FUNCTIONAL REQUIREMENTS ARE MODELED IN BETAS
 - CHECK ALL ALPHA INPUTS ARE USED
 - CHECK ALL ALPHA OUTPUTS ARE PRODUCED
- STRUCTURE OF SIMULATOR IS PROVIDED
- STRUCTURED METHODOLOGY FOR GENERATING A SIMULATOR
- SIMGEN PROVIDES DATA TYPING FOR ALL DATA

INSTRUCTOR NOTES

THE FUNCTIONAL SIMULATOR IS GENERATED IN A NUMBER OF STEPS, EACH WITH THEIR ASSOCIATED EVALUATION CRITERIA AS BEFORE. IN GENERAL, RADX IS USED TO DEMONSTRATE THE COMPLETION OF THE DATA DEFINITION AND THE BETA GENERATION. THE RADX DATA ANALYSIS PACKAGE COMPARES THE BETA CODE TO THE ALPHA DEFINITIONS.

SIMGEN IS THEN USED TO ACTUALLY BUILD THE SIMULATOR FROM THE RSL DATA DEFINITIONS, THE BETAS, THE SETS, AND THE STANDARD UTILITIES.

FUNCTIONAL SIMULATOR DEVELOPMENT

- SUMMARY OF PRIMARY STEPS

STEP 1	DEFINE SIMULATOR OBJECTIVES
STEP 2	DEFINE DATA
STEP 3	GENERATE BETAS
STEP 4	ANALYZE DATA FLOW
STEP 5	DESIGN AND DEVELOP SETS
STEP 6	EXECUTE SIMGEN
STEP 7	DEBUG SIMULATOR VIA COMPILATION
STEP 8	PLAN TEST CASES
STEP 9	EXECUTE TEST CASES
STEP 10	PLAN PRODUCTION RUNS
STEP 11	EXECUTE PRODUCTION PLANS

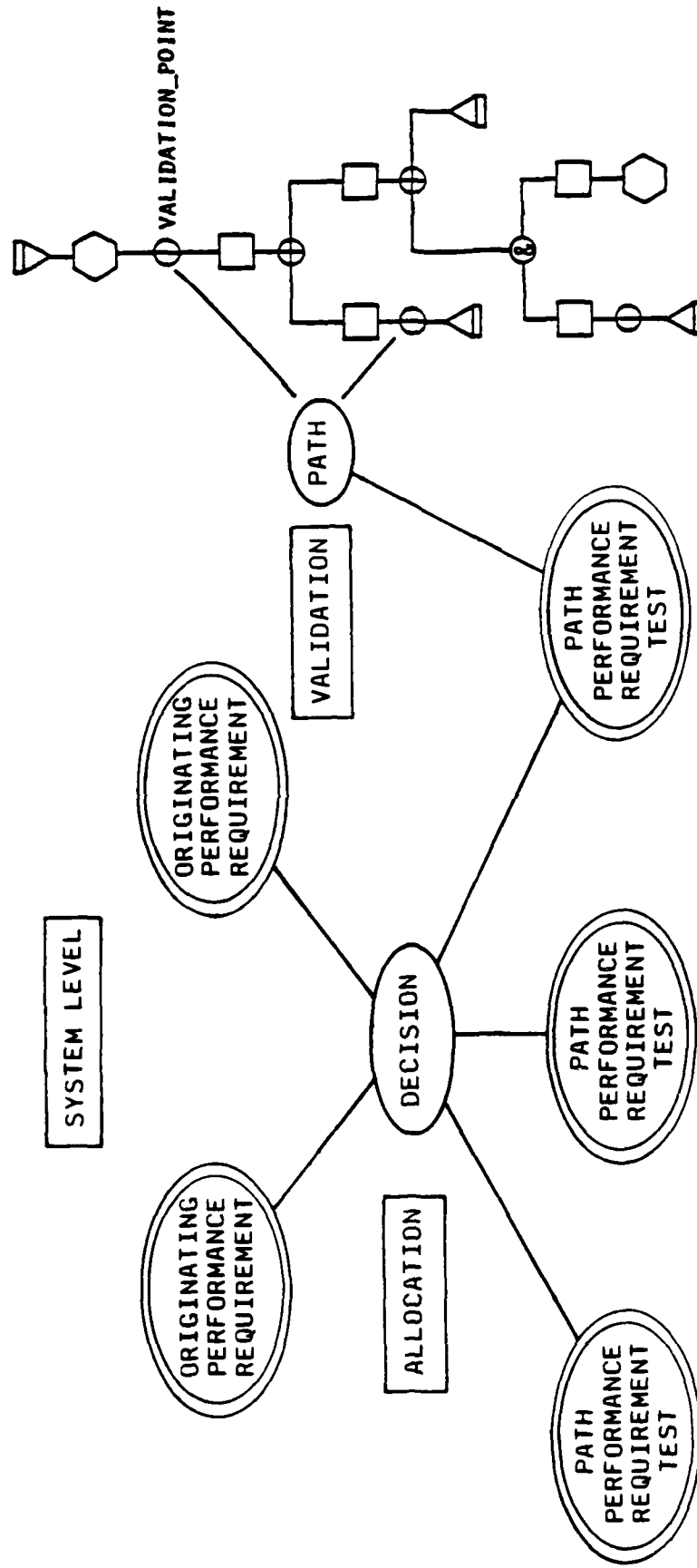
- REVS PROVIDE SUBSTANTIAL ASSISTANCE WHICH INCLUDES

- DIAGNOSTICS PERFORMED BY RADX
- SIMGEN BUILDS THE SIMULATOR
- STANDARD UTILITIES AND STRUCTURE

INSTRUCTOR NOTES

THE APPROACH TO ESTABLISHING THE PERFORMANCE_REQUIREMENTS IS SHOWN. THE SYSTEM LEVEL REQUIREMENTS ARE USED TO DERIVE THE ORIGINATING_REQUIREMENTS WHICH SPECIFY PERFORMANCE CRITERIA. NEXT, THE PERFORMANCE ORIGINATING_REQUIREMENTS ARE ALLOCATED TO PERFORMANCE_REQUIREMENTS, EITHER DIRECTLY OR THROUGH THE USE OF THE DECISION PATH. IF THE MAPPING IS ONE-TO-ONE, THEN THE PERFORMANCE ORIGINATING_REQUIREMENT CAN BE TRACED DIRECTLY TO A PERFORMANCE_REQUIREMENT TEST. IF THE MAPPING IS MANY-TO-MANY, THE DECISION MUST BE USED TO ALLOCATE THE HIGHER LEVEL REQUIREMENTS THAT IMPLY REQUIREMENTS TO THE INDIVIDUAL PERFORMANCE_REQUIREMENTS TESTS. EACH PERFORMANCE_REQUIREMENT TEST WILL CONSTRAIN A VALIDATION_PATH OR PATHS. VALIDATION_PATH(S) DEFINE THE PATH OF VALIDATION_POINTS. THESE VALIDATION_POINTS ARE INSERTED INTO THE R_NET STRUCTURE TO RECORD DATA NEEDED TO DETERMINE PASS/FAIL OF THE PERFORMANCE_REQUIREMENT TEST CRITERIA.

PHASE 6: PERFORMANCE REQUIREMENTS APPROACH



INSTRUCTOR NOTES

THIS VIEWGRAPH SHOWS A GRAPHICAL REPRESENTATION OF THE R_NET STRUCTURE, INCLUDING VALIDATION_POINTS, INTERFACES, AND ALPHA(S), AND THEIR RELATIONSHIPS TO THE VALIDATION_PATHS. THESE PATHS ARE CONSTRAINED BY THE PERFORMANCE_REQUIREMENT. EACH PERFORMANCE_REQUIREMENT TEST DEFINES A SIMPLE CRITERION. THE DATA RECORDED AT THE VALIDATION_POINTS IS INPUT INTO THE TEST ALGORITHM TO DETERMINE A PASS/FAIL CONDITION. IF THE PERFORMANCE_REQUIREMENT CRITERION IS TOO COMPLEX OR GENERAL, THEN THE PROBLEM ARISES OF WHAT TO DO IF PART FAILS AND PART PASSES. THE BURDEN IS ON THE ENGINEER TO INSURE THAT A PASS/FAIL CRITERION CAN BE DETERMINED BY THE PERFORMANCE_REQUIREMENT_TEST.

PHASE 6: PERFORMANCE REQUIREMENTS APPROACH

STEP 0: COLLECT ORIGINATING PERFORMANCE REQUIREMENTS

STEP 1: IDENTIFY ALL PROCESSING PATHS

STEP 2: ASSESS PRELIMINARY TRACEABILITY

STEP 3: DEFINE SENSITIVITY AND TRADEOFF STUDIES

STEP 4: PERFORM SENSITIVITY AND TRADEOFF STUDIES

STEP 5: DEFINE PERFORMANCE REQUIREMENTS, CONSTRAINTS, AND VALIDATION DATA

STEP 6: DEFINE PERFORMANCE TESTS

INSTRUCTOR NOTES

THE ADVANTAGES OF DOING PERFORMANCE_REQUIREMENTS IN THIS MANNER ARE LISTED ON THE VIEWGRAPH.

ALL OF THESE CAPABILITIES AID IN TRACING THE SYSTEM LEVEL PERFORMANCE REQUIREMENTS TO THE SPECIFIC AREAS OF THE SOFTWARE REQUIREMENTS. THIS PROVIDES PROJECT MANAGEMENT AND THE CUSTOMER WITH A MEANS OF DETERMINING IF THE PERFORMANCE REQUIREMENT ALLOCATION WAS DONE PROPERLY AND IF THE SOFTWARE REQUIREMENTS BEING DEFINED ARE CAPABLE OF SATISFYING THE CUSTOMER'S NEEDS.

ADVANTAGES

- TESTABLE, TRACEABLE PERFORMANCE REQUIREMENTS
- STRUCTURED FORMAT AND METHODOLOGY
- REVS AID IN
 - CHECKING COMPLETENESS
(ALL PATHS CONSTRAINED)
 - CHECKING TRACEABILITY ANALYSIS
(ALL ORIGINATING REQUIREMENTS ARE TRACED TO)
 - SAVING ALLOCATION DECISIONS FOR FUTURE MAINTENANCE
 - STRUCTURING TRACEABILITY FOR ENGINEERING REVIEW
 - PROVIDING SIMGEN TO GENERATE ALGORITHM LEVEL SIMULATIONS FOR USE IN
ALLOCATING REQUIREMENTS

INSTRUCTOR NOTES

PHASE 7 OF SREM PROVIDES FOR THE DEMONSTRATION OF DATA PROCESSING SUBSYSTEM ANALYTIC FEASIBILITY AS A PART OF THE OVERALL REQUIREMENTS VALIDATION ACTIVITY. THE DEMONSTRATION INVOLVES THE DESIGN, IMPLEMENTATION, INTEGRATION, AND TEST OF A CANDIDATE DESIGN (INCLUDING ANALYTIC ALGORITHMS OR GAMMAS) WHICH MEETS ALL THE REQUIREMENTS EXCEPT THOSE ASSOCIATED WITH TIMING. THE OBJECTIVES OF THE FEASIBILITY DEMONSTRATION ARE TO

- 1) PROVIDE A "BREADBOARD" DESIGN UNDER THE SAME LOGICAL REQUIREMENTS AS THE REALTIME PROCESS, 2) TEST THIS CANDIDATE DESIGN IN A SIMULATED ENVIRONMENT, 3) EVALUATE THE PERFORMANCE OF THE CANDIDATE PROCESS AGAINST THE SPECIFIED VALIDATION CRITERIA, AND THUS 4) INCREASE CONFIDENCE IN THE FEASIBILITY OF THE REQUIREMENTS BY ILLUSTRATING A SINGLE-POINT (NON-REALTIME) SOLUTION FOR THE REQUIREMENTS SPECIFIED.

AN EXTREMELY IMPORTANT GOAL SATISFIED BY THE DEVELOPMENT OF A FEASIBILITY DEMONSTRATION IS THE ESTABLISHMENT THAT THE SOFTWARE REQUIREMENTS ARE ACTUALLY TESTABLE. SINCE THE CANDIDATE DESIGN APPLIES THE PERFORMANCE_REQUIREMENT TEST CRITERIA IN THE SAME MANNER AS THE REALTIME PROCESS, THE PERFORMANCE EVALUATION ASSOCIATED WITH THE CANDIDATE DESIGN WILL NECESSARILY ISOLATE ANY REQUIREMENTS WHICH CANNOT BE TESTED. IF A REQUIREMENT IS FOUND TO BE UNTESTABLE OR UNVERIFIABLE THROUGH THE TESTING OF THE CANDIDATE PROCESS, THAT PARTICULAR REQUIREMENT WILL BE CHANGED TO A TESTABLE CRITERION.

RESULTS OF FORMAL TESTING OF THE CANDIDATE DESIGN AGAINST A PREDETERMINED SET OF TEST SCENARIOS CAN BE EVALUATED ALONG WITH ANY APPLICABLE UNIT TESTS THAT WERE CONDUCTED FOR INDIVIDUAL ALGORITHM VALIDATION.

PHASE 7: ANALYTIC FEASIBILITY DEMONSTRATION

- DEVELOPMENT OF FUNCTIONAL AND PERFORMANCE REQUIREMENTS IS COMPLETE

- DEVELOP CANDIDATE ENGINEERING SOLUTION

- ALGORITHMS

- GAMMAS

- ANALYTIC SIMULATION SUCCESSFUL EXECUTION COMPLETES VERIFICATION PROCESS IN SREM

INSTRUCTOR NOTES

THAT IS THE EXTENT OF THE SOFTWARE REQUIREMENTS ENGINEERING METHODOLOGY. IT IS NOW USEFUL TO PAUSE AND REFLECT ON ITS RESULTS. SREM RESULTS IN THE DEFINITION OF THE FUNCTIONAL AND PERFORMANCE REQUIREMENTS AND THEIR VALIDATION. HOW DO THESE IMPACT THE DESIGN OF THE SOFTWARE WHICH IS TO SATISFY THESE REQUIREMENTS?

FIRST, THE SOFTWARE FUNCTIONAL DESIGN MUST BE TRACEABLE OR MAPPABLE ONTO THE PROCESSING PATHS OF THE R_NETS. IT IS NEITHER NECESSARY NOR EVEN DESIRABLE THAT THE SOFTWARE DESIGN HAVE ONE SUBROUTINE FOR EACH R_NET OR ALPHA; NOR DO THE DATA AND FILES IDENTIFIED IN RSL CONSTITUTE A DATABASE DESIGN FOR THE REALTIME SOFTWARE WHICH IMPLEMENTS IT. BUT THE INPUT/OUTPUT RELATIONSHIPS OF THE R_NETS MUST BE RIGOROUSLY MAINTAINED.

NEXT, THE SOFTWARE DESIGN MUST PROVIDE A MAPPING TO POINTS IN THE SOFTWARE WHICH CORRESPOND TO THE VALIDATION_POINTS, AND TO THE DATA WHICH CONTAIN THE INFORMATION DEFINED AS RECORDED BY THE VALIDATION_POINTS. THIS IS MANDATORY FOR THE SOFTWARE TO BE TESTED. NOTE THAT THE INFORMATION CONTENT IS REQUIRED, NOT NECESSARILY THE SAME COORDINATE SYSTEM.

WHEN THESE ARE SATISFIED, THEN WHEN THE SOFTWARE IS PRODUCED, THE TESTS CAN BE USED TO TEST THE SATISFACTION OF THE PERFORMANCE_REQUIREMENTS AGAINST THE SOFTWARE.

WHERE ARE THE REQUIREMENTS IN RSL?

- SOFTWARE FUNCTIONAL DESIGN MUST BE TRACEABLE TO PROCESSING PATHS OF THE R_NETS
- SOFTWARE DESIGN MUST PROVIDE A MAPPING TO SOFTWARE EQUIVALENT OF THE VALIDATION POINTS AND ASSOCIATED RECORDED DATA
- SOFTWARE MUST SATISFY PERFORMANCE REQUIREMENTS

INSTRUCTOR NOTES

SREM IS VERY USEFUL FOR WORKING-LEVEL DOCUMENTATION. IT IS COMPACT, TO THE POINT, AND READABLE. HOWEVER, IT MAY NOT BE SUFFICIENT FOR A FORMAL SPECIFICATION. THIS CHART ILLUSTRATES HOW THE PREVIOUS INFORMATION CAN BE REFORMATTED AND PARAGRAPH NUMBERS ADDED TO GENERATE THE FORMAL DOCUMENT. CURRENTLY THIS MUST BE DONE BY HAND; HOWEVER, IT WOULD NOT BE A DIFFICULT TASK TO DEVELOP A PROGRAM TO DO THIS AUTOMATICALLY.

AUTOMATED "CLASSICAL" SPECIFICATION FORMATS COULD BE PROVIDED

3.1.4.1

INPUT INTERFACE DEFINITION.

3.1.4.1.1

MUX INPUT. THE DATA PROCESSOR SHALL INTERFACE TO THE ENGINE MULTIPLEXER AND SHALL BE CAPABLE OF RECEIVING THE ENGINE MEASUREMENTS MESSAGES.

3.1.4.1.1.1

ENGINE MEASUREMENTS MESSAGE. EACH MESSAGE OF THIS TYPE SHALL BE MADE OF SENSOR DATA AND SWITCH DATA.

3.1.4.1.1.1.1

SENSOR DATA. THIS MESSAGE COMPONENT SHALL INCLUDE THE FOLLOWING INFORMATION FIELDS:

- (a) MEASURED P1
- (b) MEASURED P2
- (c) MEASURED P3
- (d) MEASURED T1
- (e) MEASURED T2
- (f) MEASURED T3

3.1.4.1.1.1.2

SWITCH DATA. THIS MESSAGE COMPONENT SHALL INCLUDE THE FOLLOWING INFORMATION FIELDS:

- (a) MEASURED S1
- (b) MEASURED S2

INSTRUCTOR NOTES

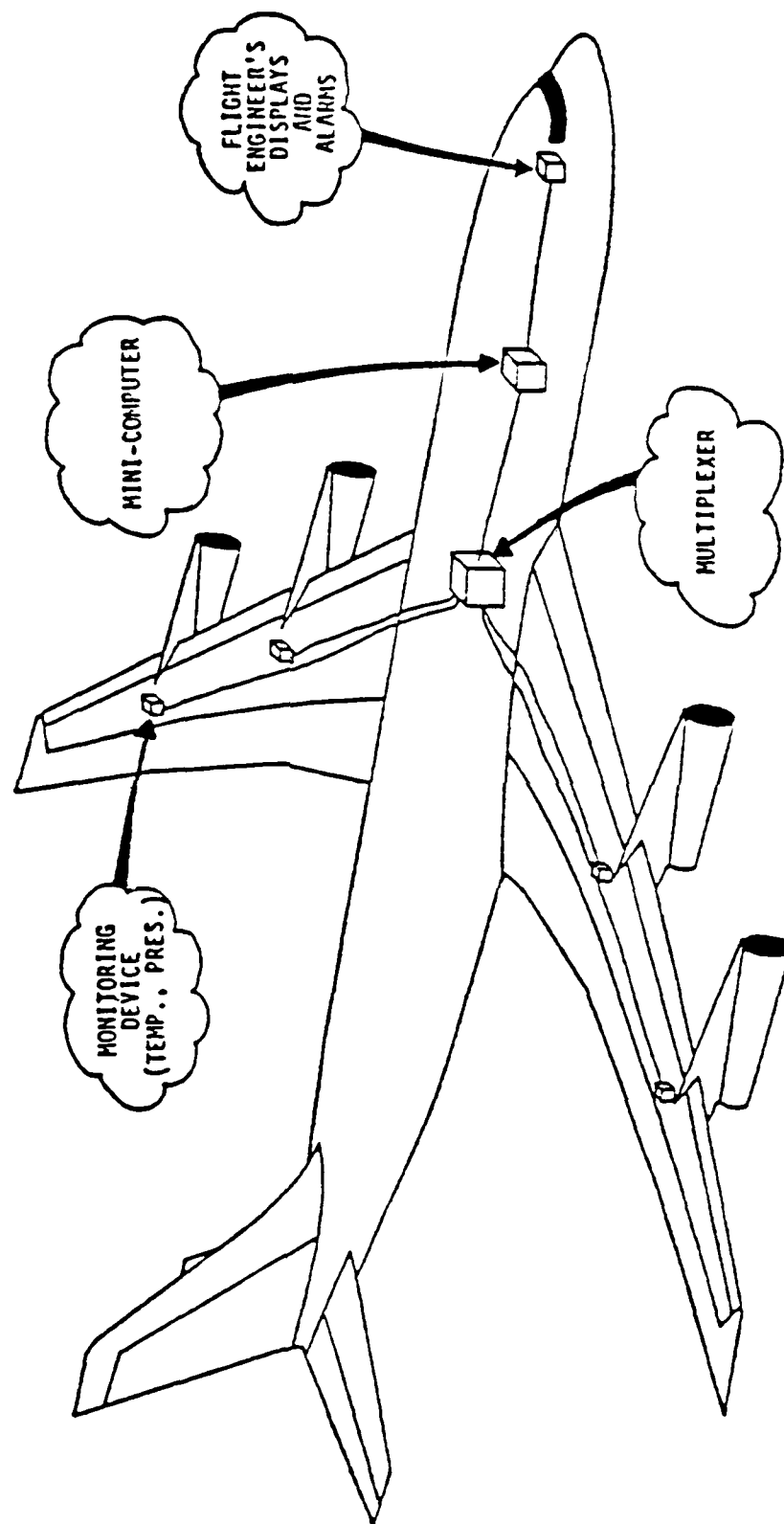
THIS SIMPLE EXAMPLE WILL BE USED TO ILLUSTRATE WHAT HAS BEEN PRESENTED THUS FAR:

AN AIRPLANE WITH MULTIPLE ENGINES HAS A DEVICE WHICH IS CONNECTED TO EACH ENGINE. THIS DEVICE MEASURES TEMPERATURE, PRESSURES AND A COUPLE OF SWITCH CLOSERS. ALL OF THE DEVICES ARE CONNECTED TO THE MULTIPLEXER. THERE IS A COMPUTER ONBOARD THE AIRCRAFT INTERROGATING THE MULTIPLEXER. THE COMPUTER SENSES WHEN A TEMPERATURE OR PRESSURE GOES OUT OF AN ALLOWED RANGE AND GIVES AN ALARM IN THE COCKPIT.

EMPHASIZE THIS IS A LOOK AT A VERY SMALL PROBLEM. WE SPECIFY REQUIREMENTS USING STEPS 1-7 OF PHASE 1. AND THEN DO A LITTLE ANALYSIS ON THESE. THIS SHOULD GIVE YOU A FEEL FOR THE MOST IMPORTANT PART OF SREM. WE CANNOT GO INTO ANY DISCUSSION OF THE OTHER PHASES.

A SREM EXAMPLE

● EXAMPLE: AN AIRCRAFT ENGINE MONITOR



● REPRESENTATIVE SEGMENTS OF PHASE 1 PROCESS WILL BE SHOWN.

INSTRUCTOR NOTES

THE FIRST STEP IS TO GENERATE THE SYSTEM SPECIFICATION AS REPRESENTED IN THIS CHART.

SYSTEM SPECIFICATION

- A MICROPROCESSOR AIRCRAFT ENGINE MONITOR FOR USE ON BOTH EXPERIMENTAL AND IN-SERVICE AIRCRAFT

- CAPABILITIES:

1. MONITOR 1 TO 10 ENGINES
2. MONITOR
 - a. 3 TEMPERATURES (0 TO 1000°C)
 - b. 3 PRESSURES (0 TO 4000 PSIA)
 - c. 2 SWITCHES (OFF, ON)
3. MONITOR EACH ENGINE AT A SPECIFIED RATE (.1 TO 10 PER SEC)
4. OUTPUT WARNING MESSAGE IF ANY PARAMETER FALLS OUTSIDE PRESCRIBED LIMITS
5. OUTPUT WARNING MESSAGE AND ACTIVATE AUDIO ALARM IF ANY PARAMETER FALLS OUTSIDE PRESCRIBED LIMITS
6. RECORD HISTORY OF EACH ENGINE

INSTRUCTOR NOTES

THIS IS AN EXAMPLE OF A BRIEF HIERARCHY OF ORIGINATING_REQUIREMENTS, A SOURCE DOCUMENT WHICH CONTAINS TWO ORIGINATING_REQUIREMENTS. THE FIRST ORIGINATING_REQUIREMENT IS BROKEN DOWN INTO ONE LOWER LEVEL WHICH IT TRACES TO THE SUBSYSTEM MULTIPLEXER AND IMPLEMENTS VERSION BASELINE_1.

PHASE 1 (STEP 1): REPRESENT DOCUMENTATION

• AN EXAMPLE:

SOURCE: SREM_CLASS_NOTES_PART_1.

DOCUMENTS

ORIGINATING_REQUIREMENT PARA_1
ORIGINATING_REQUIREMENT PARA_2.

DESCRIPTION:

"DEFINES SYSTEM, INTERFACES, AND OPERATING RULES."

ENTERED_BY: "JOHN DOE 1/1/77."

ORIGINATING_REQUIREMENT: PARA_1_1.

INCORPORATED IN PARA_1.

DESCRIPTION: "DEFINES MULTIPLEXER INTERFACES."

ENTERED_BY: "JANE SMITH 2/1/77."

TRACES TO: SUBSYSTEM MULTIPLEXER.

IMPLEMENTS: VERSION: BASELINE_1

NO-A163 300

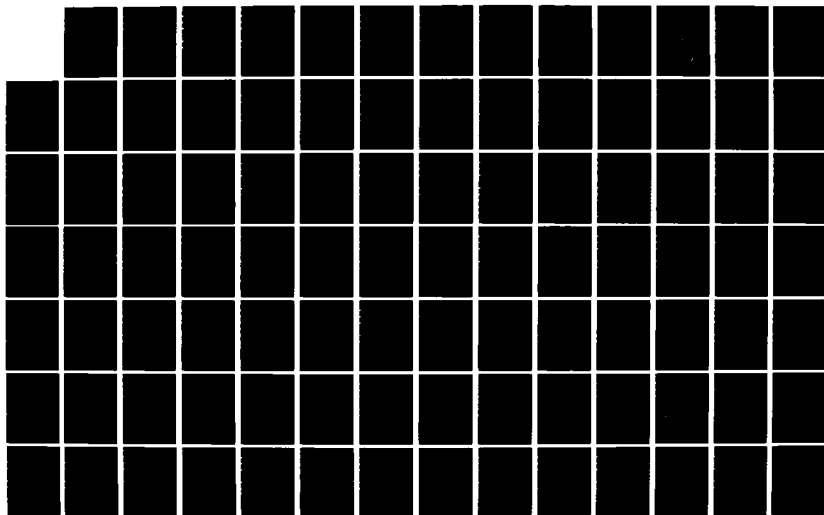
ADA (TRADEMARK) TRAINING CURRICULUM SOFTWARE
ENGINEERING METHODOLOGIES M201 TEACHER'S GUIDE VOLUME 1
(U) SOFTECH INC WALTHAM MA 1986 DAAB87-83-C-K586

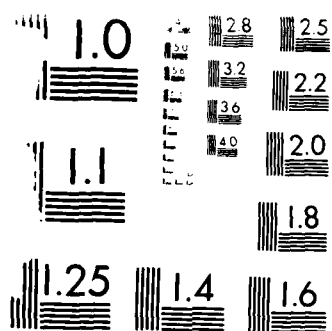
415

UNCLASSIFIED

F/G 5/9

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

INSTRUCTOR NOTES

THESE ARE EXAMPLES OF INPUT AND OUTPUT INTERFACES FOR AN ENGINE MONITORING SYSTEM.

NOTE THE INTERFACES ARE DESCRIBED AS CHANGEABLE. EVEN THOUGH THE MESSAGE DEFINITIONS SEEM COMPLETE, THE SUBSYSTEMS ARE NOT. SOME CHANGES MAY BE REQUIRED IN THE INPUT/OUTPUT INTERFACES TO COMPLETE THE SUBSYSTEM.

THE INPUT_INTERFACE "ENABLES" THE R_NET MONITOR_ENGINES. THE OUTPUT_INTERFACES DO NOT ENABLE NOR ARE THEY ENABLED BY AN R_NET. THE OUTPUT_INTERFACE FUNCTIONS WILL BE DESCRIBED IN THE NEXT SECTION ON MESSAGES.

EXAMPLES OF INPUT/OUTPUT INTERFACES

INPUT_INTERFACE : FROM_MULTI.

DESCRIPTION "3 TEMPERATURES 3 PRESSURES 2 SWITCHES."

ENABLES : R_NET MONITOR_ENGINES.

PASSES : MESSAGE ENGINE_DATA.

COMPLETENESS : CHANGEABLE.

CONNECTS TO : SUBSYSTEM MULTIPLEXER.

OUTPUT_INTERFACE : TO_ENGINEER.

DESCRIPTION "WARNING OR ALARM MESSAGE SENT WHEN AN ENGINE DATA FALLS
OUTSIDE THE LIMITS."

PASSES : MESSAGE WARNING

MESSAGE ALARM.

CONNECTS TO : SUBSYSTEM ENGINEER_STATION.

OUTPUT_INTERFACE : TO_MULTI.

DESCRIPTION "MULTIPLEXER NEEDS ENGINE ID CHANNEL NO. TO REPORT ON."

CONNECTS TO

SUBSYSTEM : MULTIPLEXER.

PASSES : MESSAGE REQUEST_NEXT_ENGINE.

COMPLETENESS : CHANGEABLE.

INSTRUCTOR NOTES

THIS CHART IS A SAMPLE OF THE DOCUMENTATION RADX CAN PROVIDE. THIS OUTPUT IS A LISTING OF THE MESSAGES AND THEIR CONTENTS PASSING ALL INPUT INTERFACES; IT IS THE FIRST STEP TO AN AUTOMATED INTERFACE SPECIFICATION. AS SUCH, IT REPRESENTS THE DEFINITION OF THE DATA REQUIRED TO BE ACCEPTED AND PROCESSED BY THE SOFTWARE.

SAMPLE OF AUTOMATED DOCUMENTATION

LIST OF INPUT INTERFACE DEFINITIONS

```
SUBSYSTEM : ENGINE_MULTIPLEXER
CONNECTED TO
  INPUT INTERFACE : MUX INPUT
  MESSAGE : ENGINE_MEASUREMENTS
    MADE BY
      DATA : MEASUREMENTS
        INCLUDES
          DATA : SENSOR_DATA
            INCLUDES
              DATA : MEASURED_P1
              DATA : MEASURED_P2
              DATA : MEASURED_P3
              DATA : MEASURED_T1
              DATA : MEASURED_T2
              DATA : MEASURED_T3
            DATA : SWITCH_DATA
              INCLUDES
                DATA : MEASURED_S1
                DATA : MEASURED_S2
          SUBSYSTEM : ENGINEERING_STATION
            CONNECTED TO
              INPUT INTERFACE : FROM_ENGINEER
              PASSES
                MESSAGE : ENGINE_SET_UP
                MADE BY
                  FILE : SET_UP_LIST
                  CONTAINS
                    DATA : SET_UP_DATA
                  INCLUDES
                    DATA : NEW_PARAMETERS
                    DATA : NEW_VALUE
                MADE BY
                  DATA : COMMAND_TYPE
                  DATA : ENG NO
                MESSAGE : HISTORY_REQUEST
```


INSTRUCTOR NOTES

THIS CHART SHOWS FOUR MESSAGES FROM THE ENGINE MONITORING SYSTEM: ENGINE DATA, REQUEST-NEXT-ENGINE, WARNING, AND ALARM.

ALL THE MESSAGES ARE MADE BY DATA AND THIS DATA IS DEFINED.

ENGINE-DATA IS AN INPUT MESSAGE. THE PASSED_BY RELATIONSHIP WAS DEFINED FOR THE INPUT_INTERFACE MULTI, THUS IT DOES NOT HAVE TO BE REDEFINED HERE.

THE REQUEST-NEXT-ENGINE, WARNING, AND ALARM MESSAGES ARE OUTPUT MESSAGES. THE FORMED_BY RELATIONSHIP WAS DEFINED FOR THE ALPHAS THAT FORM THEM, THUS THE RELATIONSHIP DOES NOT HAVE TO BE RESTATED.

EXAMPLES OF MESSAGES

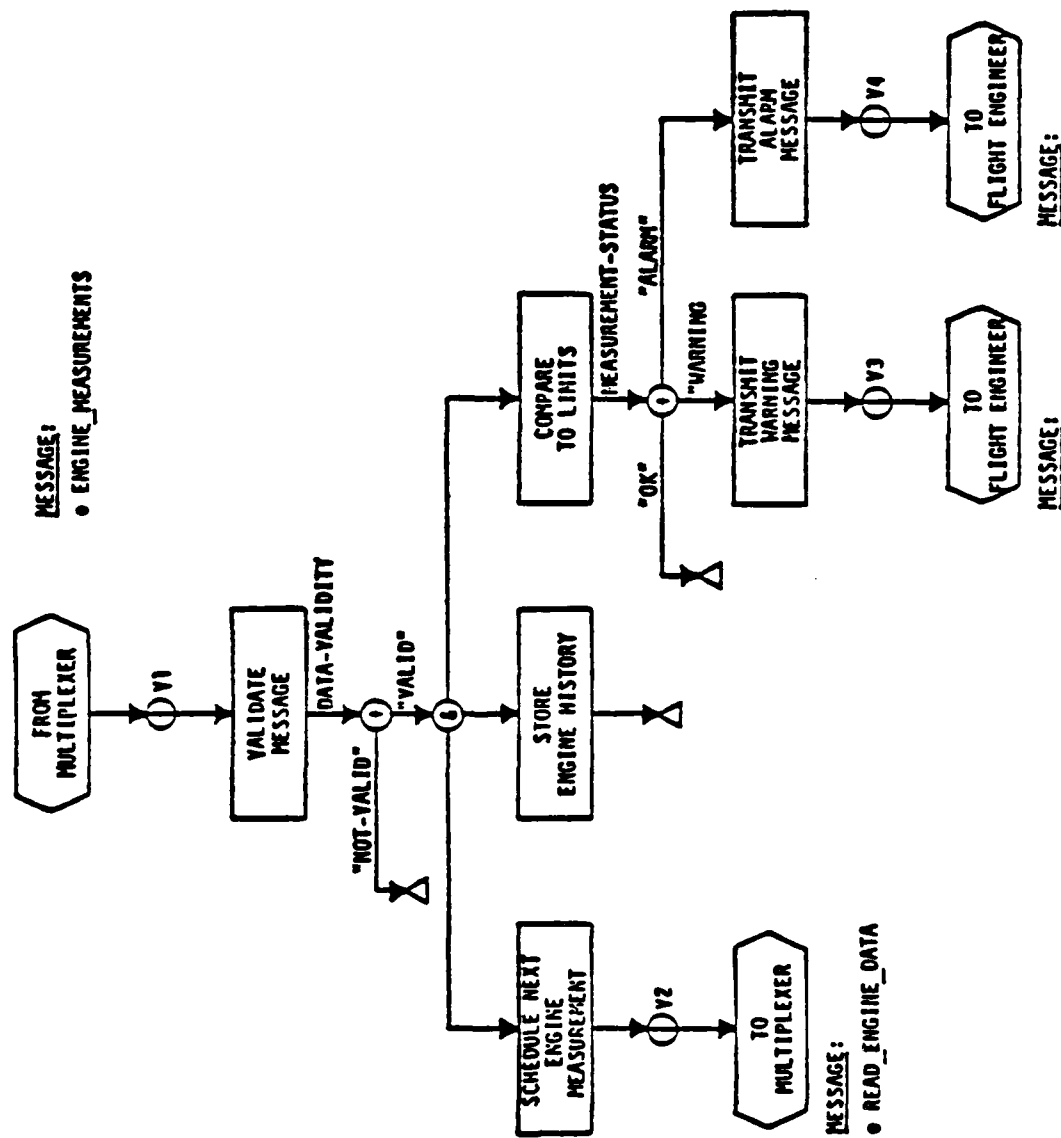
MESSAGE : ENGINE_DATA.
MADE BY
DATA MULTI_CHANNEL_NO
DATA : TEMPERATURE
DATA : PRESSURES
DATA : SWITCHES.

MESSAGE : REQUEST_NEXT_ENGINE.
MADE BY
DATA : NEXT_TIME
DATA : NEXT_CHANNEL_NO.

MESSAGE : WARNING.
MADE BY
DATA ENGINE_ID_W
DATA PARAMETER_IDS_W
DATA MESSAGE_TYPE_W.

MESSAGE : ALARM.
MADE BY
DATA ENGINE_ID_A
DATA PARAMETER_IDS_A
DATA MESSAGE_TYPE_A.

PROCESSING LOGIC IS DEFINED BY AN R-NET

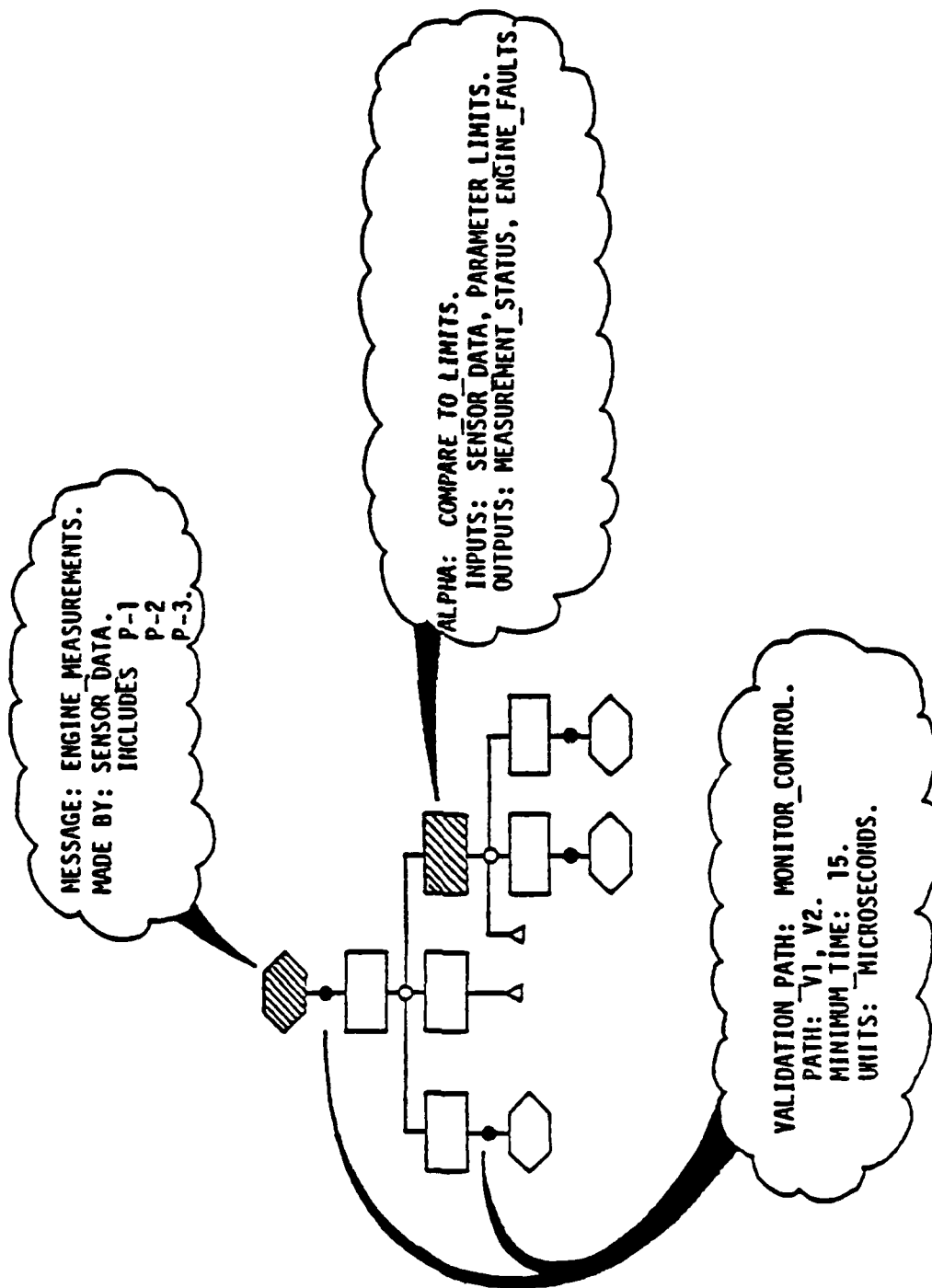


INSTRUCTOR NOTES

HERE WE SHOW THE TIE BETWEEN R-NETS AND RSL. EACH R-NET IS DETAILED BY:

- MESSAGES
- ALPHAS
- VALIDATION (CONSTRAINT) POINTS

REQUIREMENTS DETAILS ARE SPECIFIED IN RSL



INSTRUCTOR NOTES

AN EXAMPLE IS SHOWN HERE USING THE ENGINE MONITORING SYSTEM. THE ATTRIBUTES ABOVE THE DOTTED LINE ARE ENTERED TO RECORD A PROBLEM WITHOUT A SOLUTION. THIS METHOD IS USED BY REVS TO HELP THE ANALYST HIGHLIGHT UNRESOLVED PROBLEMS. THE INFORMATION BELOW THE DOTTED LINE WOULD BE ADDED AS AVAILABLE.

EXAMPLE

DECISION: MULTIPLEXER_CONTROL.

PROBLEM: "CHOICE OF MULTIPLEXER CONTROL NOT DETERMINED."

ALTERNATIVES:

- "1. SYNCHRONOUS CONTROL OF MULTIPLEXER
- 2. ASYNCHRONOUS CONTROL OF MULTIPLEXER
- 3. SYNCHRONOUS CONTROL OF MULTIPLEXER, WITH VALIDITY CHECKING."

CHOICE: "SYNCHRONOUS WITH VALIDITY CHECKING - ASSUME MULTIPLEXER LINE CAN HAVE NOISE INTERPRETED AS A MESSAGE WHICH MUST BE EDITED OUT."

COMPLETENESS: CHANGEABLE.

ENTERED_BY: "MACK ALFORD."

TRACES TO: VALIDATION_PATH REQUEST_NEXT_MEASUREMENT.

VALIDATION_PATH: REQUEST_NEXT_MEASUREMENT.

PATH: VALIDATION_POINT V1
VALIDATION_POINT V2
END.

INSTRUCTOR NOTES

IN THE EMS EXAMPLE, THE OUTSIDE WORLD OBJECTS ARE THE ENGINES. THESE OBJECTS ARE DEFINED AS AN ENTITY_CLASS. THE SPECIFIC DATA ASSOCIATED WITH EACH ENGINE IS DEFINED ON THE LEFT SIDE OF THE CHART BY HIERARCHY; IT IS EXPRESSED IN RSL ON THE RIGHT SIDE.

EXAMPLE

DEFINED BY HIERARCHY

ENTITY_CLASS.

ASSOCIATES: DATA: ENGINE_ID
DATA: MUX_CHANNEL_NO
FILE: HISTORY.

CONTAINS:

DATA MEASUREMENT.

INCLUDES:

DATA MTIME
DATA MP1
DATA MP2
DATA MP3
DATA MT1
DATA MT2
DATA MT3
DATA MS1
DATA MS2

EXPRESSED IN RSL

ENTITY_CLASS:

ENGINE.

ASSOCIATES: DATA: ENGINE_ID
DATA: MUX_CHANNEL_NO
FILE: HISTORY.

FILE: HISTORY.

CONTAINS:

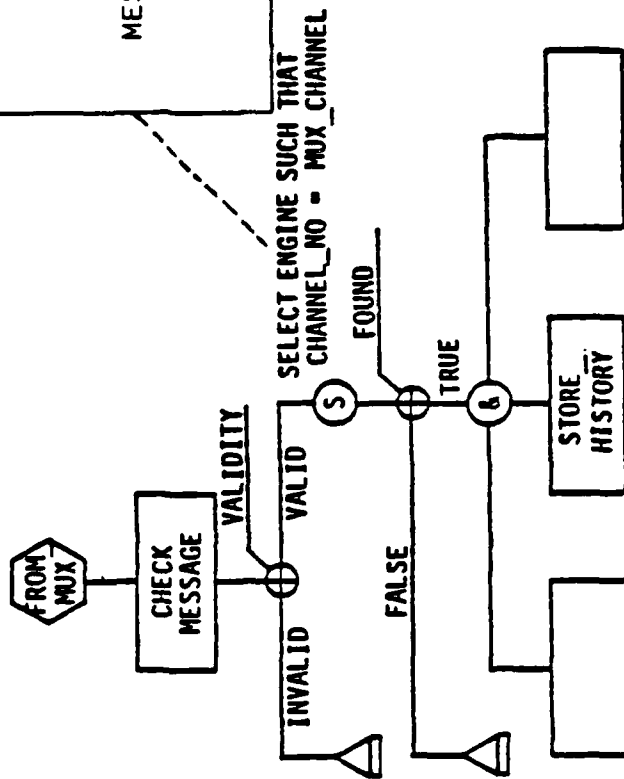
DATA: MEASUREMENT.

DATA MEASUREMENT.

INCLUDES:

DATA MTIME
DATA MP1
DATA MP2
DATA MP3
DATA MT1
DATA MT2
DATA MT3
DATA MS1
DATA MS2.

PHASE 1 (STEP 7): R-net REFINEMENT



ENTITY_CLASS: ENGINE

ASSOCIATES: DATA: ENGINE_ID
DATA: CHANNEL_NO
FILE: HISTORY

CONTAINS DATA HRECORD
INCLUDES ...

MESSAGE: ENGINE_DATA
MADE BY: DATA MUX_CHANNEL
DATA MEASUREMENT

ALPHA: STORE_HISTORY.

INPUTS: MEASUREMENT, TIME.

OUTPUTS: HRECORD.

INSTRUCTOR NOTES

REVS PROVIDES TWO MECHANISMS FOR ENTRY OF DATA INTO THE ASSM, TRANSLATION AND INTERACTIVE GRAPHICS (RNETGEN), AND A POWERFUL SET OF TOOLS FOR ANALYSIS TERMED COLLECTIVELY REQUIREMENTS ANALYSIS AND DATA EXTRACTION (RADX). TRANSLATION IS THE PROCESS OF CONVERTING RSL STATEMENTS INTO THE ASSM INFORMATION, WHERE THE SOURCE OF THE STATEMENTS MAY BE CARDS, CARD IMAGES ON TAPE, OR KEYBOARD ENTRY FROM A TERMINAL. (WE WILL BE USING CARD INPUT ONLY.)

INFORMATION HELD IN THE ASSM MAY BE SELECTED AND OUTPUT USING RADX. THAT TOOL IS RESPONSIVE TO USER DIRECTION IN SELECTING EITHER A RE-CREATION OF THE INFORMATION TRANSLATED INTO THE ASSM, OR THE FORMATTED ABSTRACTION OF THAT INFORMATION IN A USER-DEFINED HIERARCHY. THE COMBINATION OF THESE FEATURES ALLOWS COMPLEX SELECTIONS TO BE EFFECTED, SO THAT ALL INFORMATION NEEDED FOR DOCUMENTATION AND MUCH THAT IS ESSENTIAL TO CONFIGURATION MANAGEMENT CAN BE ABSTRACTED FROM THE SYSTEM WITHOUT THE ENCUMBRANCE OF IRRELEVANT DATA.

THE NEXT THREE CHARTS ARE SAMPLES OF THE ANALYSES PERFORMED BY THE RADX PORTION OF REVS. A STANDARD FILE OF RADX ANALYSES WAS APPLIED TO THE DATABASE TO CATCH SPECIFIC TYPES OF ERRORS AND TO EVALUATE THE COMPLETION OF EACH OF THE METHODOLOGY STEPS.

IN THIS CASE, REVS IDENTIFIED THE DATA SPECIFIED AS NEEDED, I.E., DATA INPUT TO SOME PROCESSING SETS, BUT WHICH HAS NO SOURCE. DATA WITHOUT A SOURCE DOES NOT HAVE AN INITIAL VALUE, IS NOT INPUT BY ANY MESSAGE, AND IS NOT OUTPUT BY ANY PROCESSING STEP. IN THIS CASE, PARAMETER LIMITS ARE REQUIRED TO DETERMINE IF WARNING OR ALARM LIMITS HAVE BEEN EXCEEDED, BUT THESE HAVE NEVER BEEN INPUT. THIS REPRESENTS EITHER AN ERROR OR A SET OF "MISSING" OR INCOMPLETE REQUIREMENTS; IN THIS CASE, IT IS THE LATTER.

AUTOMATED ERROR CHECKS PROVIDE IMMEDIATE ANALYSIS OF THE REQUIREMENTS CONSISTENCY AND COMPLETENESS

RADX

LIST OF DATA WITH NO SOURCE.

DATA : ALARM LIMITS.
DATA : CHANNEL NUM.
DATA : ENGINE ID.
DATA : ENGINE NUMBER.
DATA : ENGINE STATUS.
DATA : LIGHT ASSIGNMENT.
DATA : MONITOR RATE.
DATA : PARAMETER LIMITS.
DATA : P1_ALARM_HI.
DATA : P1_ALARM_LO.
DATA : P1_WARN_HI.
DATA : P1_WARN_LO.
DATA : P2_ALARM_HI.
DATA : P2_ALARM_LO.
DATA : P2_WARN_HI.
DATA : P2_WARN_LO.
DATA : P3_ALARM_HI.
DATA : P3_ALARM_LO.
DATA : P3_WARN_HI.
DATA : P3_WARN_LO.
DATA : S1_ALARM.
DATA : S1_WARN.
DATA : S2_ALARM.
DATA : S2_WARN.
DATA : T1_ALARM_HI.
DATA : T1_ALARM_LO.
DATA : T1_ALARM_HI.
DATA : T1_WARN_LO.

REQUIREMENTS
ANALYSIS
AND DATA
EXTRACTION

THIS DATA
IS SPECIFIED
AS NEEDED BUT
HAS NO SOURCE

REVS-RADX	INPUT-ERRORS	OUTPUT-BOTH	LOG-ALL	TRANSP-
ADD FILE ERRORS				
SELECT VIA TRACKBALL ENTRY- CONTINUE, INTERRUPT, OUTPUT OFFLINE				

INSTRUCTOR NOTES

IN THIS CHART, RADX IDENTIFIES DATA WITH NO SINK. DATA WITH NO SINK IS DATA WHICH IS PRODUCED BUT IS NEVER USED, I.E., IS NOT INPUT TO A PROCESSING STEP AND IS NOT CONTAINED IN AN OUTPUT MESSAGE. THUS THE ENGINE HISTORY THAT WAS REQUIRED TO BE MAINTAINED IS NEVER ACCESSED. THIS PROBABLY REPRESENTS AN INCOMPLETENESS, AN OVERLOOKED SET OF PROCESSING REQUIREMENTS.

THE SECOND ERROR IDENTIFIED BY RADX IS THAT NO PROVISION HAS BEEN MADE FOR ENTERING THE DATA ASSOCIATED WITH NEW ENGINES TO BE MONITORED, OR REMOVING OLD ENGINES FROM MONITORING. THIS ALSO REPRESENTS INCOMPLETENESS.

AUTOMATED ERROR CHECKS (CONTINUED)

DATA : T2 ALRM_HI.
 DATA : T2_ALRM_LO.
 DATA : T2_WARN_HI.
 DATA : T2_WARN_LO.
 DATA : T3_ALRM_HI.
 DATA : T3_ALRM_LO.
 DATA : T3_WARN_HI.
 DATA : T3_WARN_LO.

DATA : WARNING_LIMITS.
 [RADX COMMAND =
 LIST OF DATA WITH NO SINK.

DATA : ENGINE_PARAMETERS.
 [RADX COMMAND =
 LIST OF ENTITIES NOT_CREATED.

ENTITY CLASS : ENGINE.
 [RADX COMMAND =
 LIST OF ENTITIES NOT_DESTROYED.

ENTITY CLASS : ENGINE.
 [RADX COMMAND =
 LIST OF ENTITIES NOT_SET.

[RADX COMMAND =
 LIST OF UNFORMED_MESSAGES.

[RADX COMMAND =
 LIST OF RNETS NOT_ENABLED.

THIS DATA IS
 SPECIFIED AS A
 REQUIRED OUTPUT
 BUT IS NOT USED
 FOR ANYTHING

NEW ENGINES CANNOT
 BE ENTERED
 NOR DATA
 FOR OLD ONES
 ELIMINATED

REVS-RADX	INPUT-ERRORS	OUTPUT-80TH	LOG-ALL	TRANSP-
ADDFILE ERRORS				
SELECT VIA TRACKBALL ENTRY- CONTINUE, INTERRUPT, OUTPUT OFFLINE				

INSTRUCTOR NOTES

HERE, RADX IDENTIFIED THAT ACCURACIES AND RESPONSE TIME LIMITS HAVE NOT YET BEEN SPECIFIED. ALTHOUGH PERMISSIBLE AT PHASES 1 THROUGH 4 WHERE FUNCTIONAL REQUIREMENTS ARE ADDRESSED, IT SERVES TO VERIFY THAT ALL PATHS ARE COVERED BY PERFORMANCE REQUIREMENTS LATER ON.

FINALLY, THERE IS A SET OF ERROR CATEGORIES WITH NO ERRORS. THIS MEANS THAT THERE ARE NO ERRORS OF THESE TYPES.

AUTOMATED ERROR CHECKS (CONTINUED)

[RADX COMMAND =
LIST OF _UNSPECIFIED_ACCURACIES.]

VALIDATION_PATH : ALARM_RESPONSE.
VALIDATION_PATH : MULTIPLEXER_CONTROL.
VALIDATION_PATH : WARNING_RESPONSE.

[RADX COMMAND =
LIST OF _UNSPECIFIED_MAXIMUM_TIMES.]

VALIDATION_PATH : ALARM_RESPONSE.
VALIDATION_PATH : MULTIPLEXER_CONTROL.
VALIDATION_PATH : WARNING_RESPONSE.

[RADX COMMAND =
LIST OF _UNSPECIFIED_MINIMUM_TIMES.]

VALIDATION_PATH : ALARM_RESPONSE.
VALIDATION_PATH : WARNING_RESPONSE.
[RADX COMMAND =
LIST OF _MEANINGLESS_VALIDATION_PATHS.]

[RADX COMMAND =
LIST OF _USELESS_VALIDATION_POINTS.]

[RADX COMMAND =
LIST OF _USELESS_INTERFACES.]

[RADX COMMAND =
LIST OF _USELESS_ENTITIES.]

[RADX COMMAND =
LIST OF _EMPTY_ENTITIES.]

THESE MISSING
PERFORMANCE
REQUIREMENTS ARE
POTENTIAL ERRORS
OF COMPLETENESS

IF NO ERRORS OF A
GIVEN TYPE ARE LISTED,
THERE IS COMPLETE
ASSURANCE THAT
NO ERRORS
OF THAT TYPE EXIST
IN THE SPECIFICATION

REVS-RADX	INPUT-ERRORS	OUTPUT-BOTH	LOG-ALL	TRANSP-

ADDFILE ERRORS

SELECT VIA TRACKBALL ENTRY- CONTINUE, INTERRUPT, OUTPUT OFFLINE

INSTRUCTOR NOTES

SREM WAS BUILT TO ADDRESS THE DEFINITION OF PROCESSING REQUIREMENTS FOR REALTIME WEAPONS SYSTEMS WITH A SINGLE PROCESSOR. IT CAN BE EXTENDED IN MANY WAYS. WE HAVE MADE RSL AND REVS SUCH THAT RSL CAN BE EXTENDED TO RECORD AND RETRIEVE INFORMATION TAILORED TO SPECIFIC PROJECTS OR MANAGEMENT STYLES. FOR EXAMPLE, IF SOMEONE WISHED TO ADD AN ATTRIBUTE TO A DECISION WHICH RECORDED THE DUE-DATE FOR ITS RESOLUTION, OR THE NAME OF THE PERSON RESPONSIBLE, THIS COULD BE DONE IN JUST A FEW MINUTES; AND THEN RADX COULD BE USED TO RETRIEVE INFORMATION USING THIS ATTRIBUTE AS A CONDITIONAL FACTOR.

IN ADDITION, WE HAVE BEEN WORKING ON EXTENSIONS OF SREM TO ADDRESS THESE TYPES OF ISSUES WHICH ARE NOT EXPLICITLY ADDRESSED BY SREM. IN EACH CASE, SOME MINOR EXTENSIONS TO SREM AND TO RSL AND REVS ARE NECESSARY TO ADDRESS THE NEW FACTORS ASSOCIATED WITH THESE CONDITIONS.

IN CONCLUSION, WE ORIGINALLY SET THE GOALS OF DEFINING TECHNIQUES FOR LOWERING SOFTWARE COSTS WHILE GENERATING BETTER REQUIREMENTS IN A MORE VISIBLE MANNER. WE CONCLUDE THAT THESE GOALS ARE ACHIEVED BY ADDRESSING THE DESIRABLE CHARACTERISTICS SHOWN ON THE MIDDLE LEVEL OF THE CHART USING THE TECHNIQUES ON THE BOTTOM LEVEL.

SREM EXTENSIONS

- RSL EXTENSIONS

- SREM APPLICATION EXTENSIONS

- V&V EXISTING SPECIFICATIONS
- DISTRIBUTED PROCESSING
- MAN/MACHINE INTERACTIONS
- COMMAND/CONTROL PROCESSING
- BUSINESS DATA PROCESSING

INSTRUCTOR NOTES

THEME:

ENTITY DIAGRAMMING PROVIDES US WITH A DIFFERENT PERSPECTIVE ON THE REQUIREMENTS ANALYSIS PROCESS, FOCUSING ON THE INFORMATION INSTEAD OF FUNCTION.

PURPOSE:

ENTITY/RELATION/ATTRIBUTE CONCEPTS ARE USABLE BY THEMSELVES BUT ALSO SERVE AS THE BASIS FOR OTHER METHODS.

GIVES THE STUDENTS A BREAK AFTER THE NON-INTERACTIVE NATURE OF SREM.

REFERENCES:

1. MARCA, D., MCGOWAN, C., "STATIC AND DYNAMIC DATA MODELING FOR INFORMATION SYSTEM DESIGN" 6TH INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING PROCEEDINGS IEEE 82CH1795-4; SEPT. 1982.
2. BACHMAN, C., "DATA STRUCTURE DIAGRAMS" ACH SIGBDP VOL. 1, NO. 2; JOURNAL; 1973.

Section 8

ENTITY DIAGRAMMING

INSTRUCTOR NOTES

WE ARE NOW GOING TO TALK ABOUT APPROACHES THAT TAKE A SLIGHTLY DIFFERENT PERSPECTIVE ON THE ANALYSIS PROBLEM. ENTITY DIAGRAMMING EMPHASIZES DATA STRUCTURES RATHER THAN PROCESSING. BACHMAN DIAGRAMMING IS A SPECIFIC APPLICATION OF ENTITY DIAGRAMMING.

- THIS EXAMPLE COULD BE VIEWED AS A STATEMENT OF REQUIREMENTS.

REFERENCE EXAMPLE FOR THIS SECTION

- A SMALL COMMUNITY HAS ESTABLISHED A LIBRARY:
 - FOR THE USE OF LOCAL RESIDENTS
 - CONTAINING ONLY BOOKS
- THE BOOKS ARE:
 - CLASSIFIED BY CALL NUMBER
 - SHELVED SEQUENTIALLY BY CALL NUMBERS
 - CATALOGED BY AUTHORS - LISTING AUTHORS AND THE SPECIFIED BOOKS WRITTEN BY THOSE AUTHORS
 - CATALOGED BY SUBJECTS - REFERENCING BOOKS WRITTEN ON THAT SUBJECT
 - LOANED (FOR TWO WEEKS) ONLY TO LIBRARY CARD HOLDERS.
- THE LIBRARY CARDS:
 - ARE ISSUED TO RESIDENTS WHO FURNISH THEIR NAME, ADDRESS AND PHONE NUMBER TO THE LIBRARY
 - EXPIRE TWO YEARS AFTER THEY ARE ISSUED
- THE LIBRARIANS MAINTAIN AND HAVE ACCESS TO A THIRD CATALOG WHICH LISTS ALL CALL NUMBERS FOR ALL THE BOOKS IN THE LIBRARY.

OVERVIEW

- ENTITY DIAGRAM DESCRIBE INFORMATION AND ITS STRUCTURE. THEY MODEL OBJECTS AND RELATIONS IN THE REAL WORLD.
- ENTITY DIAGRAMS SUMMARIZE RELATIONSHIPS IN THE REAL WORLD.
 - HAVE BEEN USED AS THE FOUNDATION FOR OTHER METHODOLOGIES (SREM, PSL/PSA, ETC.)
- BOXES REPRESENT DATA AND ARROWS REPRESENT RELATIONSHIPS.
- WE WILL DISCUSS ...
 - ENTITY
 - ENTITY CLASS
 - RELATION
 - RELATION CLASS

INSTRUCTOR NOTES

AS ENTITY IS A SPECIFIC OBJECT. THIS PAGE IS A SPECIFIC PAGE IN THE TEACHER'S GUIDE.
THE PAGES IN THE GUIDE ARE A CLASS.

ENTITY - CHARACTERISTICS

- AN ENTITY IS ...

- STABLE; CAN BE STUDIED, DESCRIBED, OR MEASURED.
- A TANGIBLE OBJECT OR ABSTRACT CONCEPT.

- EXAMPLES

- A PARTICULAR CHAIR, TABLE, PERSON, ROOM, MACHINE
- A PARTICULAR MEETING, CONVERSATION, SPEECH
- A PARTICULAR RELIGION

INSTRUCTOR NOTES

IN THIS EXERCISE, THE STUDENT MUST IDENTIFY ENTITIES NOT CLASSES. FOR INSTANCE, "BOOK" IS NOT AN ANSWER. "A SPECIFIC BOOK" IS. THE OBJECTIVE HERE IS TO GET THE STUDENT TO EXAMINE THE REAL WORLD FROM THE APPROPRIATE VIEWPOINT: THE IDENTIFIED ENTITIES WILL BE GENERALIZED INTO ENTITY CLASSES IN ORDER TO MODEL REALITY.

1. HAWAII, BY JAMES MICHENER
2. AUTHOR IRVING WALLACE
3. SUBJECT FOSSILS
4. CALL NUMBER 813.5
5. LIBRARY CARD D13579A
6. BORROWER JOHN DOE

ENTITIES

CONSIDER THE PUBLIC LIBRARY, ITS WORKINGS, AND ITS CONTENTS. (REFER TO THE EARLIER DESCRIPTION.) LIST SOME REAL-WORLD ENTITIES INVOLVED, INCLUDING THOSE THAT ARE NOT PHYSICAL OBJECTS.

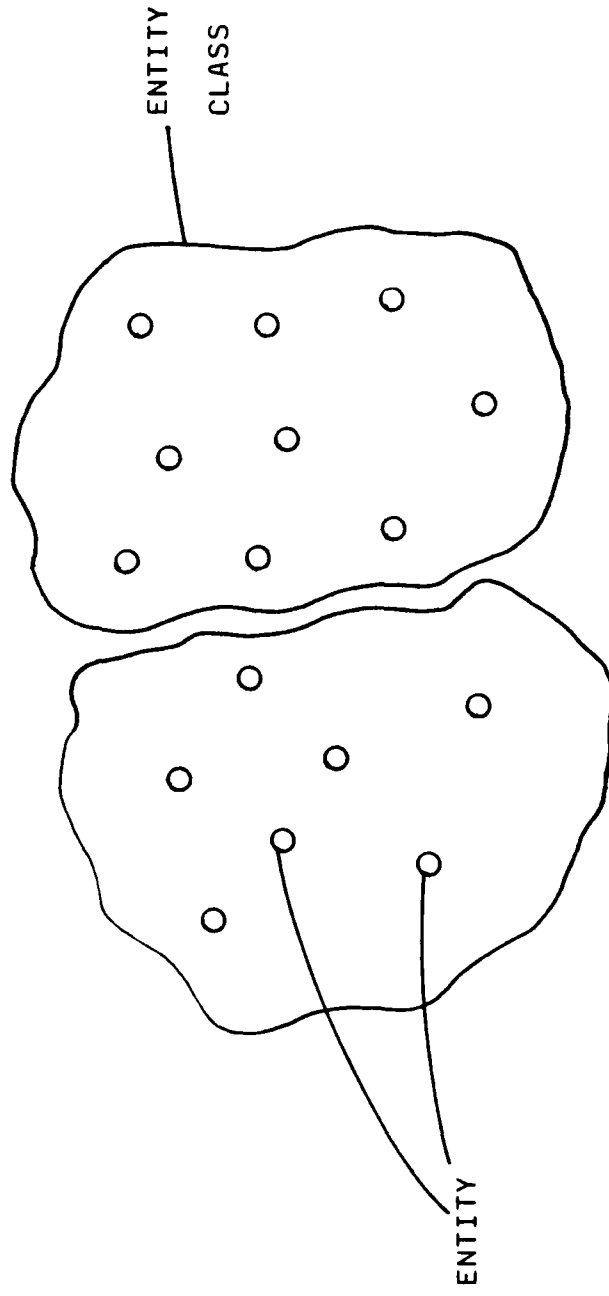
1. _____
2. _____
3. _____
4. _____
5. _____
6. _____

INSTRUCTOR NOTES

WE LOOK AT THE PROPERTIES THAT ENTITIES POSSESS IN ORDER TO GROUP ENTITIES INTO CLASSES. FOR EXAMPLE, THE PROPERTY "SEX" CAN BE USED TO DIVIDE THE CLASS INTO CLASSES OF MALES AND FEMALES.

ENTITIES - NEED FOR MEASUREMENT

- NAMED PROPERTIES OF ENTITIES MUST BE MEASURED IN ORDER TO GROUP ENTITIES TOGETHER INTO ENTITY CLASSES.



INSTRUCTOR NOTES

VG 778.1

8-6i

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
84

PROPERTIES

- A PROPERTY IS ...

- A PIECE OF INFORMATION ABOUT A SPECIFIC ENTITY.

- THE RESULT OF STUDYING, MEASURING OR EVALUATING THAT SPECIFIC ENTITY.

- EXAMPLES

(ENTITIES IN PARENTHESES) PROPERTIES UNDERLINED

(MY PEN) IS RED.

(JOHN SMITH) IS 25 YEARS OLD.

INSTRUCTOR NOTES

SOLUTION:

1. RED CLOTH COVER

367 PAGES

DECKLE EDGED PAPER (RAGGED CUT)

ROMAN TYPE FONT

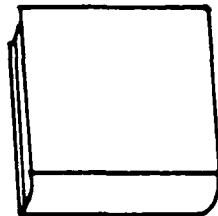
2. PUBLISHED BY ADDISON-WESLEY

TITLE IS INTRODUCTION TO DATABASES

WRITTEN BY C.J. DATE

PROPERTIES

ENTITY



1. LIST PROPERTIES FROM BOOKBINDER'S VIEWPOINT

2. LIST PROPERTIES FROM LIBRARIAN'S VIEWPOINT

INSTRUCTOR NOTES

THE CONCEPT OF AN ATTRIBUTE IS JUST A NAME WE ASSOCIATE WITH A PROPERTY OR VALUE SET.

ATTRIBUTES

- ONCE PROPERTIES ARE SPECIFIED FOR AN ENTITY, NAMES ARE GIVEN TO THOSE PROPERTIES.
- AN ATTRIBUTE IS THE NAME OF A PROPERTY.
- AN ATTRIBUTE REPRESENTS A ROLE OR INTERPRETATION OF INFORMATION ABOUT AN ENTITY.
- AN ATTRIBUTE IS DERIVED FROM CHARACTERISTICS THAT ARE IMPORTANT FOR A GIVEN PURPOSE AND VIEWPOINT.
- EXAMPLE

ENTITY CLASS
BOOK

ATTRIBUTES
TITLE
AUTHOR
PUBLISHER
PUBLICATION DATE

PERSON

NAME
ADDRESS
PHONE

INSTRUCTOR NOTES

WE HAVE TALKED ABOUT INDIVIDUAL ENTITIES THAT HAVE PROPERTIES THAT IDENTIFY THEM. WE NOW TALK ABOUT AN ENTITY CLASS WHICH GROUP ENTITIES TOGETHER BASED ON THE FACT THAT THEY ARE IDENTIFIED BY A COMMON SET OF ATTRIBUTES EACH OF WHICH MAY BE ASSIGNED DIFFERENT VALUES.

VG 778.1

8-9i

ENTITY CLASS - CHARACTERISTICS

- WHEN MEASURED, ENTITIES ARE GROUPED INTO ENTITY CLASSES
- AN ENTITY CLASS ...
 - REPRESENTS SIMILAR ENTITIES
 - HAS A NAME THAT IS A SINGULAR NOUN BY CONVENTION
 - HAS A DEFINITION THAT EXPLAINS ITS MEANING AND INTERPRETATION FOR A GIVEN PURPOSE AND VIEWPOINT

INSTRUCTOR NOTES

SOLUTION:

<u>ENTITIES</u>	<u>ENTITY CLASS DEFINITION</u>	<u>ENTITY CLASS NAME</u>
HAWAII BY JAMES MITCHNER	A BOOK SHELVED IN THE LIBRARY	BOOK
CARD #D13759A	A LIBRARY CARD ISSUED TO A PERSON	LIBRARY CARD
FOSSILS	A SUBJECT IN SEVERAL BOOKS	SUBJECT

ENTITY CLASS

- PROVIDE ENTITY CLASS NAMES FOR THE FOLLOWING ENTITIES:

<u>ENTITIES</u>	<u>ENTITY CLASS DEFINITION</u>	<u>ENTITY CLASS NAME</u>
HAWAII BY JAMES MITCHNER	A BOOK SHELVED IN THE LIBRARY	
CARD #D13759A	A LIBRARY CARD ISSUED TO A PERSON	
FOSSILS	A SUBJECT IN SEVERAL BOOKS	

INSTRUCTOR NOTES

ENTITY CLASS

BOOK

AUTHOR

PERSON

LIBRARY CARD

SUBJECT

CALL NUMBER

ATTRIBUTES

TITLE, AUTHOR, PUBLISHER

AUTHOR NAME

NAME, ADDRESS, PHONE NUMBER

CARD NUMBER

SUBJECT NAME

NUMBER

ENTITY CLASSES

• USING THE LIBRARY SCENARIO IDENTIFY IN WRITING SIX ENTITY CLASSES.

WRITE AT LEAST ONE ATTRIBUTE FOR EACH ENTITY CLASS.

ENTITY CLASS

ATTRIBUTES

INSTRUCTOR NOTES

ANOTHER IMPORTANT ASPECT OF A SYSTEM IS THE RELATIONSHIP BETWEEN ENTITIES.

RELATIONS

- ONCE ENTITIES ARE GROUPED INTO ENTITY CLASSES, THE ENTITY CLASSES ARE THEN RELATED TO EACH OTHER.
- A RELATION IS AN ASSOCIATION BETWEEN TWO ENTITIES.
- THIS ASSOCIATION DESCRIBES THE REAL-WORLD ENVIRONMENT FROM A GIVEN PURPOSE AND VIEWPOINT.
- A RELATION IS NAMED, ACCORDING TO CONVENTION, BY A VERB OR VERB PHASE.

INSTRUCTOR NOTES

SOLUTION:

<u>ENTITY</u>	<u>RELATION</u>	<u>ENTITY</u>
THE BOOK HAWAII	<u>WAS WRITTEN BY</u>	AUTHOR JAMES MICHENER
BORROWER JOHN DOE	<u>IS ASSIGNED</u>	LIBRARY CARD #D13579A
CALL NUMBER 813.5	<u>CLASSIFIES</u>	THE BOOK HAWAII BY JAMES MICHENER

RELATIONS

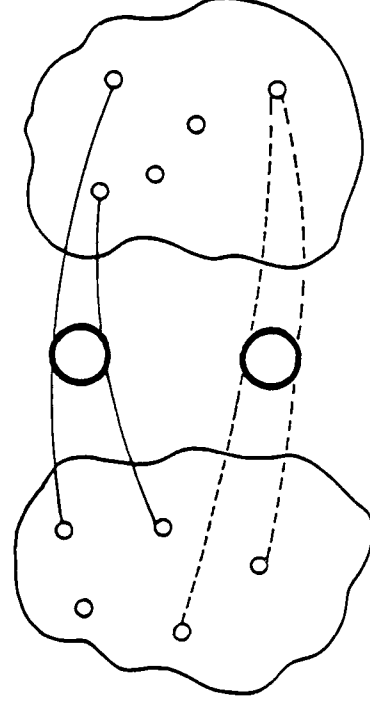
<u>ENTITY</u>	RELATION	ENTITY
THE BOOK HAWAII	_____	AUTHOR JAMES MICHENER
BORROWER JOHN DOE	_____	LIBRARY CARD #D13579A
CALL NUMBER 813.5	_____	THE BOOK HAWAII BY JAMES MICHENER

INSTRUCTOR NOTES

JUST AS WE HAVE CLASSES OF ENTITIES, WE HAVE CLASSES OF RELATIONS.

RELATION CLASS

- SIMILAR RELATIONS ARE GROUPED TOGETHER INTO RELATION CLASSES.
- A RELATION CLASS REPRESENTS A COLLECTION OF SIMILAR ASSOCIATIONS BETWEEN TWO SPECIFIED ENTITY CLASSES.
- A RELATION CLASS EXPLAINS THE ASSOCIATION BETWEEN TWO SPECIFIED ENTITY CLASSES FOR A GIVEN PURPOSE AND VIEWPOINT.
- A RELATION CLASS IS NAMED BY A VERB OR VERB PHRASE.



SOLUTION:

<u>ENTITY CLASS</u>	<u>RELATION CLASS</u>	<u>ENTITY CLASS</u>
<u>LIBRARY CARD</u>	<u>IS ASSIGNED TO</u>	<u>PERSON</u>
<u>PERSON</u>	<u>BORROWS</u>	<u>BOOK</u>
<u>AUTHOR</u>	<u>WROTE</u>	<u>BOOK</u>
<u>BOOK</u>	<u>IS ABOUT</u>	<u>SUBJECT</u>
<u>CALL NUMBER</u>	<u>CLASSIFIES</u>	<u>BOOK</u>

RELATION CLASS

- USING THE LIBRARY SCENARIO AND THE FOLLOWING ENTITY CLASS PAIRS, IDENTIFY
IN WRITING ...

ENTITY CLASS

RELATION CLASS

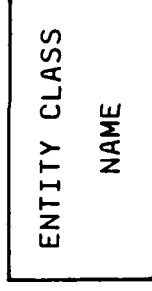
ENTITY CLASS

INSTRUCTOR NOTES

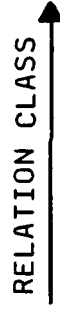
THE DIRECTIONALITY OF THE ARROW IS NOT AS SIGNIFICANT AS THE FACT THAT THE CONNECTION IS
MADE AT ALL.

SYNTAX

- ENTITY CLASSES ARE REPRESENTED BY A BOX.



- RELATIONSHIP CLASSES ARE REPRESENTED BY AN ARROW.



- ENTITY CLASS BOXES CAN BE CONNECTED TO EACH OTHER IN SEVERAL DIFFERENT WAYS.

INSTRUCTOR NOTES

NOTICE, WE COULD HAVE THE ARROW GOING THE OTHER WAY WITH THE RELATIONSHIP NOW "WAS WRITTEN BY". ANY RELATION HAS A COMPLEMENT.

VG 778.1

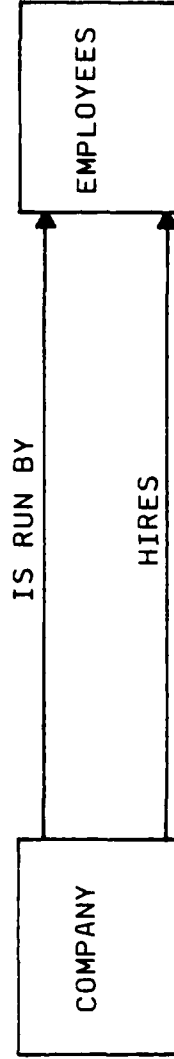
8-17i

POSSIBLE CONNECTIONS

- TWO ENTITY CLASS BOXES CAN BE CONNECTED TO EACH OTHER BY A SINGLE
RELATION CLASS ARROW.



- TWO ENTITY CLASS BOXES CAN BE CONNECTED TO EACH OTHER BY SEVERAL
RELATION CLASS ARROWS:

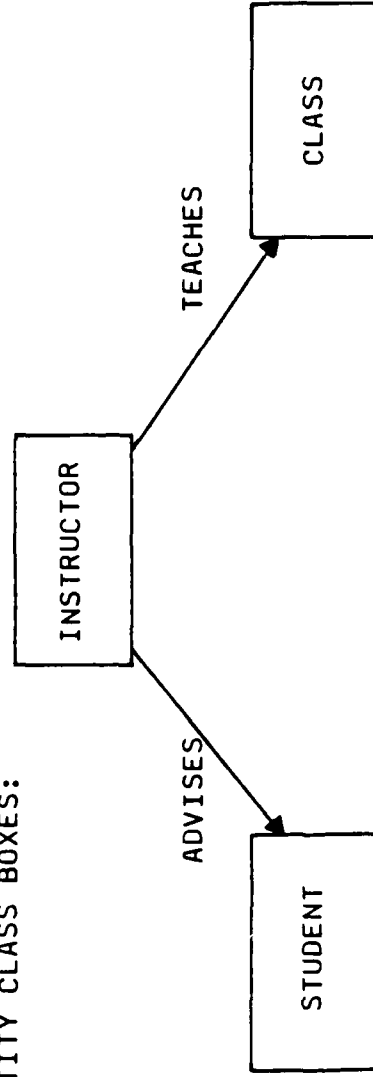


INSTRUCTOR NOTES

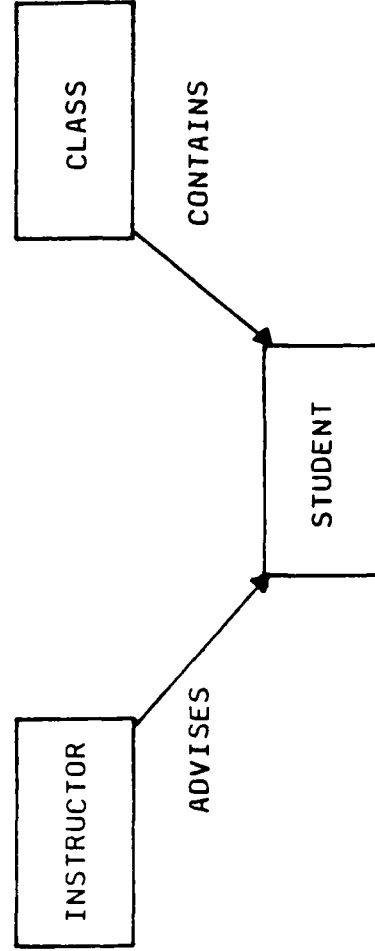
NOTE THAT "CLASS" WOULD NORMALLY CONTAIN STUDENTS, BUT HERE "CLASS" MEANS THE ABSTRACT NOTION. THE THING THAT DIFFERENTIATES THE TWO HERE ARE THEIR ATTRIBUTES. FOR INSTANCE, NAME, COLOR AND SEX FOR STUDENT, LOCATION, SUBJECT, COURSE OUTLINE FOR CLASS.

POSSIBLE CONNECTIONS

- A SINGLE ENTITY CLASS BOX CAN BE CONNECTED TO SEVERAL OTHER ENTITY CLASS BOXES:



- SEVERAL ENTITY CLASS BOXES CAN CONNECT TO THE SAME ENTITY CLASS BOX.

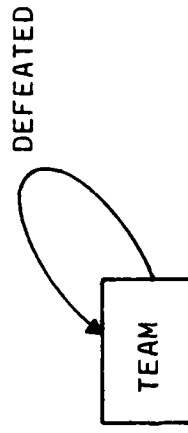


INSTRUCTOR NOTES

THIS IS A COMMON TECHNIQUE FOR HIERARCHICAL STRUCTURES.

POSSIBLE CONNECTIONS

A SINGLE ENTITY CLASS BOX CAN BE CONNECTED TO ITSELF.



SUCH A RELATIONSHIP WILL ASSOCIATE DIFFERENT ENTITIES OF THE ENTITY CLASS.

INSTRUCTOR NOTES

REMINDE THE CLASS THAT BACHMAN DIAGRAMMING IS ONE APPLICATION OF ENTITY DIAGRAMMING.

ENTITY - RELATION DIAGRAM

- A COMPLETE PICTURE OF ALL ENTITY CLASSES WITH RELATION CLASSES, IS CALLED AN ENTITY - RELATION DIAGRAM.
- BACHMAN DIAGRAMS CAN BE USED AS A FORMAT AND PROCEDURE FOR CREATING AN ENTITY-RELATION DIAGRAM.

INSTRUCTOR NOTES

BACHMAN DIAGRAMS EXTEND AS WELL AS LIMITS THE ENTITY DIAGRAMS TO MAKE THEM MORE DIRECTLY USABLE IN DATA BASE DESIGNS. THEY ALSO START TO BRIDGE THE GAP BETWEEN ANALYSIS AND DESIGN PHASES, SO SOME DESIGN LIMITATIONS ARE SPECIFIED HERE.

BACHMAN DIAGRAM DEFINITION

- BACHMAN DIAGRAMS ARE A STANDARDIZED NOTATION FOR DESCRIBING DATA AND ITS STRUCTURE. THEY MODEL DATA USED BY A SOFTWARE SYSTEM.
- BACHMAN DIAGRAMS PROVIDE A STANDARD AND CONCISE WAY OF SUMMARIZING ALL RELATIONSHIPS AMONG SOFTWARE DATA.
- BACHMAN DIAGRAMS ARE DEVELOPED FROM ENTITY DIAGRAMS, BY ASSESSING RELATION CLASS RATIOS.

INSTRUCTOR NOTES

EMPHASIZE WHAT THE RELATION CLASS RATIO IS, WHY IT IS IMPORTANT, AND
HOW THEY ARE DERIVED.

RELATION CLASS RATIO - DEFINITION

- RELATION CLASS RATIOS INDICATE THE NUMBER OF ENTITIES THAT CAN BE ASSOCIATED WITH ANOTHER ENTITY IN THE REAL WORLD.
- RELATION CLASS RATIOS PROVIDE A MECHANISM FOR EVALUATING THE RELATIONSHIPS BETWEEN TWO ENTITIES.
- THE RATIOS ARE DERIVED BY ANSWERING TWO QUESTIONS BASED UPON KNOWLEDGE OF THE SUBJECT MATTER BEING MODELED.
- RATIO QUESTIONS ARE BASED UPON A NAMED RELATION AND ITS INVERSE.



(AS DEFINED) A SPECIFIC PERSON CAN BORROW MANY BOOKS.

(INVERSE) A SPECIFIC BOOK CAN BE BORROWED BY ONE PERSON AT A TIME.

RATIO TYPES

- THERE ARE FOUR TYPES OF RELATION CLASS RATIOS FOR A GIVEN
RELATION CLASS BETWEEN TWO ENTITY CLASSES:

- ONE-TO-ONE

- ONE-TO-MANY

- MANY-TO-ONE

- MANY-TO-MANY

RATIO TYPES (ONE-TO-ONE)

- ASSUME: A PARKING SPACE IS ASSIGNED TO AN EMPLOYEE.

RATIO QUESTIONS

ANSWER

(DEFINED) 1. GIVEN A PARKING SPACE, HOW MANY

EMPLOYEES CAN IT BE ASSIGNED TO:

ZERO OR ONE

(INVERSE) 2. GIVEN AN EMPLOYEE, HOW MANY PARKING

SPACES CAN BE ASSIGNED TO HIM?

ZERO OR ONE

- THE MAXIMUM IN THE INVERSE SENSE IS ONE, AND THE MAXIMUM IN THE DEFINED SENSE IS ONE, SO PARKING SPACE IS RELATED TO EMPLOYEE IN A ONE-TO-ONE RELATION CLASS.

INSTRUCTOR NOTES

HERE'S AN EXAMPLE OF ONE-TO-MANY

RATIO TYPES (ONE-TO-MANY)

- ASSUME: A DEPARTMENT CAN CONTAIN EMPLOYEES (EMPLOYEES BELONG TO ONLY ONE DEPARTMENT).

RATIO QUESTIONS

ANSWER

(DEFINED) 1. GIVEN A DEPARTMENT, HOW MANY
EMPLOYEES CAN IT CONTAIN?

ZERO, ONE, OR MORE

(INVERSE) 2. GIVEN AN EMPLOYEE, HOW MANY
DEPARTMENTS CAN CONTAIN HIM?

ZERO OR ONE

- THE MAXIMUM IN THE INVERSE SENSE IS ONE, AND THE MAXIMUM IN THE DEFINED SENSE IS MANY, SO DEPARTMENT IS RELATED TO EMPLOYEE IN A ONE-TO-MANY RELATION CLASS.

INSTRUCTOR NOTES

HERE'S AN EXAMPLE OF MANY-TO-ONE

RATIO TYPES (MANY-TO-ONE)

- ASSUME: AN EMPLOYEE CAN SELECT AN INSURANCE PACKAGE.

RATIO QUESTIONS

ANSWER

(DEFINED) 1. GIVEN AN EMPLOYEE, HOW MANY

INSURANCE PACKAGES CAN HE SELECT?

ZERO OR ONE

(INVERSE) 2. GIVEN AN INSURANCE PACKAGE, HOW

MANY EMPLOYEES CAN SELECT IT?

ZERO, ONE, OR MORE

- THE MAXIMUM IN THE INVERSE SENSE IS MANY, AND THE MAXIMUM IN THE DEFINED SENSE IS ONE, SO AN EMPLOYEE IS RELATED TO AN INSURANCE PACKAGE IN A MANY-TO-ONE RELATION CLASS.

INSTRUCTOR NOTES

HERE'S AN EXAMPLE OF MANY-TO-MANY

VG 778.1

8-27i

RATIO TYPES

(MANY-TO-MANY)

- ASSUME: AN EMPLOYEE CAN BE ASSIGNED TO PROJECTS.

RATIO QUESTIONS

ANSWER

(DEFINED) 1. GIVEN AN EMPLOYEE, HOW MANY

PROJECTS CAN HE BE ASSIGNED TO?

ZERO, ONE, OR MORE

(INVERSE) 2. GIVEN A PROJECT, HOW MANY

EMPLOYEES CAN BE ASSIGNED TO IT?

ZERO, ONE, OR MORE

- THE MAXIMUM IN THE INVERSE SENSE IS MANY, AND THE MAXIMUM IN THE DEFINED SENSE IS MANY, SO AN EMPLOYEE IS RELATED TO A PROJECT IN A MANY-TO-MANY RELATION CLASS.

RATIO SUMMARY CHART



QUESTION		QUESTION
"A" RELATED THRU "R" TO "B"		"A" RELATED THRU "R" INVERSE "B"
ANSWERS	RATIO	ANSWERS
ZERO OR ONE	ONE TO ONE	ZERO TO ONE
ZERO, ONE OR MANY	ONE TO MANY	ZERO TO ONE
ZERO OR ONE	MANY TO ONE	ZERO, ONE OR MANY
ZERO, ONE OR MANY	MANY TO MANY	ZERO, ONE OR MANY

INSTRUCTOR NOTES

VG 778.1

8-29i

RATIO REPRESENTATION

- THE RATIO IS EXPRESSED AS TWO VALUES SEPARATED BY A COLON AND SURROUNDED BY PARENTHESES.
- THE INVERSE SENSE IS THE FIRST VALUE AND THE DEFINED SENSE IS THE SECOND VALUE.

RATIO

SYNTAX OF REPRESENTATION

ONE-TO-ONE

(1:1)

ONE-TO-MANY

(1:n)

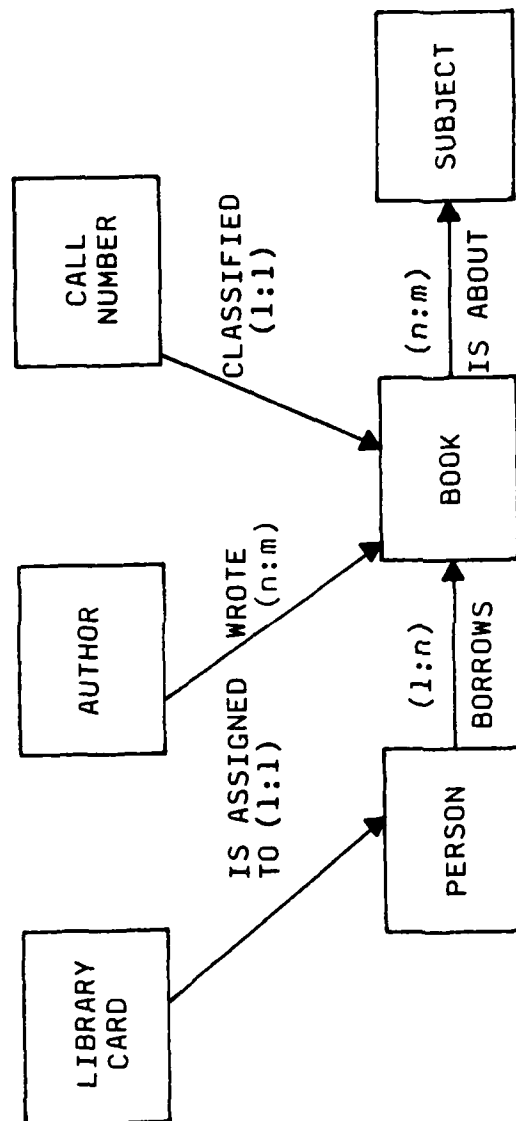
MANY-TO-ONE

(n:1)

MANY-TO-MANY

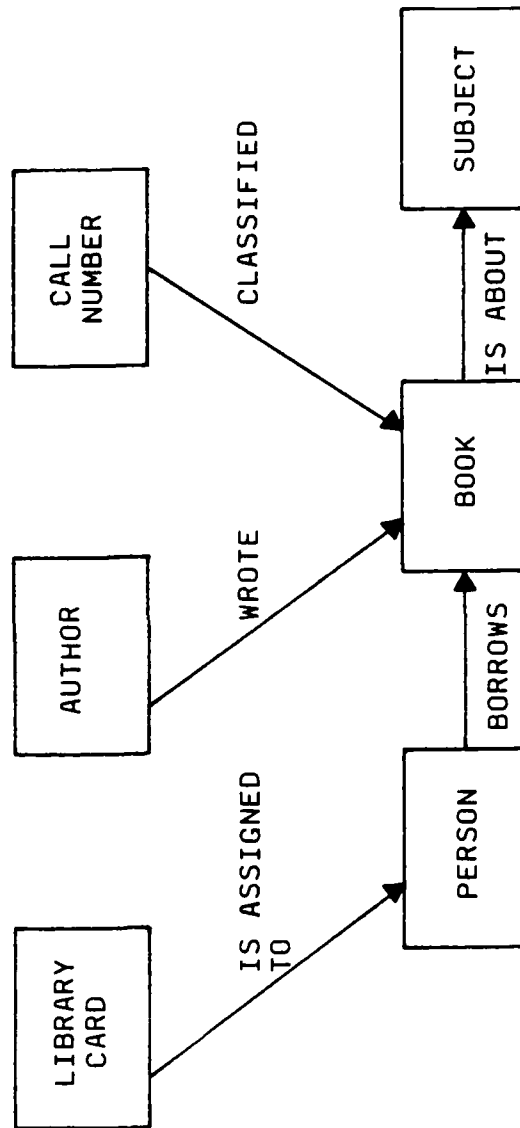
(n:m)

INSTRUCTOR NOTES



ANNOTATING AN ENTITY DIAGRAM

- GIVEN AN ENTITY RELATION DIAGRAM, ANNOTATE THE DIAGRAM WITH
RELATION CLASS RATIOS:



INSTRUCTOR NOTES

IF STUDENTS HAVE TROUBLE ACCEPTING WHY BACHMAN DIAGRAMS CAN BE MORE EASILY IMPLEMENTED THAN ENTITY RELATION DIAGRAMS, HAVE THEM ASSUME IT IS CORRECT FOR THE SAKE OF THE LECTURE. HAVE THEM TRY TO IMPLEMENT ENTITY DIAGRAMS AND BACHMAN DIAGRAMS BEFORE THE WORKSHOP BEGINS. LET THEM TRY TO DISPROVE YOU.

CONVERSION

- ONCE AN ENTITY RELATION DIAGRAM HAS BEEN ANNOTATED WITH RATIOS, THE DIAGRAM CAN THEN BE CONVERTED TO A BACHMAN DIAGRAM.
- BACHMAN DIAGRAMS ARE A RESTRICTED FORM OF ENTITY RELATION DIAGRAMS, ALLOWING ONLY THE (1:1) AND (1:n) RELATION CLASS RATIOS.
- BACHMAN DIAGRAMS CAN BE MORE EASILY IMPLEMENTED THAN ENTITY RELATION DIAGRAMS.
- THE UNACCEPTABLE RELATIONS WILL BE REPLACED BY OTHER RELATIONS AND ENTITY CLASSES.

INSTRUCTOR NOTES

THESE ENTITY CLASS RELATIONS ARE PROHIBITED IN BACHMAN DIAGRAMS.

VG 778.1

8-32i

UNACCEPTABLE/UNDESIRABLE RELATION CLASSES

- THERE ARE THREE KINDS OF UNACCEPTABLE/UNDESIRABLE RELATION CLASSES:

- UNACCEPTABLE RELATION CLASSES ARE THOSE WITH RATIOS OF:

- MANY-TO-ONE

- MANY-TO-MANY

- UNDESIRABLE RELATION CLASS OCCURS WHEN AN ENTITY CLASS

- IS RELATED TO ITSELF

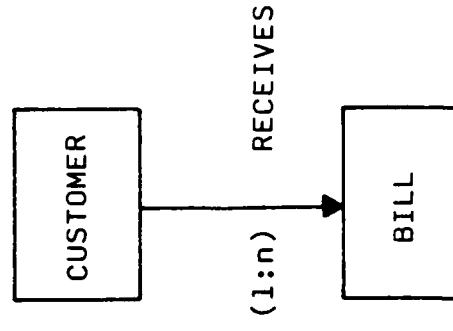
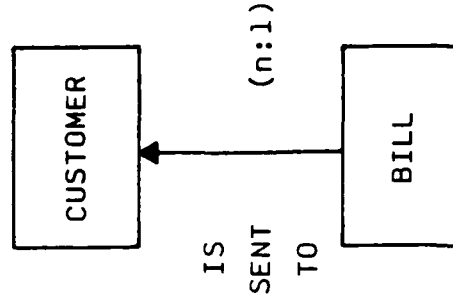
- UNDESIRABLE BECAUSE IT IS USUALLY DIFFICULT TO IMPLEMENT

INSTRUCTOR NOTES

HERE'S AN EXAMPLE ON HOW TO REPLACE THE UNACCEPTABLE MANY-TO-ONE RELATION CLASS

REPLACING MANY-TO-ONE RATIOS

- TO REPLACE A MANY-TO-ONE RELATION CLASS:
 - A) REVERSE THE ARROW DIRECTION,
 - B) CHANGE THE RELATION CLASS NAME TO ITS IMPLIED INVERSE SENSE,
AND
 - C) CHANGE THE RATIO TO ITS INVERSE SENSE.



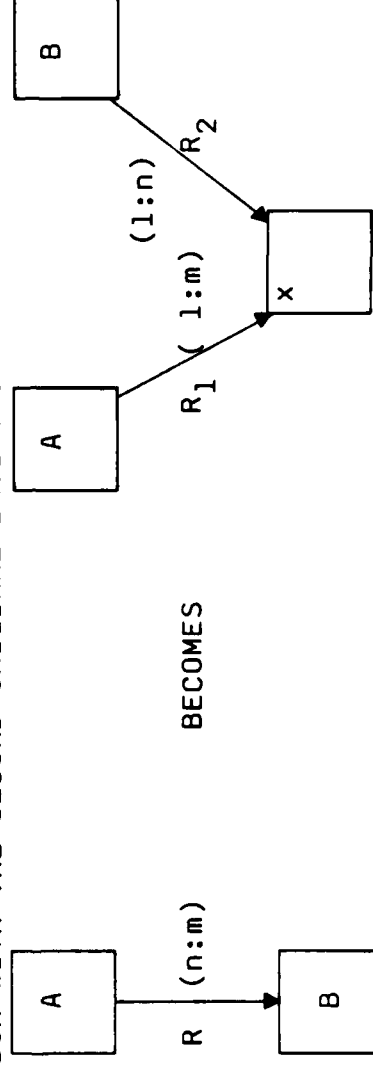
INSTRUCTOR NOTES

HERE'S AN EXAMPLE OF HOW TO REPLACE THE UNACCEPTABLE MANY-TO-MANY RATIO

REPLACING MANY-TO-MANY RATIOS

- TO REPLACE A MANY-TO-MANY RELATION CLASS:

- CREATE A NEW ENTITY CLASS BOX,
- BREAK THE ORIGINAL $(n:m)$ RELATION CLASS INTO TWO $(1:n)$ RELATION CLASSES,
- ONE $(1:n)$ RELATION CLASS RELATES THE NEW ENTITY CLASS BOX WITH THE FIRST ORIGINAL ENTITY CLASS BOX, AND
- THE OTHER $(1:n)$ RELATION CLASS RELATES THE NEW ENTITY CLASS BOX WITH THE SECOND ORIGINAL ENTITY CLASS BOX.



- THE NEW ENTITY CLASS BOX IS USUALLY LEFT UNNAMED, AND HAS A SMALL x IN THE UPPER LEFT-HAND CORNER.

INSTRUCTOR NOTES

EMPHASIZE THAT THE PLACEHOLDER IS ANONYMOUS. IT IS USUALLY NOT GIVEN A NAME.

REPLACING MANY-TO-MANY RATIOS

- THE NEW ENTITY CLASS BOX IS SIMPLY A PLACEHOLDER, PROVIDING A MECHANISM FOR RELATING ONE ENTITY IN AN ENTITY CLASS WITH MANY ENTITIES IN ANOTHER ENTITY CLASS.
- EITHER OF THE TWO NEW RELATION CLASS NAMES MAY TAKE ON:
 - A) THE ORIGINAL RELATION CLASS NAME,
 - B) A NAME SIMILAR TO THE ORIGINAL RELATION CLASS NAME, OR
 - C) THE INVERSE OF THE ORIGINAL RELATION CLASS NAME.

AD-A165 388

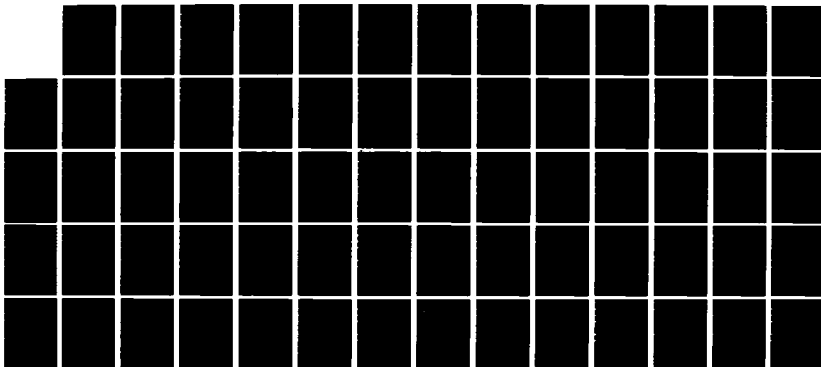
ADA (TRADEMARK) TRAINING CURRICULUM SOFTWARE
ENGINEERING METHODOLOGIES M201 TEACHER'S GUIDE VOLUME 1
(U) SOFTECH INC WALTHAM MA 1986 DAB887-83-C-K506

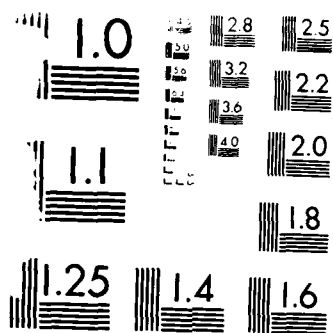
5/5

UNCLASSIFIED

F/G 5/9

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

INSTRUCTOR NOTES

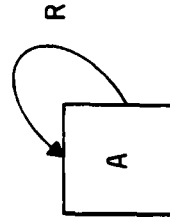
HERE'S AN EXAMPLE OF REPLACING UNDESIRABLE CYCLIC RELATION CLASSES

VG 778.1

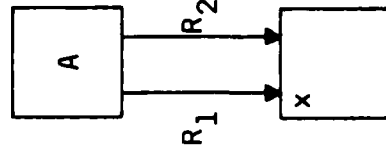
8-36i

CYCLIC RELATION CLASS

- IT IS REPLACED BY A NEW ENTITY CLASS BOX AND TWO NEW RELATION CLASS ARROWS.



BECOMES



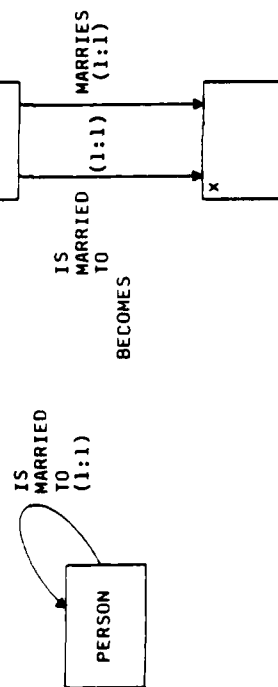
- THE NAMES OF THE TWO NEW RELATION CLASSES ARE DERIVED AS FOLLOWS:

- R_1 HAS THE SAME NAME AS R , AND
- R_2 HAS THE INVERSE NAME OF R .

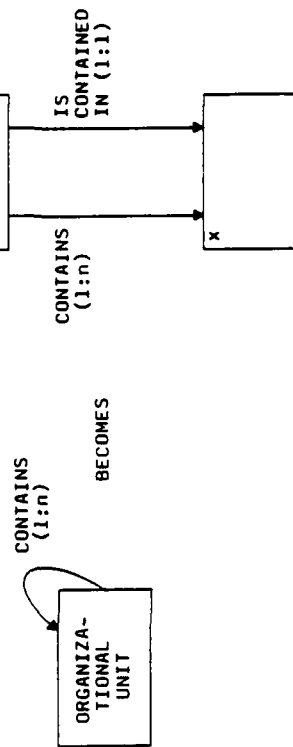
- THE NEW ENTITY CLASS BOX IS, AGAIN, A PLACEHOLDER.

REPLACING CYCLE RELATION CLASSES

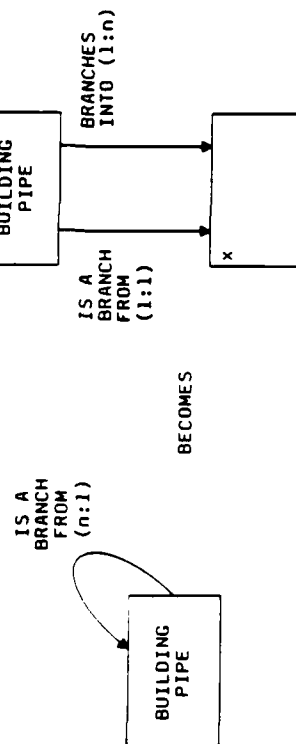
CYCLIC (1:1)



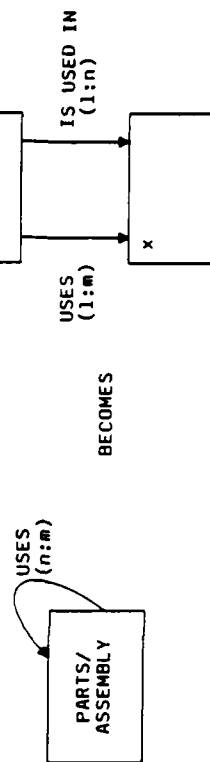
CYCLIC (1:n)



CYCLIC (n:1)



CYCLIC (n:m)



INSTRUCTOR NOTES

THEME: PSL/PSA IS A TEXTUALLY BASED METHOD FOR SPECIFYING AND ANALYZING REQUIREMENTS.

PURPOSE: PROVIDE AN OVERVIEW OF THE FAIRLY POPULAR METHODOLOGY.

REFERENCES: 1. TEICHROEW, D., HERSHEY, E., "PSL/PSA: A COMPUTER-AIDED TECHNIQUE FOR STRUCTURED DOCUMENTATION AND ANALYSIS OF INFORMATION PROCESSING SYSTEMS" IEEE TRANSACTIONS ON SOFTWARE ENGINEERING VOL. 3, NO. 1; JANUARY 1977.

2. TEICHROEW, D., "AN INTRODUCTION TO PSL/PSA" ISDOS WORKING PAPER NO. 86, UNIVERSITY OF MICHIGAN; MARCH 1974.

Section 9

PSL/PSA

INSTRUCTOR NOTES

- NOTE THAT PSL/PSA DOESN'T HELP YOU FORMALIZE THE REQUIREMENTS: IT'S JUST A TOOL.
- PSL/PSA IS BASED ON WHAT'S CALLED AN ENTITY/RELATIONSHIP MODEL. BASICALLY, RELATIONSHIPS (SPECIFIED AND IMPLIED) BETWEEN OBJECTS ARE DESCRIBED AND ARE RETRIEVABLE.
- THE TOOL IS CLAIMED TO BE METHODOLOGY INDEPENDENT. HOWEVER, BECAUSE OF THE RELATIONSHIPS AND OBJECTS ABLE TO BE DESCRIBED, AND THE PSL/PSA APPLICATION GUIDEBOOK, AN AD HOC METHODOLOGY EXISTS.
- PSL/PSA WAS ORIGINALLY DEVELOPED (CALLED URL/URA BY THE U.S. AIR FORCE) AS A DOCUMENTATION TOOL. OTHER USES OF THE TOOL FOLLOWED.
- PSL/PSA IS A DESCRIPTION VEHICLE. THAT'S WHERE ITS POWER LIES.

PSL/PSA INTRODUCTION

- PSL/PSA WAS DEVELOPED AS AN APPROACH TO IMPROVING SYSTEM/SOFTWARE DEVELOPMENT
- BASED ON THREE PREMISES:
 - MORE EFFORT AND ATTENTION SHOULD BE DEVOTED TO "FRONT-END" PORTIONS OF THE DEVELOPMENT PROCESS
 - MAKE MAXIMUM USE OF AUTOMATION DUE TO LARGE AMOUNT OF INFORMATION THAT MUST BE HANDLED
 - PUT AUTOMATION IN CORRECT PERSPECTIVE - EMPHASIZE NEED FOR DOCUMENTATION
- USE OF PSL/PSA SIMILAR TO THAT OF SREM

INSTRUCTOR NOTES

PSA HELPS VERIFY COMPLETENESS AND CONSISTENCY OF PSL DESCRIPTION. IT WILL NOT FIND ITEMS NOT DESCRIBED, OR NOT HAVING RELATIONSHIPS TO OTHER PARTS OF THE SYSTEM.

THERE ARE CASES WHERE PSA CAN HELP DETERMINE THE VIABILITY OF IMPLEMENTATION APPROACHES.

OVERVIEW

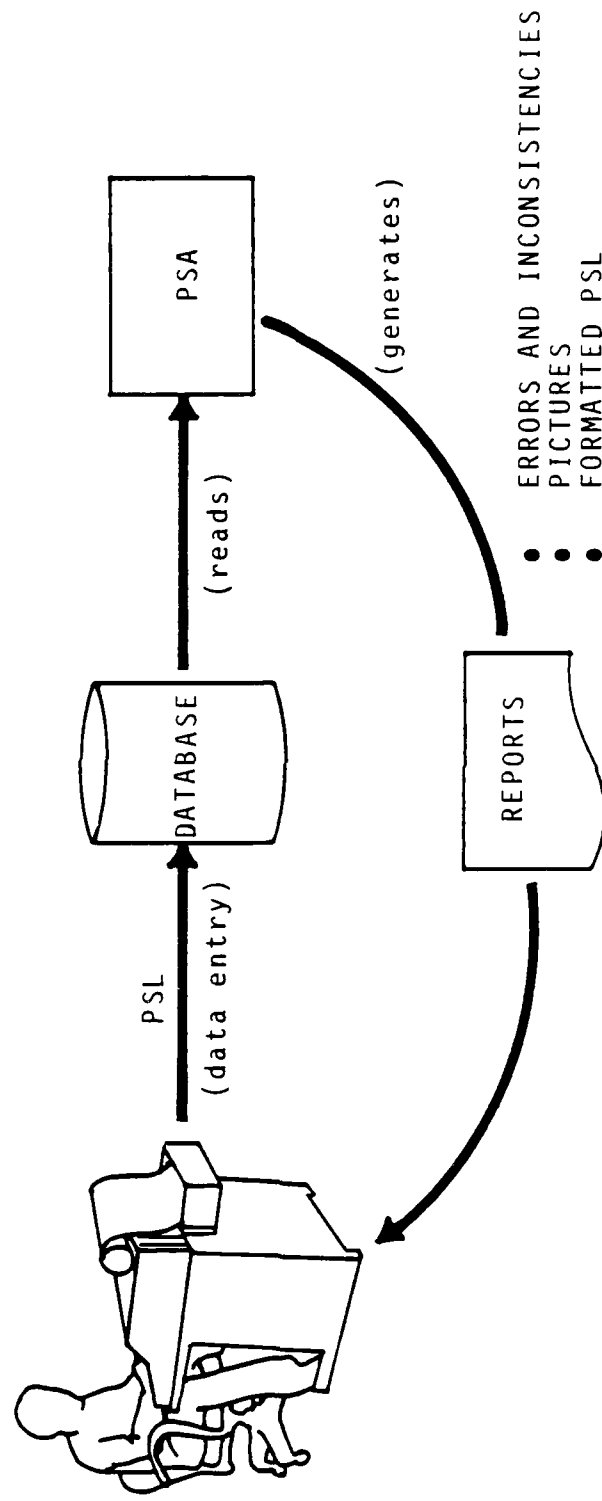
- PSL/PSA IS A TOOL:
 - DEVELOPED BY D. TEICHROEW, UNIVERSITY OF MICHIGAN,
THAT DOCUMENTS AND ANALYZES REQUIREMENTS.

- IT HAS TWO DISTINCT PARTS - PROBLEM STATEMENT LANGUAGE (PSL)
AND PROBLEM STATEMENT ANALYZER (PSA)
 - PSL EXPRESSES REQUIREMENTS
 - PSA HELPS VERIFY COMPLETENESS AND CONSISTENCY
 - PSL/PSA DOES NOT HELP
 - DETERMINE REQUIREMENTS
 - VERIFY ACCURACY OF ORIGINAL PROBLEM STATEMENT
 - DECIDE IMPLEMENTATION APPROACHES

INSTRUCTOR NOTES

PSL CAN BE INPUT IN A RANDOM MANNER SINCE IT'S NON-PROCEDURAL. PSA WILL STRUCTURE THE INPUT CORRECTLY.

INFORMATION CYCLE



INSTRUCTOR NOTES

THE STRUCTURING OF DATA AND THE POSSESSION OF PROPERTIES DEFINES A DATA DICTIONARY.

DESCRIPTION POWER

- PSL DESCRIBES A SYSTEM FROM SEVERAL ASPECTS ...
 - HIERARCHIC STRUCTURE (OF FUNCTIONS)
 - DYNAMICS (SUMMARIZE SYSTEM BEHAVIOR)
 - STRUCTURE OF DATA (HELPS ONE DEFINE A DATABASE)
 - PROPERTIES (DESCRIBE CHARACTERISTICS ABOUT THE DATA)

INSTRUCTOR NOTES

VG 778.1

9-5i

PROBLEM STATEMENT LANGUAGE (PSL)

- OBJECTIVE - EXPRESS IN SYNTACTICALLY ANALYZABLE FORM AS MUCH INFORMATION ABOUT A SYSTEM AS POSSIBLE
- ASPECTS OF A SYSTEM COVERED
 - SYSTEM INPUT/OUTPUT FLOW
 - SYSTEM STRUCTURE
 - DATA STRUCTURE
 - DATA DERIVATION
 - SYSTEM SIZE AND VOLUME
 - SYSTEM DYNAMICS
 - SYSTEM PROPERTIES
 - PROJECT MANAGEMENT
- ASPECTS DESCRIBED IN TERMS OF A SET OF OBJECT TYPES AND RELATIONSHIPS AMONG THEM
 - CONCEPTUALLY SIMILAR TO ENTITY DIAGRAMMING

INSTRUCTOR NOTES

- (a) AN OBJECT IS ANYTHING GIVEN A PSL NAME BY THE PSL/PSA USER.
- (b) USING THESE OBJECT TYPES AND RELATIONSHIPS IN THIS MANNER, STRUCTURES CAN BE DESCRIBED SUCH AS HIERARCHICAL TREES.
- (c) MENTION THAT THERE ARE OVER 20 OBJECT AND 50 RELATIONSHIPS IN PSL/PSA.

MAJOR PSL KEYWORDS

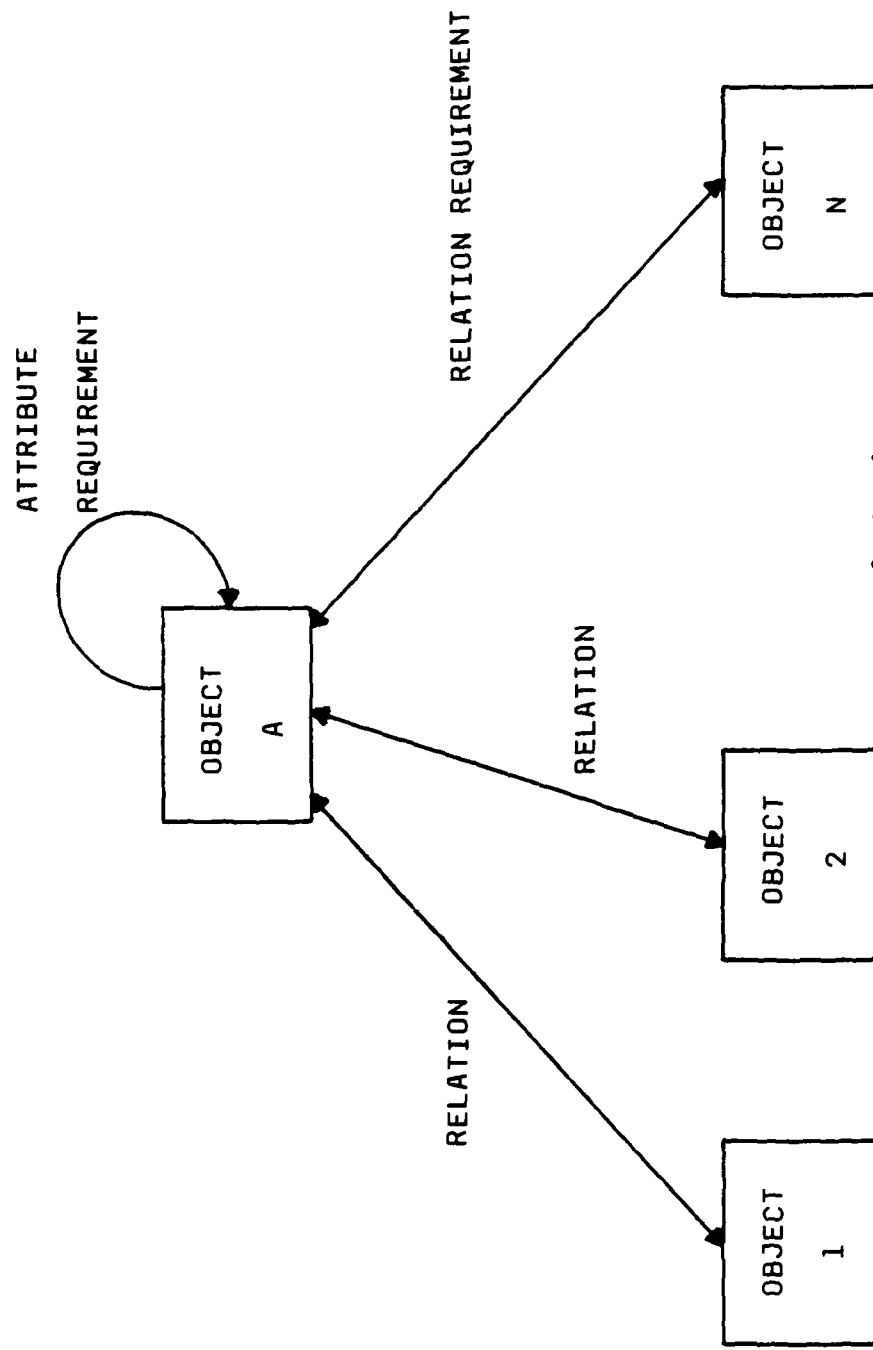
<u>OBJECTS</u>	<u>RELATIONSHIPS</u>
INPUT	PART OF
OUTPUT	CONTAINED
INTERFACE	USE
PROCESS	DERIVE
SET	UPDATE
ELEMENT	RECEIVE
GROUP	GENERATE
RELATION	CONSIST
ENTITY	CONSUMES
SYSTEM-PARAMETER	PERFORMED BY
INTERVAL	MAKES...
CONDITION	TERMINATION
EVENT	INCEPTION
	HAPPENS
	TRIGGERS

.NSTRUCTOR NOTES

THIS IS A CONCEPTUAL MODEL OF PSL. SOME OBJECT "A" POSSESSING SOME ATTRIBUTES IS
RELATED TO OTHER OBJECTS IN A PARTICULAR WAY.

DRAW THE PARALLEL TO ER DIAGRAMS.

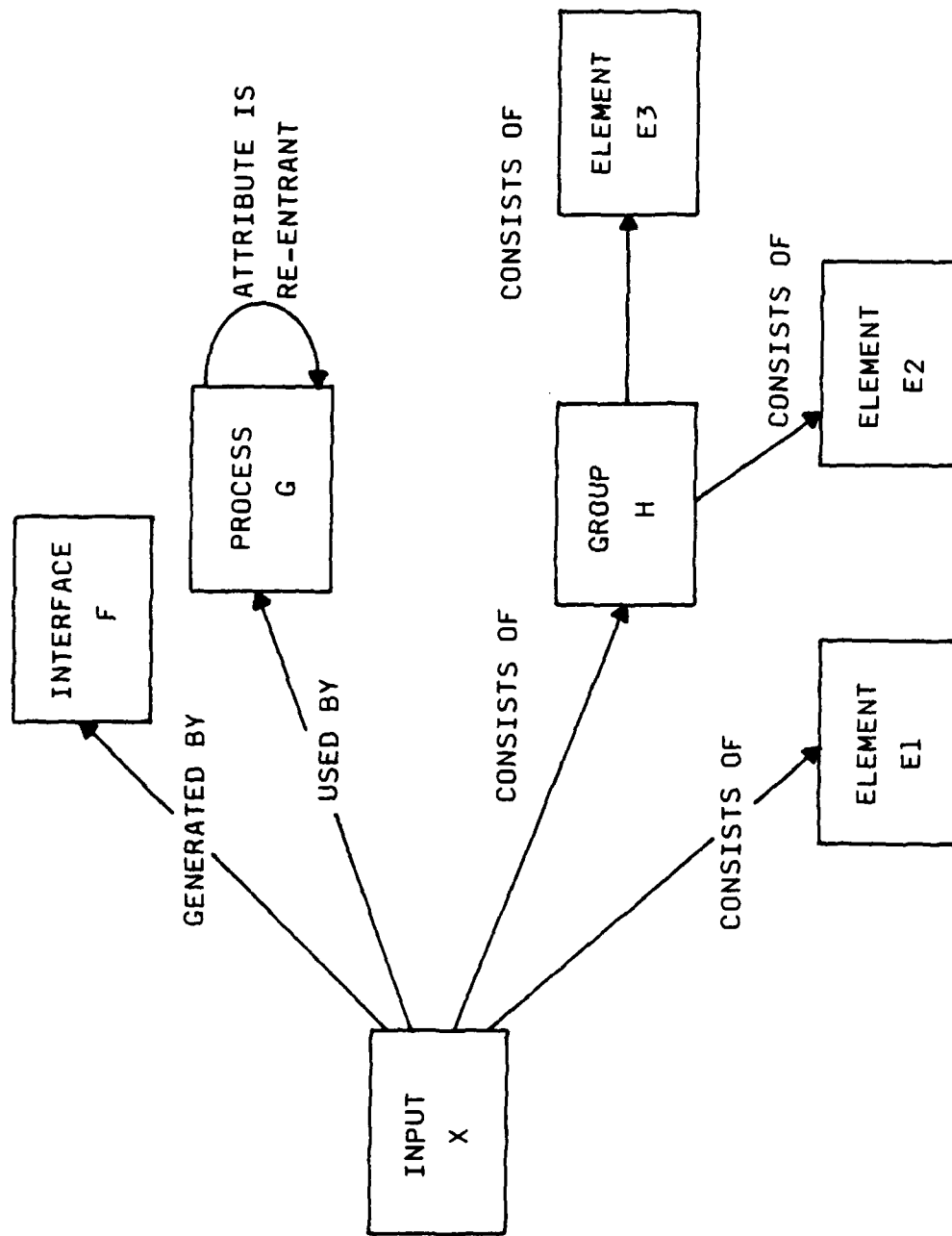
BASIS OF PSL



INSTRUCTOR NOTES

THIS SHOWS SOME DATA OBJECT AND ITS RELATIONSHIPS.

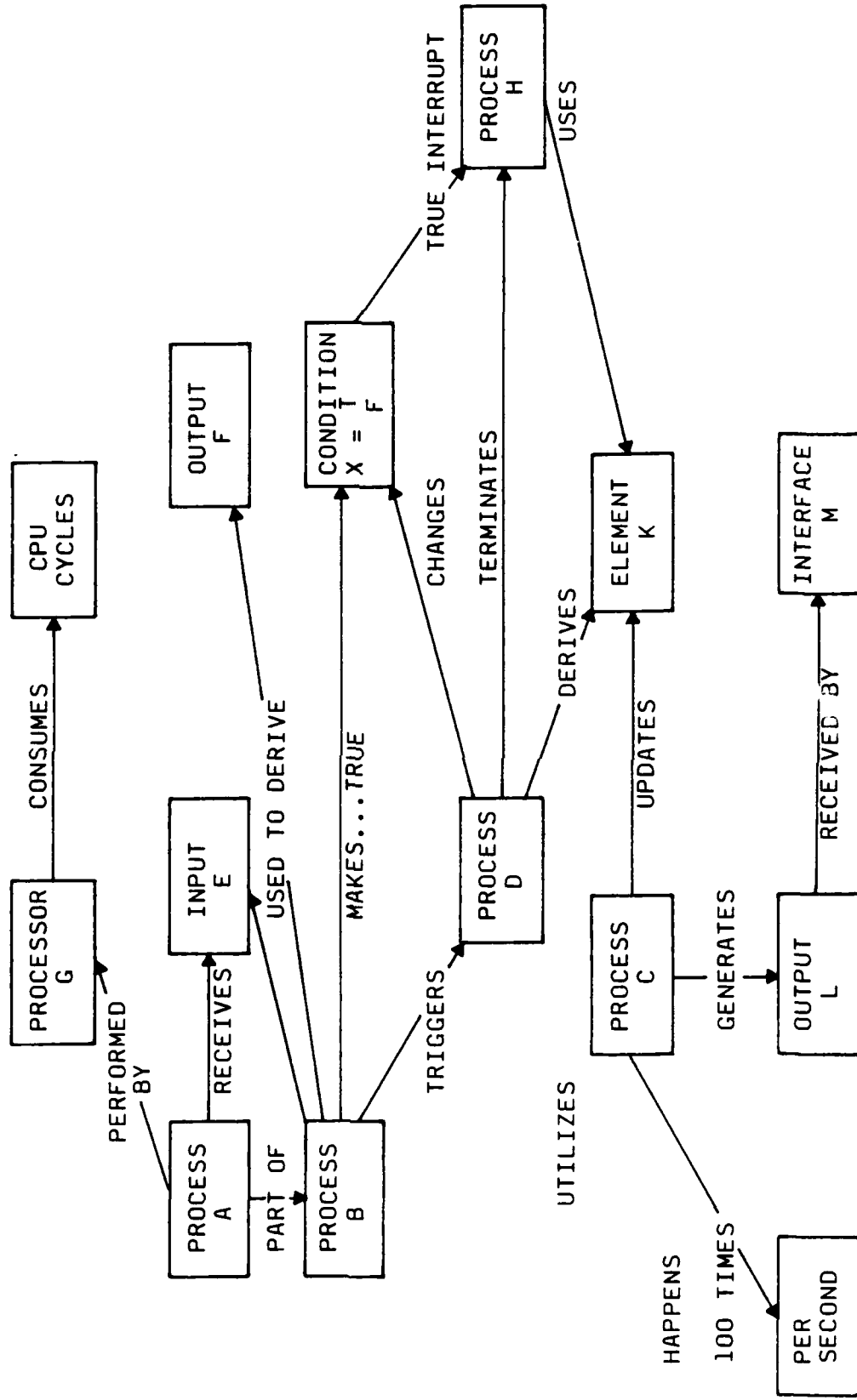
DATA EXAMPLE



INSTRUCTOR NOTES

THIS SHOWS A CONCEPTUAL MODEL EXAMPLE OF A PROCESS.

PROCESS EXAMPLE



INSTRUCTOR NOTES

JUST A QUICK WALKTHROUGH HERE. NOTE THE KEYWORD TITLES ON THE LEFT. HIGHLIGHT THE HIGHLY TEXTUAL FORMAT.

MENTION THAT THE "NOTE" IS ONLY INTERNAL COMMENTING AND IS NOT CHECKED BY THE TOOL.
"STATUS CODE" HAS TO BE DESCRIBED ELSEWHERE TO BE CHECKED.

WHAT IT LOOKS LIKE

PSL FORMATTED
PROBLEM STATEMENT FRAGMENT

PROCESS

GENERATES
RECEIVES
SUBPARTS ARE

PART OF
DERIVES
USING
DERIVES
USING
DERIVES
USING
PROCEDURE

HOURLY-EMPLOYEE-PROCESSING:

: PAY-STATEMENT, ERROR-LISTING, HOURLY-EMPLOYEE-REPORT
: TIME-CARD;
: HOURLY-PAY-CHECK-VALIDATION, HOURLY-EMP-UPDATE,
: HOURLY-REPORT-ENTRY-GENERATION,
: HOURLY-PAYCHECK-PRODUCTION;
: PAYROLL-PROCESSING;
: PAY-STATEMENT
: TIME-CARD, HOURLY-EMPLOYEE-RECORD;
: HOURLY-EMPLOYEE-REPORT
: TIME-CARD, HOURLY-EMPLOYEE-RECORD;
: ERROR-LISTING
: TIME-CARD, HOURLY-EMPLOYEE-RECORD;
: 1. COMPUTE GROSS PAY FROM TIME CARD DATA.
: 2. COMPUTE TAX FROM GROSS PAY.
: 3. SUBTRACT TAX FROM GROSS PAY TO OBTAIN NET PAY.
: 4. UPDATE HOURLY EMPLOYEE RECORD ACCORDINGLY.
: 5. UPDATE DEPARTMENT RECORD ACCORDINGLY.
: 6. GENERATE PAYCHECK.
NOTE: IF STATUS CODE SPECIFIES THAT THE EMPLOYEE DID NOT WORK
THIS WEEK, NO PROCESSING WILL BE DONE FOR THIS EMPLOYEE;
HAPPENS : NUMBER-OF-PAYMENTS TIMES-PER PAY-PERIOD;
TRIGGERED BY : HOURLY-EMP-PROCESSING-EVENT;
TERMINATION-CLAUSES: NEW-EMPLOYEE-PROCESSING-EVENT;
SECURITY IS : COMPANY-ONLY;

INSTRUCTOR NOTES

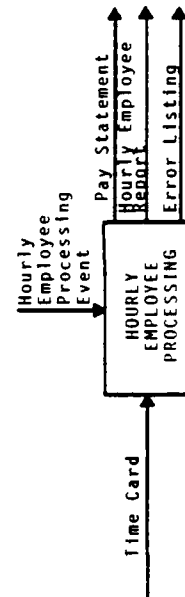
SHOW HOW THE "HAPPENS" RELATIONSHIP COULD BE PLACED ON THE PSEUDO SADT DIAGRAM.

SUMMARY INFORMATION

PROCESS	HOURLY-EMPLOYEE-PROCESSING:
GENERATES	: PAY-STATEMENT, ERROR-LISTING, HOURLY-EMPLOYEE-REPORT
RECEIVES	: TIME-CARD;
SUBPARTS ARE	: HOURLY-PAY CHECK-VALIDATION, HOURLY-EMP-UPDATE, HOURLY-REPORT-ENTRY-GENERATION, HOURLY-PAYCHECK-PRODUCTION;
PART OF	: PAYROLL-PROCESSING;
DERIVES	: PAY-STATEMENT
USING	: TIME-CARD, HOURLY-EMPLOYEE-RECORD;
DERIVES	: HOURLY-EMPLOYEE-REPORT
USING	: TIME-CARD, HOURLY-EMPLOYEE-RECORD;
DERIVES	: ERROR-LISTING
USING	: TIME-CARD, HOURLY-EMPLOYEE-RECORD;
PROCEDURE	: 1. COMPUTE GROSS PAY FROM TIME CARD DATA. 2. COMPUTE TAX FROM GROSS PAY. 3. SUBTRACT TAX FROM GROSS PAY TO OBTAIN NET PAY. 4. UPDATE HOURLY EMPLOYEE RECORD ACCORDINGLY. 5. UPDATE DEPARTMENT RECORD ACCORDINGLY. 6. GENERATE PAYCHECK.

NOTE: IF STATUS CODE SPECIFIES THAT THE EMPLOYEE DID NOT WORK THIS WEEK, NO PROCESSING WILL BE DONE FOR THIS EMPLOYEE.

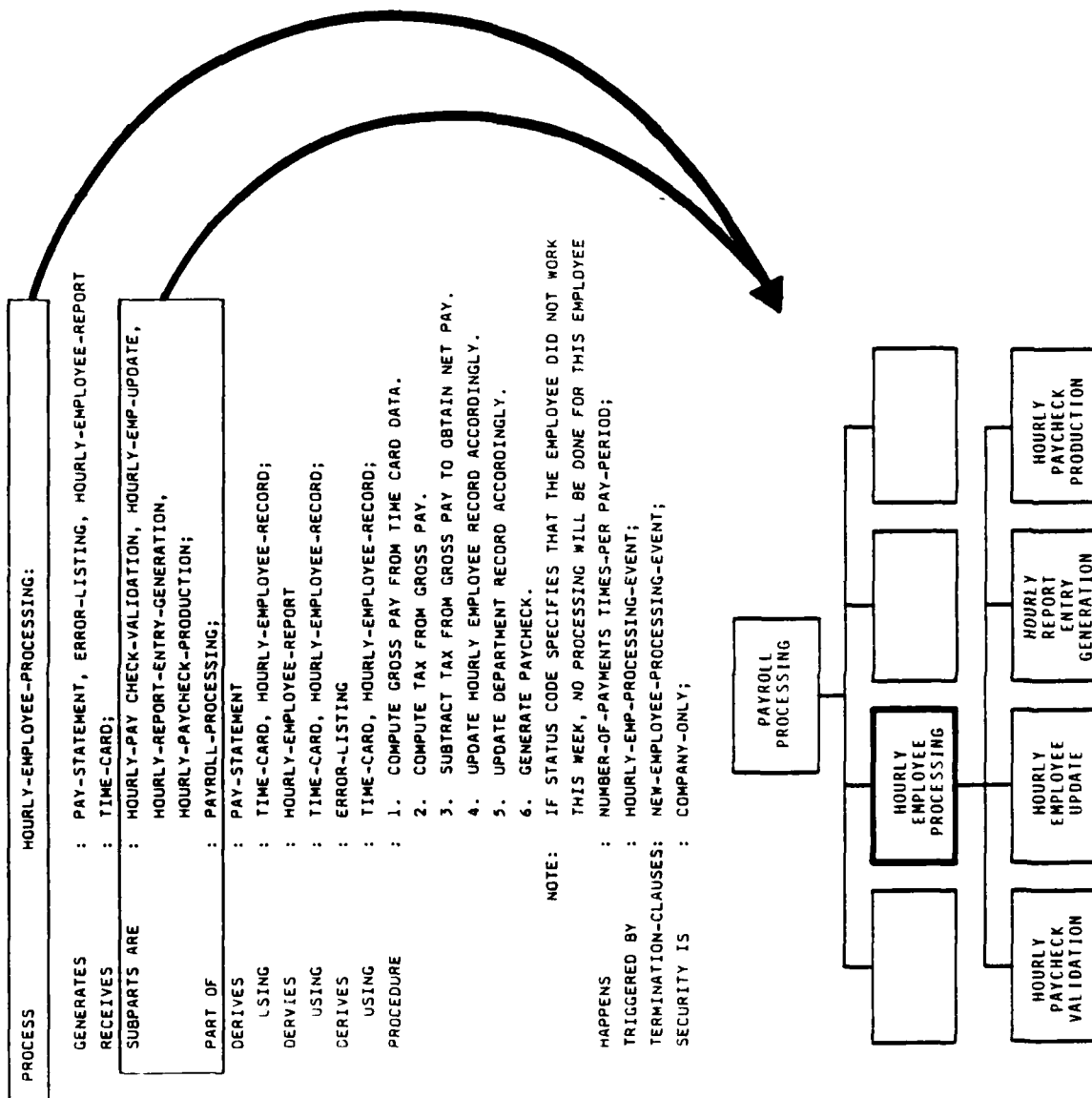
HAPPENS	: NUMBER-OF-PAYMENTS TIMES-PER PAY-PERIOD;
TRIGGERED BY	: HOURLY-EMP-PROCESSING-EVENT;
TERMINATION-CLAUSES	: NEW-EMPLOYEE-PROCESSING-EVENT;
SECURITY IS	: COMPANY-ONLY;



INSTRUCTOR NOTES

- POINT OUT THAT PARENT AND CHILDREN ARE DOCUMENTED IN EACH STATEMENT.

SYSTEM HIERARCHY



PROCEDURAL DETAILS

PROCESS HOURLY-EMPLOYEE-PROCESSING:

GENERATES : PAY-STATEMENT, ERROR-LISTING, HOURLY-EMPLOYEE-REPORT
 RECEIVES : TIME-CARD;
 SUBPARTS ARE : HOURLY-PAY CHECK-VALIDATION, HOURLY-EMP-UPDATE,
 HOURLY-REPORT-ENTRY-GENERATION,
 HOURLY-PAYCHECK-PRODUCTION;

PART OF : PAYROLL-PROCESSING;

DERIVES : PAY-STATEMENT

USING : TIME-CARD, HOURLY-EMPLOYEE-RECORD;

DERIVES : HOURLY-EMPLOYEE-REPORT

USING : TIME-CARD, HOURLY-EMPLOYEE-RECORD;

DERIVES : ERROR-LISTING

USING : TIME-CARD, HOURLY-EMPLOYEE-RECORD;

PROCEDURE : 1. COMPUTE GROSS PAY FROM TIME CARD DATA.

2. COMPUTE TAX FROM GROSS PAY.

3. SUBTRACT TAX FROM GROSS PAY TO OBTAIN NET PAY.

4. UPDATE HOURLY EMPLOYEE RECORD ACCORDINGLY.

5. UPDATE DEPARTMENT RECORD ACCORDINGLY.

6. GENERATE PAYCHECK.

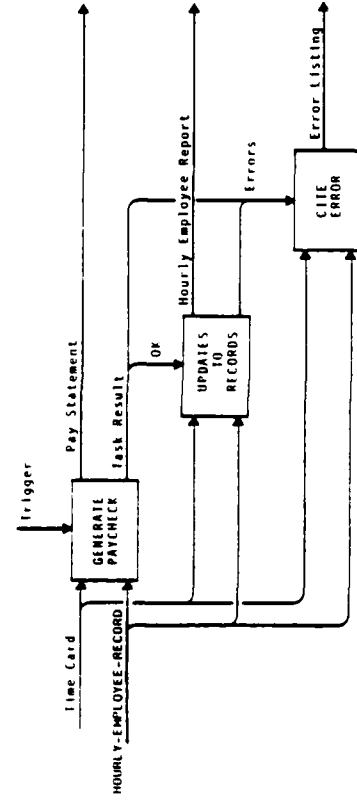
NOTE: IF STATUS CODE SPECIFICS THAT THE EMPLOYEE DID NOT WORK
 THIS WEEK, NO PROCESSING WILL BE DONE FOR THIS EMPLOYEE;

HAPPENS : NUMBER-OF-PAYMENTS TIMES-PER PAY-PERIOD;

TRIGGERED BY : HOURLY-EMP-PROCESSING-EVENT;

TERMINATION-CLAUSES: NEW-EMPLOYEE-PROCESSING-EVENT;

SECURITY IS : COMPANY-ONLY;



INSTRUCTOR NOTES

PSA IS THE "HEART" OF THE SYSTEM. OVER 50 REPORTS ARE AVAILABLE FROM WHICH TO ANALYZE THE SYSTEM.

PROBLEM STATEMENT ANALYZER (PSA) REPORT

- DATABASE MODIFICATION REPORTS
 - RECORD CHANGES THAT HAVE BEEN MADE
 - PROVIDE DIAGNOSTICS AND WARNINGS OF POSSIBLE DATA INCONSISTENCIES
- REFERENCE REPORTS
 - NAME LIST REPORT IDENTIFIES ALL OBJECTS, THEIR TYPE AND DATE OF LAST CHANGE
 - FORMATTED PROBLEM STATEMENT REPORT LISTS ALL PROPERTIES AND RELATIONSHIPS FOR A PARTICULAR OBJECT
 - DICTIONARY REPORT GIVES INFORMATION ABOUT DATA WITHIN THE SYSTEM IN A DATA DICTIONARY FORMAT
- SUMMARY REPORTS
 - DATABASE SUMMARY REPORT PROVIDES PROJECT MANAGEMENT INFORMATION
 - STRUCTURE REPORT SHOWS SYSTEM HIERARCHY
 - EXTENDED PICTURE REPORT SHOWS GRAPHICALLY DATA FLOW

INSTRUCTOR NOTES

VG 778.1

9-151

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000

PROBLEM STATEMENT ANALYZER (PSA) REPORT

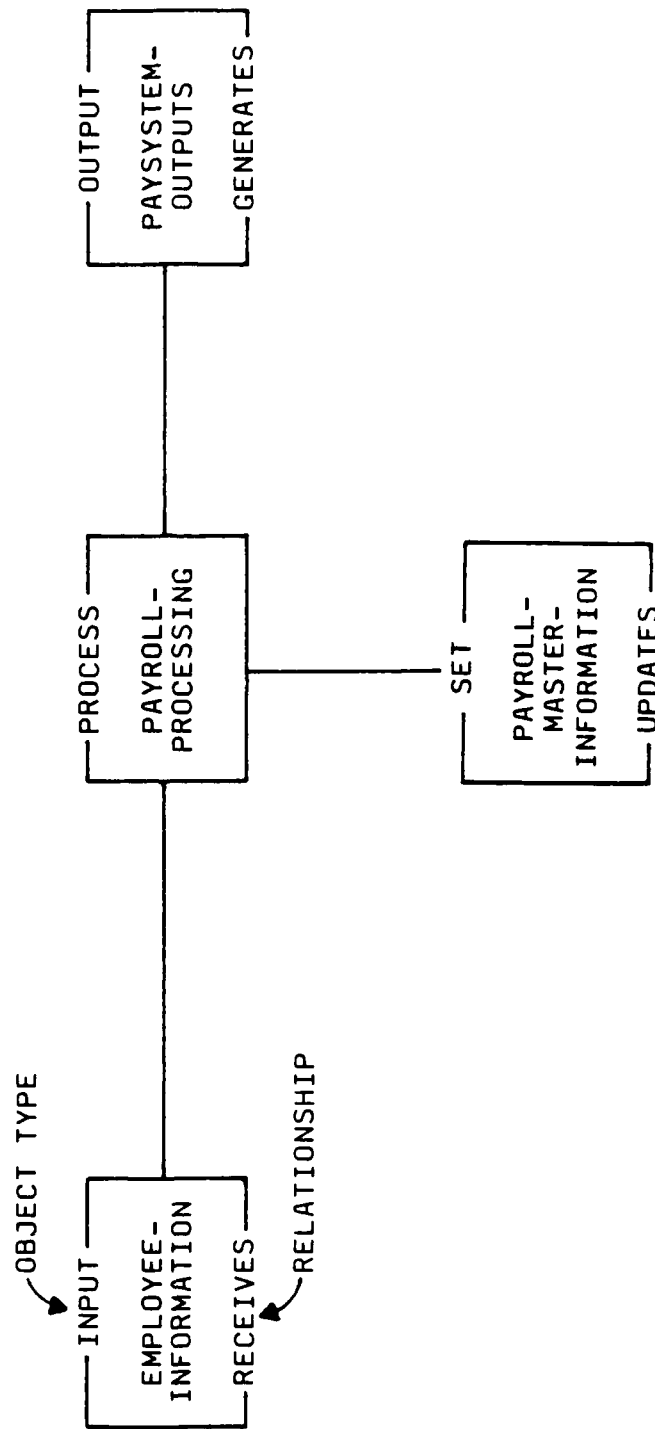
- ANALYSIS REPORTS
 - CONTENTS COMPARISON REPORT ANALYZES SIMILARITY OF INPUTS AND OUTPUTS
 - DATA PROCESS INTERACTION REPORT DETECTS GAPS IN INFORMATION FLOW OR UNUSED DATA OBJECTS
 - PROCESS CHAIN REPORT SHOWS THE DYNAMIC BEHAVIOR OF THE SYSTEM
- PSA CAN BE USED TO EXTRACT IN A SEMI-AUTOMATED WAY INFORMATION NEEDED FOR THE REQUIREMENTS SPECIFICATION DOCUMENTS

INSTRUCTOR NOTES

- "FLOW" RELATIONSHIPS GO HORIZONTALLY, I.E., LEFT TO RIGHT
- "STRUCTURE" OR "UPDATING" RELATIONSHIPS GO VERTICALLY, I.E., TOP BOTTOM
- PSL STATEMENT ARE IN CAPITAL LETTERS, NAMES OF OBJECTS APPEAR IN LOWER CASE
- EXAMPLE SHOWN IS ONE LEVEL ABOVE IN THE STRUCTURE HIERARCHY THAN THE PREVIOUS EXAMPLES. PAYROLL PROCESSING IS A PART OF PAYROLL MASTER INFORMATION.

A PSA EXAMPLE EXTENDED PICTURE REPORT

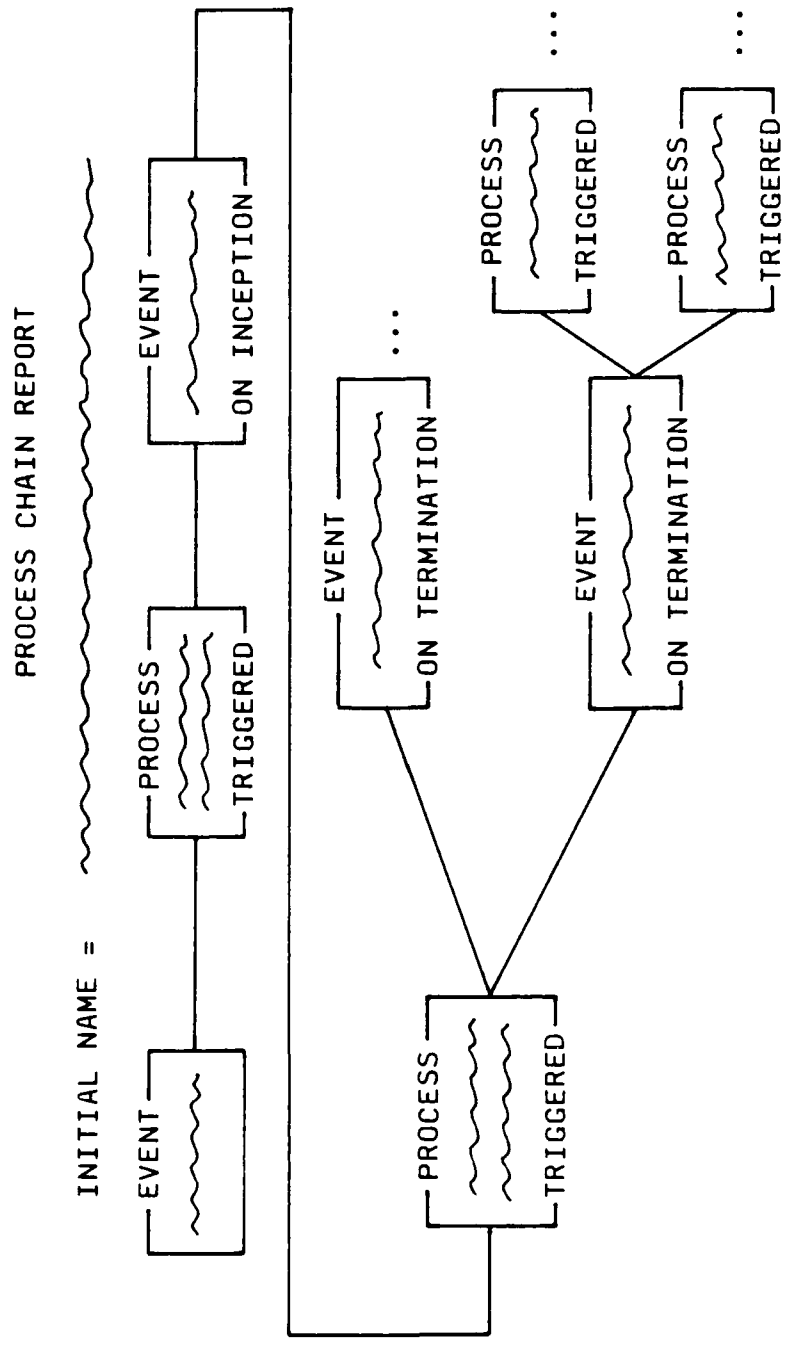
PAYROLL-PROCESSING



INSTRUCTOR NOTES

AN EXAMPLE OF A PROCESS CHAIN REPORT.

PSA REPORT



INSTRUCTOR NOTES

SUMMARIZES PSL/PSA'S STRENGTHS AND WEAKNESSES. CONTRAST TO SOME OF THE
PREVIOUS METHODS PRESENTED.

PSL/PSA SUMMARY

STRENGTHS

- HIGH QUALITY SYSTEM REQUIREMENTS
- FACILITATE COMMUNICATION
- AUTOMATED CHECKS
- REDUCE DESIGN, DEVELOPMENT AND TESTING COST
- EFFECTIVE DOCUMENTATION
- RESOURCE ESTIMATION

WEAKNESSES

- COMPLEXITY OF SYNTAX AND STRUCTURE
- LARGE NUMBER OF RESERVED WORDS
- EXPENSIVE TO OPERATE BOTH IN PEOPLE AND INSTALLATION

INSTRUCTOR NOTES

THEME: OTHER GRAPHICAL ANALYSIS METHODS THAT ARE BASED ON DATA FLOW THROUGH THE SYSTEM.

PURPOSE: TO ROUND OUT THE ANALYSIS METHODS BY INCLUDING SOME OF THE MORE POPULAR METHODS USED IN BUSINESS APPLICATIONS.

REFERENCES: 1. GANE, C., SARSON, T., "STRUCTURED ANALYSIS TOOLS AND TECHNIQUES"
IMPROVED SYSTEMS TECHNOLOGY, NYC; 1977.

2. DEMARCO, T., "STRUCTURED ANALYSIS AND SYSTEM SPECIFICATIONS"
PRENTICE-HALL, NJ; 1978.

Section 10

STRUCTURED SYSTEMS ANALYSIS METHODS

INSTRUCTOR NOTES

MENTION THAT THESE METHODS EXPRESS "WHAT" THE SYSTEM IS REQUIRED TO DO, WHILE NOT MAKING ANY COMMITMENT AS TO "HOW" THE SYSTEM SHOULD PHYSICALLY BE IMPLEMENTED.

OVERVIEW

- STRUCTURED SYSTEMS ANALYSIS METHODS ...

- ARE A COLLECTION OF TECHNIQUES
- ARE SPINOFFS FROM STRUCTURED DESIGN (WE WILL SEE LATER)
- WERE PIONEERED BY DEMARCO AND GANE AND SARSON
- PUSH THE BUBBLE CHART MODELING TOOL INTO THE ANALYSIS PHASE

- ALL SSA TECHNIQUES ...

- HAVE A GRAPHIC NOTATION
- RESULT IN MODELS
- ARE LIMITED IN EXPRESSING DECOMPOSITION AND HIERARCHY
- EMPHASIZE KEEPING A DICTIONARY OF TERMINOLOGY

INSTRUCTOR NOTES

TELL CLASS TO REMEMBER BACK TO SADT AND HOW THESE ARE SIMILAR OR DIFFERENT. FOR
EXAMPLE, SADT BOXES HAVE SIDES WHICH MEAN SOMETHING SPECIFIC.

OVERVIEW

- THE SSA TECHNIQUES WE WILL COVER ARE ...

- DEMARCO DATA FLOW DIAGRAMS

- GANE AND SARSON DATA FLOW DIAGRAMS

- DATA DICTIONARIES

DEMARCO DATA FLOW DIAGRAMS

- DEMARCO PICTURES ...

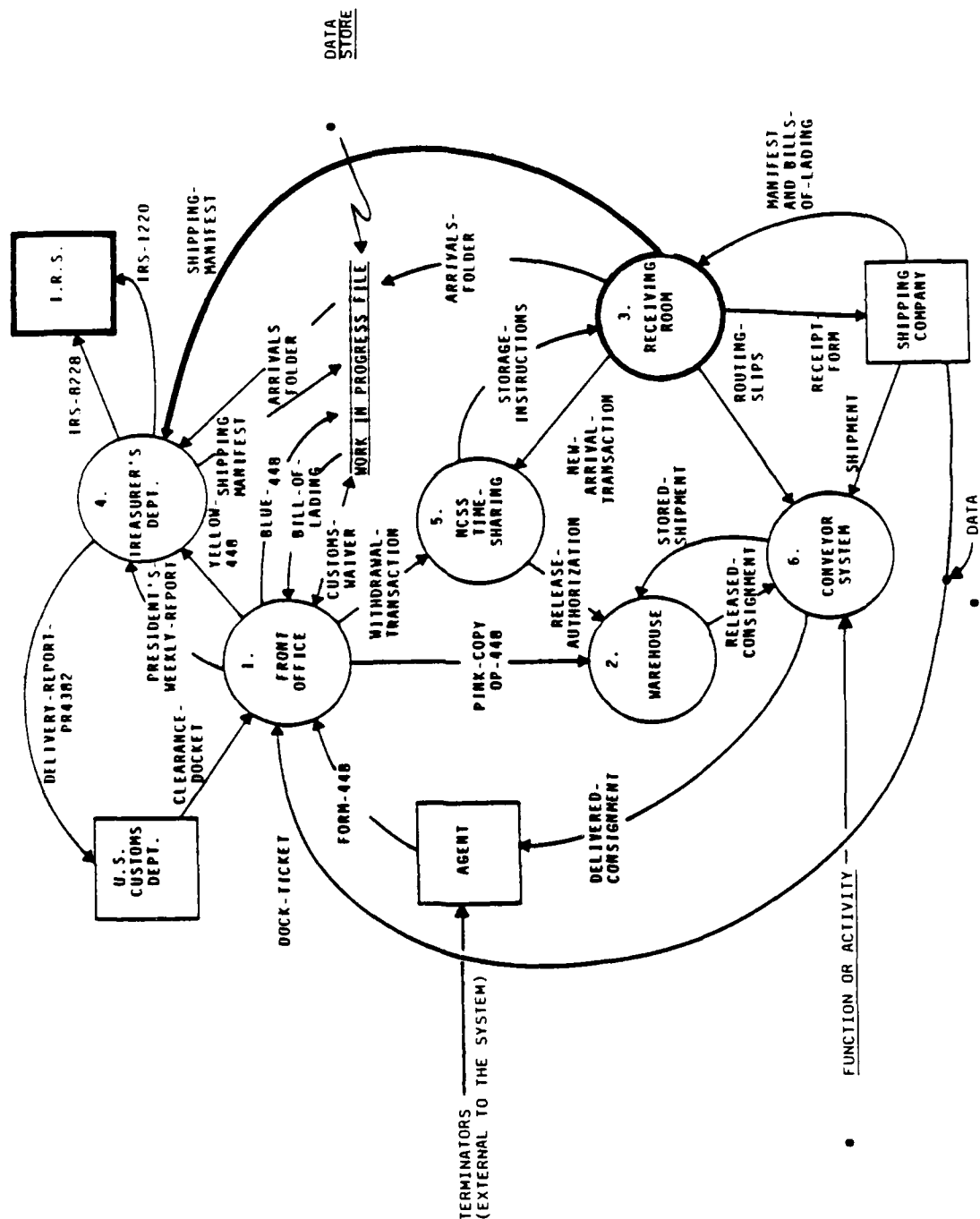
- USE THE SAME SYNTAX AS THE BUBBLE CHARTS OF STRUCTURED DESIGN (WE WILL SEE IN A LATER SECTION)
- EMPHASIZE ONLY DATA FLOWS, SOURCES AND SINKS
- ARE FLAT DIAGRAMS (HAVING NO HIERARCHY OF DIAGRAMS)*
- USE TERMINOLOGY DEFINED IN A DATA DICTIONARY

*EXTENSIONS TO THE TECHNIQUES SUPPORT HIERARCHY (YOURDON ANALYSIS)

INSTRUCTOR NOTES

- THE SQUARE IS AN EXTERNAL SYMBOL. IT IS THE SOURCE AND/OR DESTINATION OF DATA OUTSIDE THE SYSTEM.
- THE ARROW IS THE DIRECTION OF DATA FLOW. IT'S THE PATHWAY ALONG WHICH DATA MOVES.
- THE "BUBBLE" (CIRCLE) IS THE PROCESS SYMBOL. IT'S JUST A FUNCTION WHICH TRANSFORMS THE DATA.
- THE PARALLEL LINES ARE THE DATA STORE SYMBOL. IT'S THE PLACE IN THE SYSTEM WHERE DATA IS STORED IN SOME WAY (IT MAY BE PEOPLE, CARDS, DISKS, BOOKS, ETC.).

DEMARCO DATA FLOW DIAGRAM



INSTRUCTOR NOTES

NOTE THE NUMBER OF ITERATIONS INVOLVED.

VG 778.1

10-5i

INSTRUCTOR NOTES

A LOGICAL DATA DICTIONARY IS A PLACE WHERE ALL DETAILED DEFINITION OF DATA OBJECTS ARE STORED.

VG 778.1

10-6i

DEMARCO DATA DICTIONARY

- A DATA DICTIONARY DOCUMENTS IN A RIGOROUS WAY
THE STRUCTURE OF DATA FLOWS IN THE DIAGRAMS

- DEMARCO DATA DICTIONARY SYMBOLS:

= HIERARCHY (COMPRISES)

+ SEQUENCE (AND)

[] SELECTION

{ } REPETITION

() OPTIONAL

INSTRUCTOR NOTES

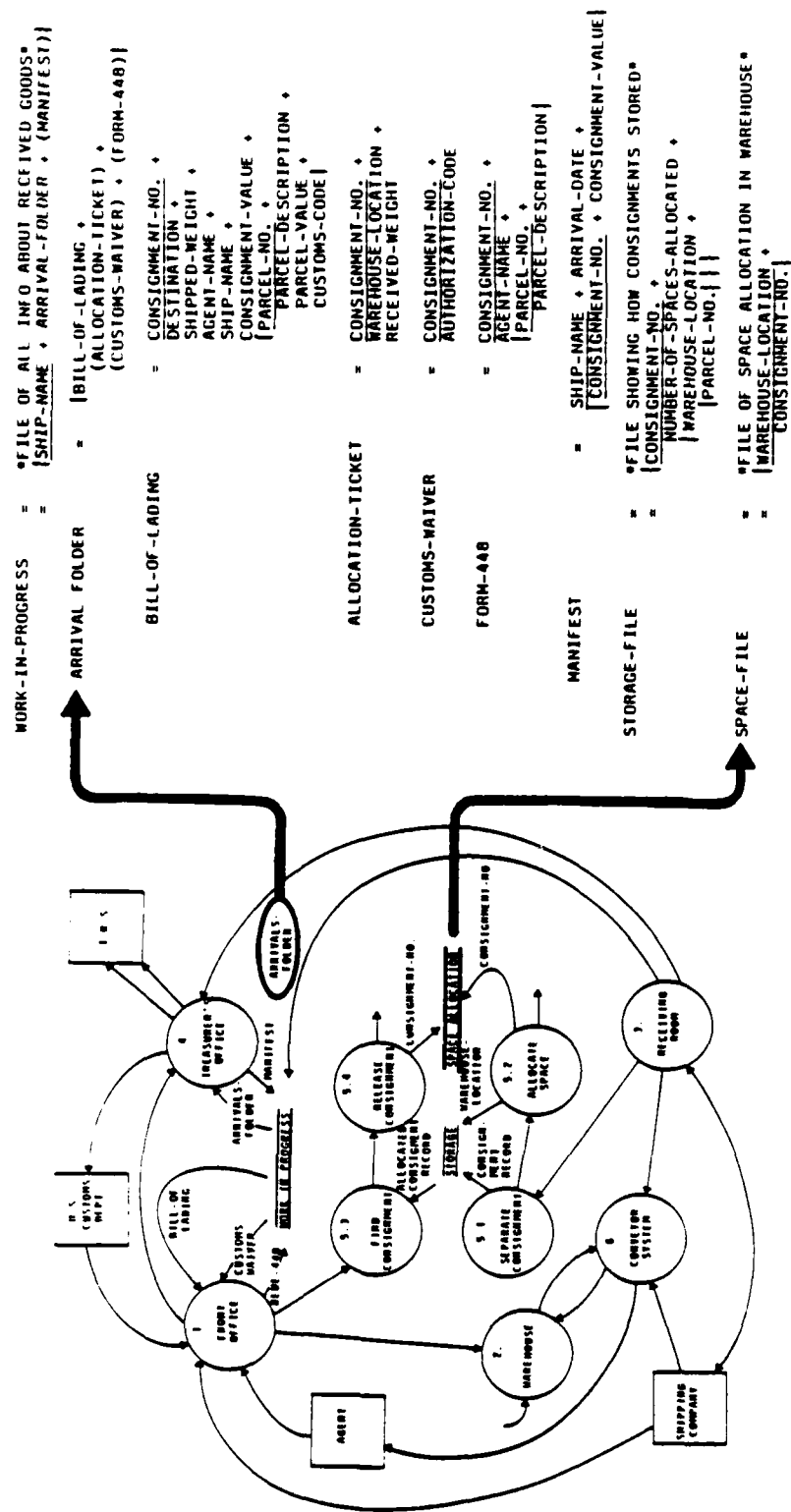
ASK CLASS HOW PSL/PSA COULD MODEL THESE RELATIONSHIPS.

VG 778.1

10-7i

DEMARCO

DATA FLOW DIAGRAM



INSTRUCTOR NOTES

- ONLY REAL DIFFERENCE BETWEEN DEMARCO AND GANE AND SARSON IS THE "SQUARE BUBBLE"
- TOP, BOTTOM AND SIDES OF "SQUARE BUBBLE" HAVE NO SIGNIFICANCE (CONTRAST TO SADT)
- HIERARCHY OF DIAGRAMS NOT TIED TOGETHER (NEEDS AN SADT-LIKE NOTATION)

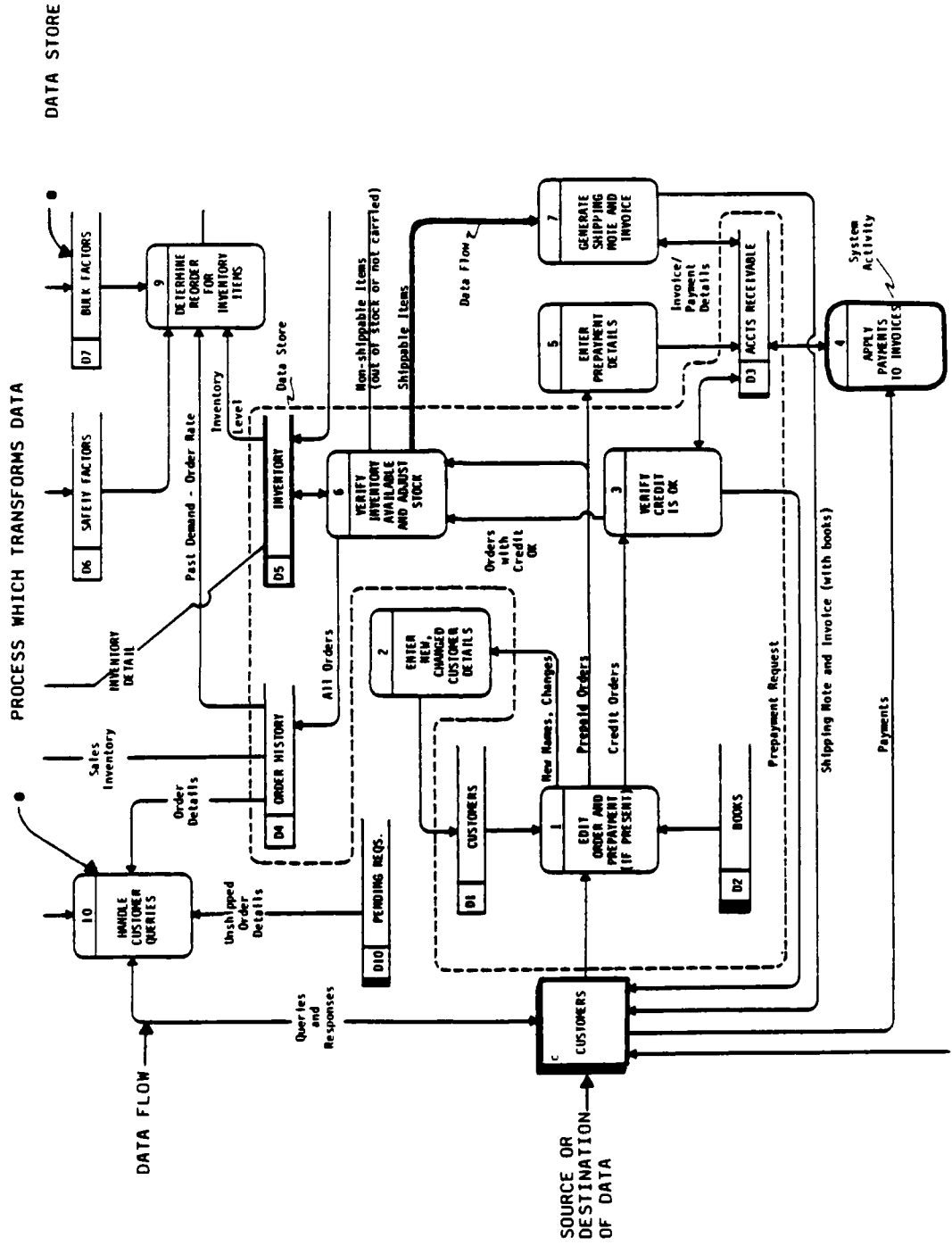
GA NE AND SARSON

- GANE AND SARSON PICTURES ...

- USE BUBBLE CHART SYNTAX, BUT BUBBLES ARE "SQUARE"
- EMPHASIZE ONLY DATA FLOWS, SOURCES AND SINKS
- ARE HIERARCHIC, BUT LACKS DIAGRAM-TO-DIAGRAM INTERFACES
- USE DEFINED TERMINOLOGY

- GANE AND SARSON DATA DICTIONARY ENTRIES ARE JUST LISTS OF ATTRIBUTES

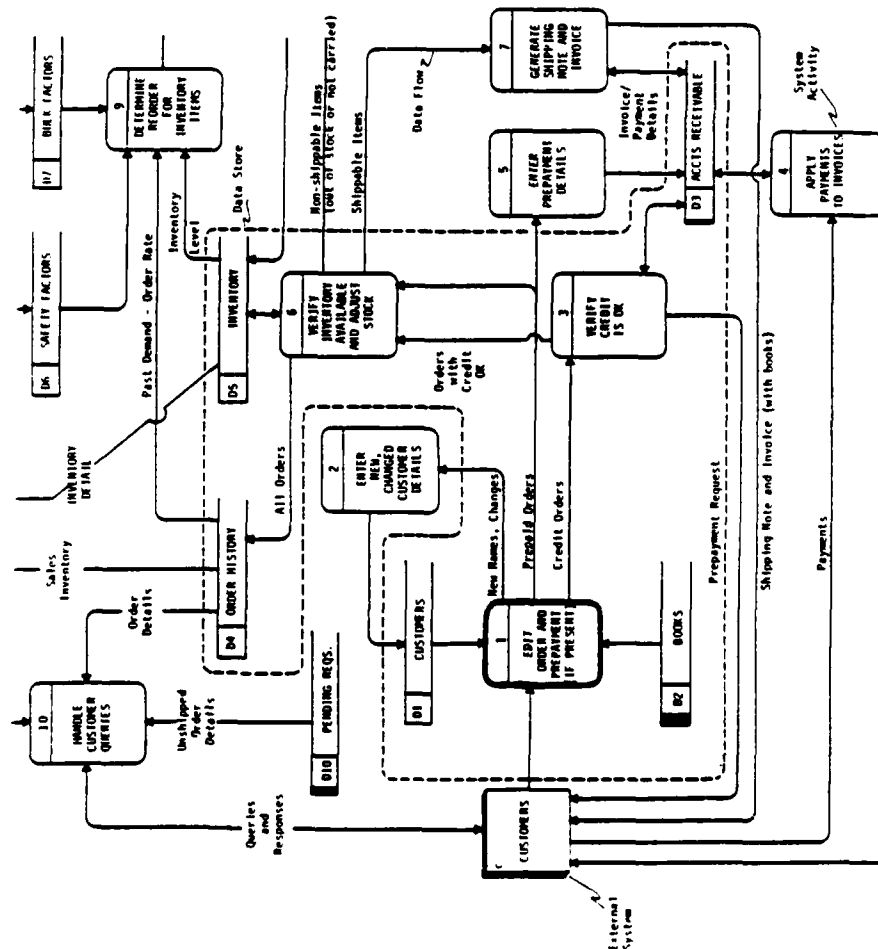
ANE AND SARSON DATA FLOW DIAGRAMS



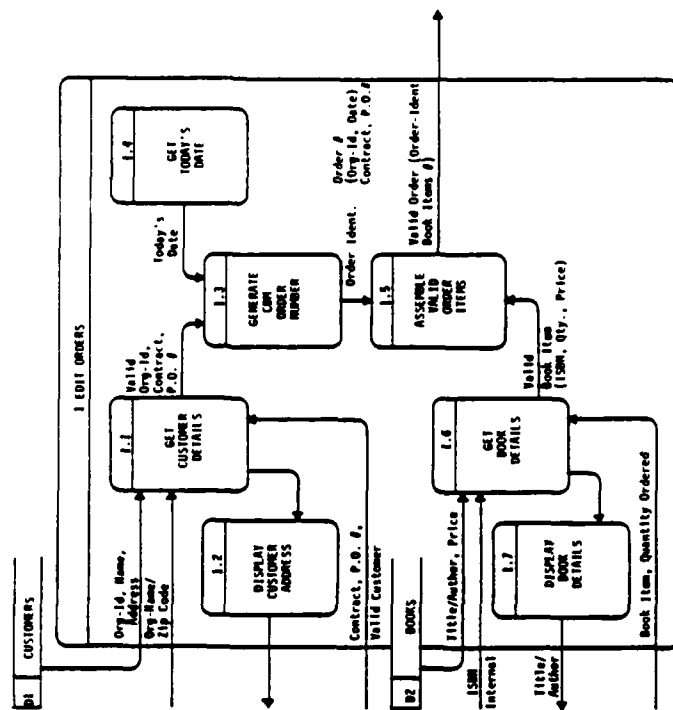
GANE AND SARSON

• HOW A "SQUARE BUBBLE" IS DECOMPOSED ...

PARENT DIAGRAM



PICTURE OF BUBBLE 1



INSTRUCTOR NOTES

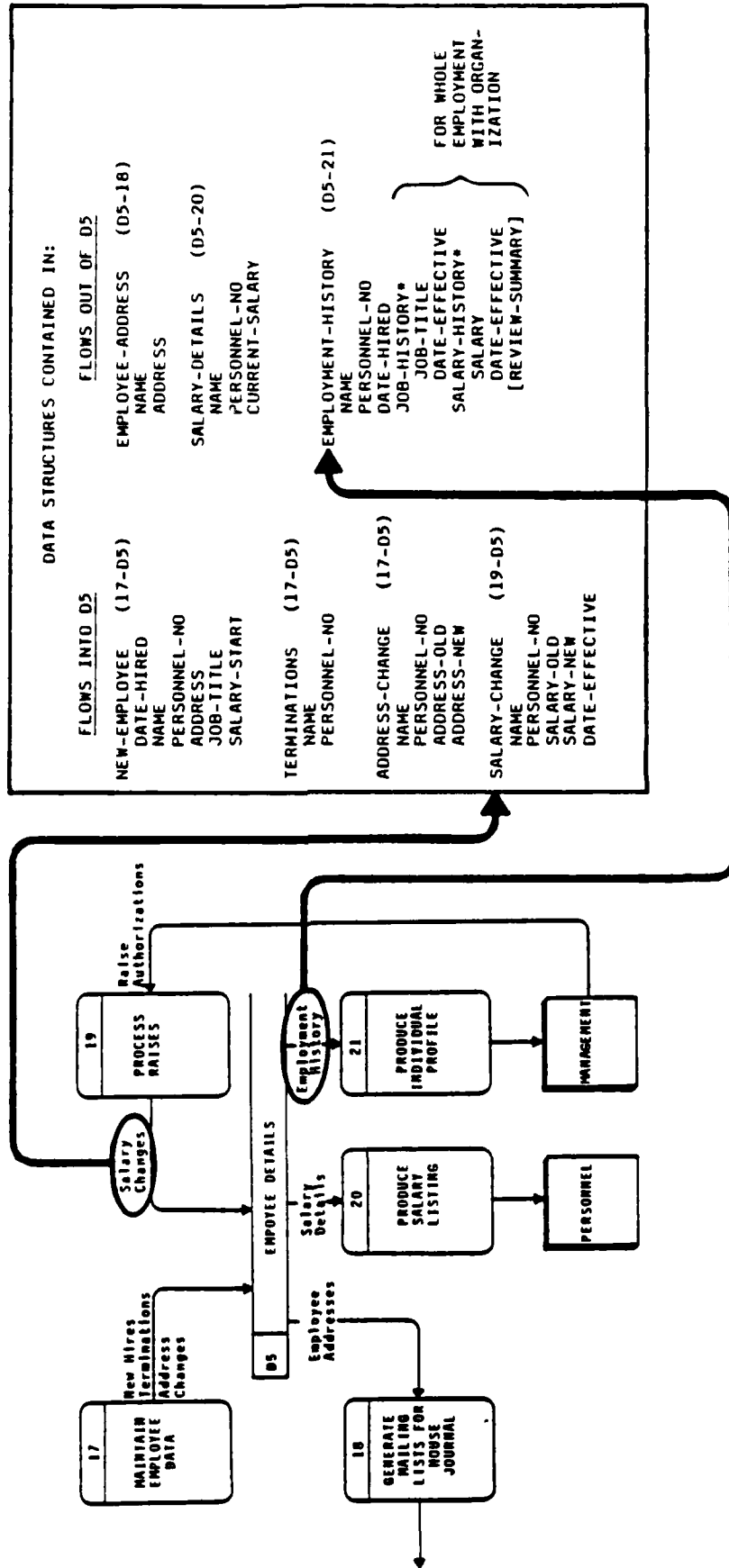
SUMMARIZE THE TWO TECHNIQUES AND ASK CLASS WHICH THEY WOULD RATHER USE: SADT, PSL/PSA,
SA, GANE AND SARSON, ENTITY DIAGRAMMING.

ASK WHY?

GANE AND SARSON DATA DICTIONARY

DATA FLOW

DATA DESCRIPTION



Material: Software Engr'g Methodologies (M201), Volume 1

We would appreciate your comments on this material and would like you to complete this brief questionnaire. The completed questionnaire should be forwarded to the address on the back of this page. Thank you in advance for your time and effort.

1. Your name, company or affiliation, address and phone number.

2. Was the material accurate and technically correct?

Yes ☐

No ☐

Comments:

3. Were there any typographical errors?

Yes ☐

No ☐

If yes, on what pages?

4. Was the material organized and presented appropriately for your applications?

Yes ☐

No ☐

Comments:

5. General Comments:

place
stamp
here

COMMANDER
US ARMY MATERIEL COMMAND
ATTN: AMCDE-SB (OGLESBY)
5001 EISENHOWER AVENUE
ALEXANDRIA, VIRGINIA 22233

END
DTIC
FILMED
4-86