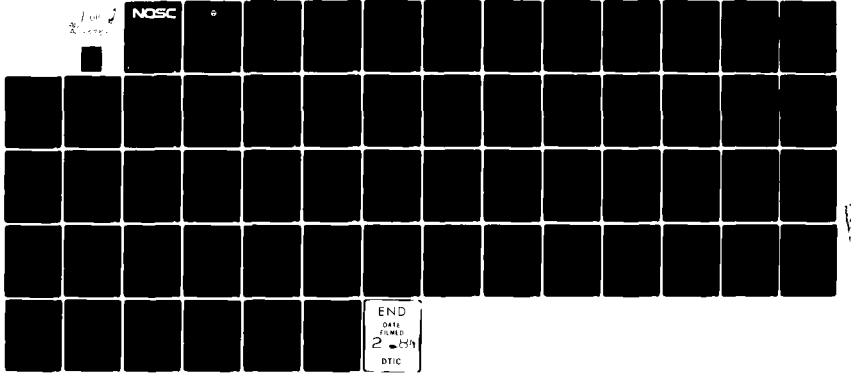
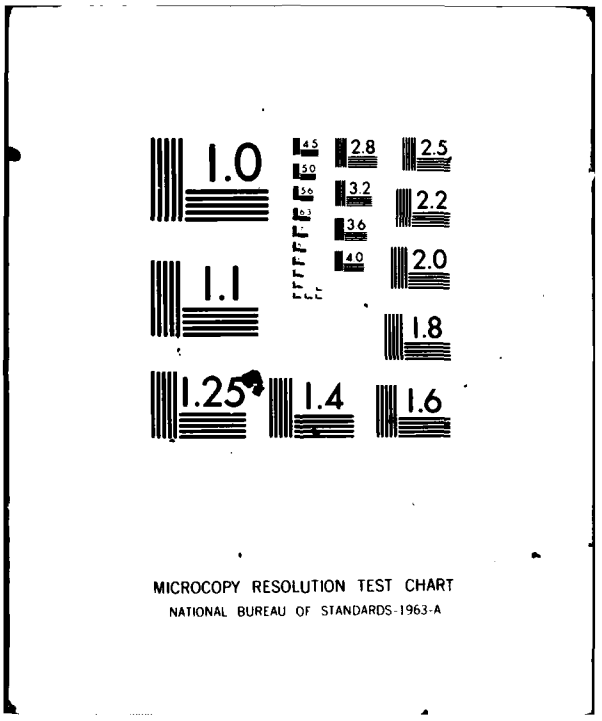
AD-A093 766 NAVAL OCEAN SYSTEMS CENTER SAN DIEGO CA F/G 17/3
EXPERIMENTAL TESTS OF PTAPS PERFORMANCE IN THREE TYPES OF PRODU--ETC(U)
SEP 80 R A DILLARD

UNCLASSIFIED NOSC/TD-385 NL



END
DATE FILMED
2-81
DTIC



LEVEL II

(12)

NOSC

NOSC TD 385

NOSC TD 385

AD A093766

Technical Document 385

EXPERIMENTAL TESTS OF PTAPS PERFORMANCE IN THREE TYPES OF PRODUCTION SYSTEM STRUCTURES

An Investigation of the Compatibility of a Platform-Track Association Production Subsystem with STAMMER1, STAMMER2, and ROSIE

Robin A. Dillard

17 September 1980

Prepared for
Naval Electronic Systems Command
(ELEX 330)
Washington DC 20360

DDC FILE COPY

Approved for public release; distribution unlimited

DTIC
ELECTE
JAN 14 1981
S D
A

NAVAL OCEAN SYSTEMS CENTER
SAN DIEGO, CALIFORNIA 92152

81 1 14 001

NAVAL OCEAN SYSTEMS CENTER, SAN DIEGO, CA 92152

AN ACTIVITY OF THE NAVAL MATERIAL COMMAND

SL GUILLE, CAPT, USN
Commander

HL BLOOD
Technical Director

ADMINISTRATIVE INFORMATION

The work was performed by members of the C^2 Information Processing Branch, Tactical Command and Control Division, Command Control – Electronic Warfare Systems and Technology Department under element 61153N, project XR01408, subproject XR0140801. This document was released for publication 2 October 1980.

Released by
RC Kolb, Head
Tactical Command and
Control Division

Under authority of
JH Maynard, Head
Command Control – Electronic Warfare
Systems and Technology Department

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: NOSC Technical Document 385 (TD 385)
2. GOVT ACCESSION NO.: AD-A093 766
3. RECIPIENT'S CATALOG NUMBER: 9

4. TITLE (and Subtitle): EXPERIMENTAL TESTS OF PTAPS PERFORMANCE IN THREE TYPES OF PRODUCTION SYSTEM STRUCTURES • An Investigation of the Compatibility of a Platform-Track Association Production Subsystem with STAMMER 1, STAMMER 2, and ROSIE
5. TYPE OF REPORT & PERIOD COVERED: Interim report, FY80
6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s): Robin A. Dillard
8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS: Naval Ocean Systems Center, San Diego, CA 92152
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61153N, XR01408, XR0140801

11. CONTROLLING OFFICE NAME AND ADDRESS: Naval Electronic Systems Command (ELEX 330), Washington, DC 20360
12. REPORT DATE: 17 September 1980
13. NUMBER OF PAGES: 54

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): 59
15. SECURITY CLASS. (of this report): Unclassified
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution is unlimited

NOSC/TD-385

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Artificial intelligence
Production systems
Tactical Situation Assessment (TSA)
Platform-Track Association Production Subsystem (PTAPS)
STAMMER 1
STAMMER 2
ROSIE

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The capability of a production system applied to Tactical System Assessment (TSA) is extended by adding a "package" of system-logic rules, the Platform-Track Association Production Subsystem (PTAPS). PTAPS performs much of the logical reasoning needed to match tracks to specific platforms. This document illustrates the performance of PTAPS rules in three very different production system structures and discusses the efforts required to combine PTAPS rules compatibly with a much broader set of TSA rules. The PTAPS experiments described were run in INTERLISP programs on ARPANET hosts.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

393 159

TABLE OF CONTENTS

1. INTRODUCTION

In earlier work[1], a method was developed of extending the capability of a production* system applied to tactical situation assessment (TSA) by adding a "package" of system-logic rules. The implementation of these rules within such a production system was termed a Platform-Track Association Production Subsystem (PTAPS). The function of PTAPS is to perform much of the logical reasoning, such as process-of-elimination reasoning, needed to match tracks to specific platforms. (A "track" represents positional and other sensor-derived information about a platform.) This document illustrates the performance of PTAPS rules in three very different production system structures and discusses the effort required to combine PTAPS rules compatibly with a much broader set of TSA rules. The PTAPS experiments were run in Interlisp programs on ARPANET hosts.

This effort has been just one phase of a larger research effort to develop automated data-fusion techniques. The automation of data fusion will require the integration of many interacting subprocesses[2], and the automation of various constituent fusion functions may be practicable with production system technology. Another major applicable technology being investigated under the larger effort is natural language processing[3] and its interface with production systems[4]. As part of this latter work, an experimental program is being written which uses the formatted parts of tactical messages to interpret pertinent narrative parts, and restructures the information for input to a production system.

---

*A "production" is an if-then rule implemented in a "production system," a system also having a data base and a number of control mechanisms.

---

1. NOSC TD 288, Higher Order Logic for Platform Identification in a Production System, by R. A. Dillard, 17 October 1979.
2. NOSC TR 364, New Methodologies for Automated Data Fusion, by R. A. Dillard, September 1978.
3. NOSC TD 324, Natural Language Processing Applied to Navy Tactical Messages, by Davis M. Keirsey (Systems Development Corporation), February 1980.
4. Dillard, R. A., Text-Understanding Techniques Applied to Partly Formatted Navy Tactical Messages, NOSC TD, in preparation.

2. PTAPS OVERVIEW

Many of the PTAPS rules have the sole function of building into the data base an "intermediate framework" of membership files which permit, via other rules, chains of reasoning not otherwise possible. This framework includes many kinds of "track files" and "platform files." To become a member of some track file or platform file, a track or platform must satisfy the conditions of a certain membership rule, and a member is removed by another rule when the original conditions are no longer all satisfied. Of particular importance are "OR-files." The members of the OR-file of a platform are those tracks which have not been ruled out as the track of that platform. A platform is a member of a track's OR-file if that track has not been ruled out as a track of that platform. The OR-file of an emission has, as members, platforms which have not been ruled out as the emitting platform.

Reference 1 describes the various kinds of files and other underlying concepts, such as "impossible relationships," and lists many PTAPS rules in addition to the ones used in the experiments shown in this document. Brief descriptions of the files and other constructions are also contained in the explanation parts of the first demonstration shown in appendix I.

Some of the rules needed to support the chains of logical reasoning in PTAPS are also individually useful in an unextended system, and some of these require routine but extensive geometry calculations. Most of the latter were omitted from the experiments, and the data they would have contributed were entered instead as data from subsystems. The geometry functions involved in evaluating the conditions of the omitted rules could be implemented without difficulty, but would increase execution time while not serving a purpose relative to the intent of the investigations.

3. PTAPS EXPERIMENTS

3.1 PTAPS IN A MODIFIED STAMMER1

Proof-of-concept experiments with PTAPS rules were conducted in FY 79 in a modified version of STAMMER, a System for Tactical Assessment of Multisource Messages, Even Radar[5]. STAMMER was developed to serve as a demonstration of the applicability of rule-based inference technique to the problem of tactical situation assessment, and was initially applied to the specific problem of distinguishing merchants from other platforms by using radar and external messages. Because of the later introduction of a revised version of STAMMER, the original system is referred to here as STAMMER1. A small, fast skeleton version of STAMMER1 was created for PTAPS experiments by stripping the original of its confidence mechanisms, explanation functions, and graphics interface.

STAMMER1 receives and stores data as two-node assertions of the form (node-A relation node-B), or, when the relation is "is a," of the form (node-A node-B). For example, the assertions (P1 PLATFORM) and (DELTA CLASS P1) represent the knowledge that P1 is a platform and that Delta is the class of P1.

The experiments involved two basic scenarios: one concerned with the identification of submarines, and the other with the identification of members of a Soviet task group with the help of satellite reconnaissance data. Special LISP functions were loaded with the program to explain the scenario and the principles of PTAPS during a demonstration. Typescripts of demonstrations of the two scenarios are given in sections I.2 and I.3 of appendix I. (A summary table was inserted at the end of each typescript.) The PTAPS rules used in the two experiments are shown in STAMMER1 syntax in section I.4. An asterisk indicates that the binding of the variable so marked occurs upon the successful evaluation of that condition of the rule.

3.2 PTAPS IN STAMMER2

STAMMER2 is a revised version of the system described in section 3.1, and is described in reference 6. STAMMER2 differs from the original STAMMER in a number of ways. Most importantly, the efficiency of the system was greatly improved by organizing the data into "streams," which simulate parallel processing and permit "automatic" suspension and resumption of processes. Under this approach, which may be described as "incremental deduction," whenever a

---

5. NOSC TD 252, STAMMER: System for Tactical Assessment of Multisource Messages, Even Radar, by R. J. Bechtel and P. H. Morris (Systems Development Corporation), May 1979.
6. NOSC TD 298, Vol 1 and 2, STAMMER 2: A Production System for Tactical Situation Assessment, by D. C. McCall (NOSC), P. H. Morris, D. F. Kibler, and R. J. Bechtel (Systems Development Corporation), October 1979.

condition of a rule fails, a "suspension" is created that corresponds to the remainder of the rule. Even when a condition succeeds, if there are other ways for it to be satisfied, a suspension is left behind. Another change is the simpler formatting of rule conditions and action. In addition, the components of the assertions are in a different order than in STAMMER1: (relation node-A node-B).

A typescript of the two-submarine scenario run in STAMMER2 is given in appendix II. The rules appear in STAMMER2 syntax within the typescript. In the rules, nodes which are variables in the assertions are prefixed by asterisks. The binding of a variable occurs upon evaluation of the first condition containing it. The actions of the rules used in the experiments are asserted with confidence +1.0, with the exception of the "ORFILEREDUC" rule, whose actions are asserted with the confidence -1.0.

Computational functions, known as "oracles," are treated in the writing of STAMMER2 rule conditions in much the same way as relations: (oracle argument-1 ... argument-n). Two oracles not already in the STAMMER2 program were needed for this application, so were loaded while still at the LISP level. The oracle ORFNUM provides numbers prefixed by F for labeling OR-files. The oracle MEMBERCOUNT1 determines whether a file has exactly one member.

The initial information representing a "snapshot view" of the data base at the time of the first detection was loaded as the first message. Although not needed to exercise PTAPS rules, formatted messages giving track positions also were entered, which made possible an accurate graphical display of the Persian Gulf and the relative positions of the tracks.

3.3 PTAPS IN ROSIE

The PTAPS rules needed in the two-submarine scenario were also implemented in ROSIE (A Rule-Oriented System for Implementing Expertise), a system under development by the Rand Corporation. The version of ROSIE used is now referred to as ROSIE-1, since a new design, ROSIE-2, is being implemented. The specifications for ROSIE-1 are published in reference 7.

There is a major difference in data representation between ROSIE-1 and the STAMMER systems. STAMMER allows the same attribute (ie, relation) of a node to have any number of node values; eg, the data base can contain the assertion (relation node-A node-B1) and also the assertion (relation node-A node-B2). ROSIE-1 constrains an attribute to a single value, but allows a "list-value" via the action: PUT (value) INTO (attribute) OF (name). In the PTAPS application, therefore, membership in a track or platform file is represented by a list value in ROSIE-1 and by multiple values of the attribute (ie, relation) "member" in the STAMMER systems. (At this stage of its development, ROSIE-2 permits multiple values of an attribute, so an assertional data structuring equivalent to that in STAMMER could be used in ROSIE-2.) The most noticeable

---

7. RAND Corporation report N-1158-1-ARPA, Design for a Rule-Oriented System for Implementing Expertise, by D. A. Waterman, R. H. Anderson, F. Hayes-Roth, P. Klahr, G. Martins, and S. J. Rosenscheim, May 1979.

difference between the ROSIE and the STAMMER systems is in the rule syntax — ROSIE rules are written in an English-like syntax while STAMMER rules are coded in statements involving two-node assertions.

A typescript of PTAPS rules run in ROSIE-1 is given in appendix III. For convenience, the "snapshot" background data were entered with the rules. (They were entered as the first message in the STAMMER experiments.) The messages were typed in, although they could have been entered from a file. The PTAPS application of ROSIE uses its tracing and explanation facility but does not exploit some of its other features, such as property inheritance, subroutine rule sets, and pattern matching.

4. CONCLUSIONS

Reference 1 discusses the additional kinds of rules and capabilities that must be included in an operational PTAPS and the problems involved in integrating PTAPS rules into an actual tactical situation assessment (TSA) system. None of these conclusions has changed, but the problems of integration will be reviewed and discussed further here. A general conclusion reached from the recent investigations of different production system structures is that PTAPS rules should work in any system in which conventional TSA rules will work.

A problem mentioned in reference 1 is the need for uniformly representing tracks and platforms throughout the integrated system. The default ruleset and the default memory in STAMMER2 use simple semantic net structuring to represent platforms and "sightings" of platforms. The concept of a "track" is not used because it is not needed for that set of TSA rules. To represent each component of what would be a track, the assertions (SIGHTING (some label) SIGHTINGi), (SOURCE SIGHTINGi RADAR), (TIME SIGHTINGi 945), etc, are used, where (some label) is the plaform name, when known, and otherwise is an arbitrary label such as CONTACT3 or REDB. By making (some label) a track label, say T00059, and then asserting that T00059 is a track and, if the platform is known, also asserting that T00059 is a track of the respective platform node (eg, P00392), the concept of a track would be made consistent with PTAPS rules.

Making STAMMER terminology concerning platforms consistent with PTAPS rules would require simpler but more extensive changes, primarily in the retyping of the memory (the initial data base). The current convention is, for example: (PLATFORM PROVORNY), (CLASS PROVORNY KASHIN), (TYPE PROVORNY DESTROYER), (ID PROVORNY HOSTILE), etc. For compatibility with PTAPS, a platform node name would be the primary label: (PLATFORM P00891) (NAME P00891 PROVORNY), (CLASS P00891 KASHIN), etc. Some of the conditions of several rules which involve attributes of known platforms would also have to be rewritten. The only needed change to STAMMER2 itself (since any ruleset, memory, or messagefile can be loaded) would be in the assignment of a track node and its association with a known platform (only when known) upon message receipt, although, alternatively, messages which provide the necessary data and label could be typed in from the terminal.

The most difficult problem with compatibility in STAMMER2 concerns the handling of confidence values. PTAPS does not use confidence values and must be constrained from operating on assertions (put in the data base by the actions of TSA rules) that have less than a near-certainty confidence value. There are several ways of doing this but all would require at least a slight change in STAMMER2. For temporary experimental integration, however, an additional oracle (computational function -- see section 3.2) which returns the confidence value of an assertion could be loaded while still at the LISP level, and the PTAPS rules could include conditions which use this oracle and compare the returned value with a threshold. Whatever method is used would be applied also to assertions given negative near-certainty confidence; eg, if TSA rules determine that the type of some track is certainly not merchant, then a PTAPS rule would assert it to be an impossible track of each platform whose type is merchant.

In discussions regarding confidence values, reference 1 describes how conclusions which would logically follow from different assumptions about particular tracks or platforms could be determined by PTAPS and assigned confidence values based on the confidence values of the initial data. Implementing this would not be an easy task.

Under an exploratory development program, many of the TSA rules run in STAMMER2 were also run in ROSIE-1. The TSA rules and the data base were structured in a way completely compatible with PTAPS rules. ROSIE-1 does not have a mechanism for computing confidences, but one could be implemented by means of a subroutine ruleset, which would work with both kinds of rules. Because ROSIE-1 will soon be replaced by ROSIE-2, no attempt was made to integrate the two; ie, to run TSA rules with PTAPS rules which would appropriately interact.

The logical reasoning that can be implemented with PTAPS rules is essential to the function of associating tracks with platforms. If the other reasoning functions of tactical situation assessment are to be performed in a production system, then probably the PTAPS function also should be performed within that system, so that the functions can be easily coordinated and can share the data base. A possible alternative would be to create a specialized problem-solving technique for platform-track association and interface it with the production system, but in such a case, coordination and data base sharing would be more difficult.

The next desirable step in continuing PTAPS investigations is to integrate experimentally PTAPS rules with other TSA rules in a production system. Unfortunately, current production systems such as STAMMER2 and ROSIE-1 are inadequate for this large an application. Of the production systems investigated, the most promising for future experiments is ROSIE-2. When ROSIE-2 or some other production system is found to be suitable, then experiments should continue with the creation and implementation of interacting PTAPS rules and TSA rules.

APPENDIX I. PTAPS IN A MODIFIED STAMMER1

I.1 INTRODUCTION

A few explanations may be helpful when reading the typescripts in sections I.2 and I.3.

1. Lines beginning with the prompt "@" or the LISP prompt "_" are those typed by the user.

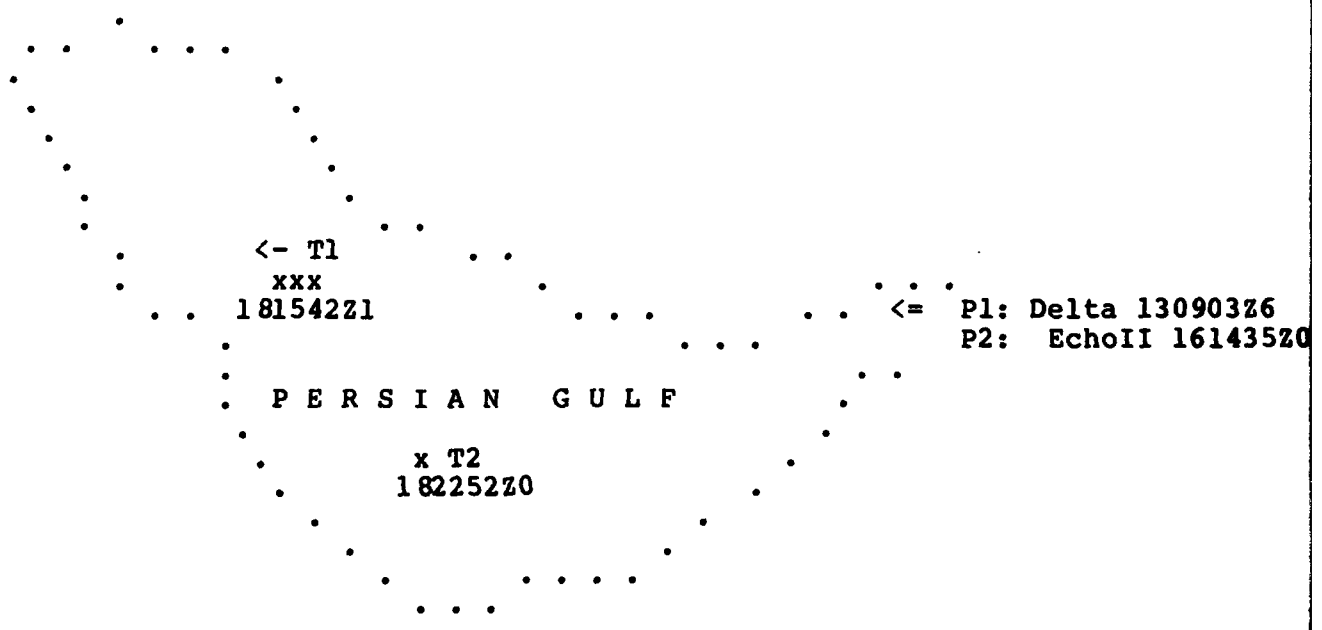
2. The system returns "NIL" when it has finished responding to the user's command.

3. All commands shown except ENTERMSG and RUNPD are optional. The RUNPD command cycles through the rules once. In some cases, more than one cycle of the rules is needed, since the actions of some rules can satisfy the conditions of other rules accessed earlier.

4. The actual messages are lists of assertions. For example, in the two-submarine demonstration the "snapshot" background information, "two submarines are presently in the region, a Delta and an Echo II," is represented by the assertions (subsurf category P2), (Delta class P1), (Echo II class P2), and so forth. For convenience, the expressions P1, P2, etc, are used to label platforms, and T1, T2, etc, to label tracks.

I.2 Typescript of Two-Submarine Scenario

```
 @demo.sub

INTERLISP-10  10-AUG-79 ...

Hello.

(<GDILLARD>DEMO.SUB;1 . <LISP>LISP.SAV;132)
_(EXPLAIN 'EXAMPLE1)


TWO-SUBMARINE EXAMPLE

Only two submarines could be in the region, a Delta and an EchoII, and
these are designated in the system as platforms P1 and P2,
respectively.  Two subsurface tracks, T1 and T2, are reported, and the
acoustic signature of T1 shows that it cannot be a Delta.  The
production system is able to conclude that T1 is a track of P2 and
that T2 is a track of P1.

* * * * * * * * * * * *

NIL

_(PICTURE)
```



NIL
_(EXPLAIN 'FILES)

TRACK FILES & PLATFORM FILES

RTF: The Region's Track File contains all surface tracks and subsurface tracks in the region, except for ownforce tracks. [Ownforce tracks can be included when positions are uncertain, but otherwise they are more conveniently handled separately.]

RPF: The Region's Platform File contains all surface and subsurface platforms which are known to be or thought possibly to be inside the region, with the exception of ownforce platforms.

SUBSET FILES: The system also maintains platform files that are subsets of RPF and track files that are subsets of RTF. For example, RPF has a subset file for subsurface platforms, and also can have a subset for destroyers and a subset for a particular class of destroyer.

COMPLETE: A platform file is complete if it is known to contain every platform of that kind which is in the region or could possibly be in the region. A track file is complete if it is known to contain the tracks of all platforms of that kind in the region.

CORRESPONDING FILES: A track file containing tracks of subsurface platforms, for example, has as its corresponding file the platform file of subsurface platforms thought to be in the region.

NIL
_(EXPLAIN 'ORFILES)

OR-FILES

A track is a member of a platform's OR-file if it has not been ruled out as a track of that platform.

A platform is a member of a track's OR-file if the track has not been ruled out as a track of that platform.

The production system gives each OR-file it creates a name, eg, F0015.

NIL
_(BACKGROUND)

Background Data: The entrance/exit area of an enclosed body of water is continuously monitored by acoustic devices. It is known that two submarines are presently in the region, a Delta and an Echo II, but their locations are not known. (The assertions now in the data base

```
can be seen by typing BD.)
NIL
_(ENTERMSG BD)


MESSAGE ENTERED
NIL
_(READMSG1)


Message 1:   T1  is  a  track;  T1 is inside-region; subsurface is the
category of T1.
NIL
_(ENTERMSG M1)


MESSAGE ENTERED
NIL
_(RUNPD)

RTF MEMBER RULE FIRES.  CONCLUSION:
T1 IS A MEMBER OF RTF, THE REGION'S TRACK FILE;
F0045 IS THE OR-FILE OF T1.

TRACK-CATEGORY MEMBER RULE FIRES.  CONCLUSION:
T1 IS A MEMBER OF THE CATEGORY-SUBSET OF RTF WHOSE
CATEGORY IS SUBSURF.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P1 AND T1 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P2 AND T1 ARE MEMBERS OF EACH OTHERS' OR-FILES.

COMPLETE TRACK-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF T1 IS COMPLETE BECAUSE T1
 IS A MEMBER OF A TRACK-FILE WHOSE
CORRESPONDING PLATFORM-FILE IS COMPLETE.
NIL
_(ORFILE 'P1)

The OR-file of P1 is not known to be complete;
its members are:  (T1).
      NIL
_(ORFILE 'P2)

The OR-file of P2 is not known to be complete;
its members are:  (T1).
      NIL
_(ORFILE 'T1)

The OR-file of T1 is complete;
```

```
its members are:  (P1 P2).
      NIL
_(ORFILE 'T2)

The OR-file of T2 is not known to be complete;
its members are:  NIL.
      NIL
_(READMSG2)


Message 2:  T2 is a track; T2 is inside-region; subsurface is the
category of T2.
NIL
_(ENTERMSG M2)


MESSAGE ENTERED
NIL
_(RUNPD)

RTF MEMBER RULE FIRES.  CONCLUSION:
T2 IS A MEMBER OF RTF, THE REGION'S TRACK FILE;
F0068 IS THE OR-FILE OF T2.

TRACK-CATEGORY MEMBER RULE FIRES.  CONCLUSION:
T2 IS A MEMBER OF THE CATEGORY-SUBSET OF RTF WHOSE
CATEGORY IS SUBSURF.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P1 AND T2 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P2 AND T2 ARE MEMBERS OF EACH OTHERS' OR-FILES.

COMPLETE TRACK-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF T2 IS COMPLETE BECAUSE T2
 IS A MEMBER OF A TRACK-FILE WHOSE
CORRESPONDING PLATFORM-FILE IS COMPLETE.

COMPLETE TRACK-FILE RULE FIRES.  CONCLUSION:
THE FILE OF SUBSURF
 TRACKS IS COMPLETE BECAUSE ITS CORRESPONDING
PLATFORM-FILE IS COMPLETE AND HAS THE SAME NUMBER OF MEMBERS.

NIL
_(READMSG3)


Message 3:  Delta is a member of the impossible-class-file of the
acoustic-data of T1.
NIL
_(ENTERMSG M3)
```

```
MESSAGE ENTERED
NIL
_(RUNPD)

IMPOS-TRACK BY ACOUSTIC-DATA RULE FIRES.  CONCLUSION:
T1  IS AN IMPOSSIBLE-TRACK OF   P1.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T1 and P1 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T1
 IS AN IMPOSSIBLE TRACK OF P1.

COMPLETE PLATFORM-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF P1 IS COMPLETE BECAUSE P1
IS A MEMBER OF A PLATFORM-FILE WHOSE CORRESPONDING
TRACK-FILE IS COMPLETE.


COMPLETE PLATFORM-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF P2 IS COMPLETE BECAUSE P2
IS A MEMBER OF A PLATFORM-FILE WHOSE CORRESPONDING
TRACK-FILE IS COMPLETE.

NIL
_(RUNPD)

AND-THEN-THERE-WAS-ONE PLATFORM RULE FIRES.  CONCLUSION:
T1 IS THE TRACK OF P2!!

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T2 IS AN IMPOSSIBLE-TRACK OF P2 BECAUSE T1 IS THE TRACK OF P2.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T2 and P2 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T2
 IS AN IMPOSSIBLE TRACK OF P2.
NIL
_(RUNPD)

AND-THEN-THERE-WAS-ONE PLATFORM RULE FIRES.  CONCLUSION:
T2 IS THE TRACK OF P1!!
NIL
_(ORFILE 'P1)

The OR-file of P1 is complete;
its members are:  (T2).
      NIL
_(ORFILE 'P2)

The OR-file of P2 is complete;
its members are:  (T1).
      NIL
```

```
_(ORFILE 'T1)

The OR-file of T1 is complete;
its members are:  (P2).
      NIL
_(ORFILE 'T2)

The OR-file of T2 is complete;
its members are:  (P1).
      NIL
_(LOGOUT)
```

SUMMARY OF EXAMPLE 1

Message 1: Subsurface track T1 is reported.

Message 2: Subsurface track T2 is reported.

Message 3: Acoustic data associated with track T1 indicates it is not a Delta.

	MEMBERS OF TRACK'S OR-FILE		
TRACK	AFTER MESSAGE 1	AFTER MESSAGE 2	AFTER MESSAGE 3
T1	P1 P2 (COMPLETE)	P1 P2 (COMPLETE)	P2 (DELTA) (COMPLETE)
T2	_____	P1 P2 (COMPLETE)	P1 (ECHO II) (COMPLETE)

	MEMBERS OF PLATFORM'S OR-FILE		
PLATFORM	AFTER MESSAGE 1	AFTER MESSAGE 2	AFTER MESSAGE 3
P1	T1	T1 T2 (COMPLETE)	T1 (COMPLETE)
P2	T1	T1 T2 (COMPLETE)	T2 (COMPLETE)

I.3 Typescript of UNREP Scenario

```
 @DEMO.UNREP

INTERLISP-10  10-AUG-79 ...

Good morning.

(<GDILLARD>DEMO.UNREP;1 . <LISP>LISP.SAV;132)
_(EXPLAIN 'EXAMPLE2)


HIGH-ALTITUDE SURVEILLANCE EXAMPLE

                    Soviet UNREP Group

                         <== <==


                       *  T2

                 *  T1

                                        *  T4

                              *  T3
```

No radar tracks are available to ownship, because of EMCON conditions, but recent positions on all major surface ships have been obtained from a satellite radar map. The positions of ownforce ships are known, and the locations of two commercial ships are known sufficiently that they can be associated with their tracks on the map.

There are four remaining tracks (T1, T2, T3, T4) and it is concluded that these correspond to a small Soviet UNREP group (CG155, DDG233, AO7, AE12) that earlier had been reported heading for the area.

A patrol aircraft had overflown the oiler two hours earlier, and it is calculated that the oiler could not have reached the position of T1 or T2.

T1 is in the lead position, so T1 is ruled out as being either the oiler or ammunition ship.

A signal intercept is reported by the ESM system at a bearing consistent with the positions of T3 and T4. A list of ship classes having that emitter type are determined from the emitter/class file,

and, of the ships in the Soviet group, only the class of the DDG223 is on this list.

The production system is able to conclude that:

T1 is the track of CG155

T2 is the track of AE12

T3 and T4 are tracks of DDG223 and AO7.

* * * * * * * *

NIL
_(BACKGROUND)

Background Data: Platforms P1 - P4 comprise a small Soviet UNREP group thought to be in the region or entering it soon, while P5 and P6 are commercial ships whose locations have recently been confirmed. It is known that no other surface ships could have reached the region. There are presently no active surface tracks in RTF, the region's [non-ownforce] track file.

- o P1 is CG155, a Kara class guided missile cruiser
- o P2 is DDG233, a Krivak class guided missile destroyer
- o P3 is A07, an oiler
- o P4 is AE12, an ammunition ship
- o P5 and P6 are known merchants

NIL
_(ENTERMSG BD)

MESSAGE ENTERED
NIL
_(DESCRIBEMSG1)

A satellite radar map provides positions on all major surface ships in the region. The track-correlation subsystem preprocesses the data, successfully eliminating ownforce tracks and associating two of the tracks with the two merchants. The subsystem fails to associate four of the tracks with any platform in the region's platform file. In addition to positional data, the track-correlation subsystem sends the following information to the system data base.

MESSAGE M1

```
  o  T1 is a track, T2 is a track, ..., T6 is a track

  o  T1 is in-region, T2 is in-region, ..., T6 is in-region

  o  surface is the category of T1, ..., surface is the category of T6

  o  T5 is the track of P5

  o  T6 is the track of P6

  o  Complete is the status of the track-file whose category is surface

NIL
_(ENTERMSG M1)


MESSAGE ENTERED
NIL
_(RUNPD)

RTF MEMBER RULE FIRES.  CONCLUSION:
T1 IS A MEMBER OF RTF, THE REGION'S TRACK FILE;
F0119 IS THE OR-FILE OF T1.

RTF MEMBER RULE FIRES.  CONCLUSION:
T2 IS A MEMBER OF RTF, THE REGION'S TRACK FILE;
F0124 IS THE OR-FILE OF T2.

RTF MEMBER RULE FIRES.  CONCLUSION:
T3 IS A MEMBER OF RTF, THE REGION'S TRACK FILE;
F0129 IS THE OR-FILE OF T3.

RTF MEMBER RULE FIRES.  CONCLUSION:
T4 IS A MEMBER OF RTF, THE REGION'S TRACK FILE;
F0134 IS THE OR-FILE OF T4.

RTF MEMBER RULE FIRES.  CONCLUSION:
T5 IS A MEMBER OF RTF, THE REGION'S TRACK FILE;
F0139 IS THE OR-FILE OF T5.

RTF MEMBER RULE FIRES.  CONCLUSION:
T6 IS A MEMBER OF RTF, THE REGION'S TRACK FILE;
F0144 IS THE OR-FILE OF T6.

TRACK-CATEGORY MEMBER RULE FIRES.  CONCLUSION:
T1 IS A MEMBER OF THE CATEGORY-SUBSET OF RTF WHOSE
CATEGORY IS SURFACE.

TRACK-CATEGORY MEMBER RULE FIRES.  CONCLUSION:
T2 IS A MEMBER OF THE CATEGORY-SUBSET OF RTF WHOSE
CATEGORY IS SURFACE.
```

TRACK-CATEGORY MEMBER RULE FIRES. CONCLUSION:
T3 IS A MEMBER OF THE CATEGORY-SUBSET OF RTF WHOSE
CATEGORY IS SURFACE.

TRACK-CATEGORY MEMBER RULE FIRES. CONCLUSION:
T4 IS A MEMBER OF THE CATEGORY-SUBSET OF RTF WHOSE
CATEGORY IS SURFACE.

TRACK-CATEGORY MEMBER RULE FIRES. CONCLUSION:
T5 IS A MEMBER OF THE CATEGORY-SUBSET OF RTF WHOSE
CATEGORY IS SURFACE.

TRACK-CATEGORY MEMBER RULE FIRES. CONCLUSION:
T6 IS A MEMBER OF THE CATEGORY-SUBSET OF RTF WHOSE
CATEGORY IS SURFACE.

IMPOS-TRACK BY TRACK-ELIM RULE FIRES. CONCLUSION:
T5 IS AN IMPOSSIBLE-TRACK OF P1 BECAUSE IT IS THE TRACK OF P5.

IMPOS-TRACK BY TRACK-ELIM RULE FIRES. CONCLUSION:
T5 IS AN IMPOSSIBLE-TRACK OF P2 BECAUSE IT IS THE TRACK OF P5.

IMPOS-TRACK BY TRACK-ELIM RULE FIRES. CONCLUSION:
T5 IS AN IMPOSSIBLE-TRACK OF P3 BECAUSE IT IS THE TRACK OF P5.

IMPOS-TRACK BY TRACK-ELIM RULE FIRES. CONCLUSION:
T5 IS AN IMPOSSIBLE-TRACK OF P4 BECAUSE IT IS THE TRACK OF P5.

IMPOS-TRACK BY TRACK-ELIM RULE FIRES. CONCLUSION:
T5 IS AN IMPOSSIBLE-TRACK OF P6 BECAUSE IT IS THE TRACK OF P5.

IMPOS-TRACK BY TRACK-ELIM RULE FIRES. CONCLUSION:
T6 IS AN IMPOSSIBLE-TRACK OF P1 BECAUSE IT IS THE TRACK OF P6.

IMPOS-TRACK BY TRACK-ELIM RULE FIRES. CONCLUSION:
T6 IS AN IMPOSSIBLE-TRACK OF P2 BECAUSE IT IS THE TRACK OF P6.

IMPOS-TRACK BY TRACK-ELIM RULE FIRES. CONCLUSION:
T6 IS AN IMPOSSIBLE-TRACK OF P3 BECAUSE IT IS THE TRACK OF P6.

IMPOS-TRACK BY TRACK-ELIM RULE FIRES. CONCLUSION:
T6 IS AN IMPOSSIBLE-TRACK OF P4 BECAUSE IT IS THE TRACK OF P6.

IMPOS-TRACK BY TRACK-ELIM RULE FIRES. CONCLUSION:
T6 IS AN IMPOSSIBLE-TRACK OF P5 BECAUSE IT IS THE TRACK OF P6.

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES. CONCLUSION:
T1 IS AN IMPOSSIBLE-TRACK OF P5 BECAUSE T5 IS THE TRACK OF P5.

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES. CONCLUSION:
T2 IS AN IMPOSSIBLE-TRACK OF P5 BECAUSE T5 IS THE TRACK OF P5.

```
IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T3 IS AN IMPOSSIBLE-TRACK OF P5 BECAUSE T5 IS THE TRACK OF P5.

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T4 IS AN IMPOSSIBLE-TRACK OF P5 BECAUSE T5 IS THE TRACK OF P5.

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T1 IS AN IMPOSSIBLE-TRACK OF P6 BECAUSE T6 IS THE TRACK OF P6.

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T2 IS AN IMPOSSIBLE-TRACK OF P6 BECAUSE T6 IS THE TRACK OF P6.

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T3 IS AN IMPOSSIBLE-TRACK OF P6 BECAUSE T6 IS THE TRACK OF P6.

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T4 IS AN IMPOSSIBLE-TRACK OF P6 BECAUSE T6 IS THE TRACK OF P6.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P1 AND T1 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P1 AND T2 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P1 AND T3 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P1 AND T4 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P2 AND T1 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P2 AND T2 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P2 AND T3 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P2 AND T4 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P3 AND T1 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P3 AND T2 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P3 AND T3 ARE MEMBERS OF EACH OTHERS' OR-FILES.
```

```
OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P3 AND T4 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P4 AND T1 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P4 AND T2 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P4 AND T3 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P4 AND T4 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P5 AND T5 ARE MEMBERS OF EACH OTHERS' OR-FILES.

OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P6 AND T6 ARE MEMBERS OF EACH OTHERS' OR-FILES.

COMPLETE TRACK-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF T1 IS COMPLETE BECAUSE T1
 IS A MEMBER OF A TRACK-FILE WHOSE
CORRESPONDING PLATFORM-FILE IS COMPLETE.

COMPLETE TRACK-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF T2 IS COMPLETE BECAUSE T2
 IS A MEMBER OF A TRACK-FILE WHOSE
CORRESPONDING PLATFORM-FILE IS COMPLETE.

COMPLETE TRACK-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF T3 IS COMPLETE BECAUSE T3
 IS A MEMBER OF A TRACK-FILE WHOSE
CORRESPONDING PLATFORM-FILE IS COMPLETE.

COMPLETE TRACK-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF T4 IS COMPLETE BECAUSE T4
 IS A MEMBER OF A TRACK-FILE WHOSE
CORRESPONDING PLATFORM-FILE IS COMPLETE.

COMPLETE TRACK-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF T5 IS COMPLETE BECAUSE T5
 IS A MEMBER OF A TRACK-FILE WHOSE
CORRESPONDING PLATFORM-FILE IS COMPLETE.

COMPLETE TRACK-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF T6 IS COMPLETE BECAUSE T6
 IS A MEMBER OF A TRACK-FILE WHOSE
CORRESPONDING PLATFORM-FILE IS COMPLETE.

COMPLETE PLATFORM-OR-FILE RULE FIRES.  CONCLUSION:
```

```
THE OR-FILE OF P1 IS COMPLETE BECAUSE P1
IS A MEMBER OF A PLATFORM-FILE WHOSE CORRESPONDING
TRACK-FILE IS COMPLETE.


COMPLETE PLATFORM-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF P2 IS COMPLETE BECAUSE P2
IS A MEMBER OF A PLATFORM-FILE WHOSE CORRESPONDING
TRACK-FILE IS COMPLETE.


COMPLETE PLATFORM-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF P3 IS COMPLETE BECAUSE P3
IS A MEMBER OF A PLATFORM-FILE WHOSE CORRESPONDING
TRACK-FILE IS COMPLETE.


COMPLETE PLATFORM-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF P4 IS COMPLETE BECAUSE P4
IS A MEMBER OF A PLATFORM-FILE WHOSE CORRESPONDING
TRACK-FILE IS COMPLETE.


COMPLETE PLATFORM-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF P5 IS COMPLETE BECAUSE P5
IS A MEMBER OF A PLATFORM-FILE WHOSE CORRESPONDING
TRACK-FILE IS COMPLETE


COMPLETE PLATFORM-OR-FILE RULE FIRES.  CONCLUSION:
THE OR-FILE OF P6 IS COMPLETE BECAUSE P6
IS A MEMBER OF A PLATFORM-FILE WHOSE CORRESPONDING
TRACK-FILE IS COMPLETE.

NIL
_(DESCRIBEMSG2)
```

The track-geometry subsystem, interacting with the production system, compares the last-inactive-track of P3 (from an earlier overflight of the oiler) with each track and concludes that the oiler could not have reached the positions of T1 and T2. Also, a number of task-group system-logic rules fire [omitted from this abbreviated demonstration], and the subsystem interacts with the production system to conclude that track T1 is leading the task group. The track-geometry subsystem provides the data base with the following information.

MESSAGE M2

- o T1 is an unreachable of P3
- o T2 is an unreachable of P3

o Lead-position is a function of T1

```
NIL
_(ENTERMSG M2)


MESSAGE ENTERED
NIL
_(RUNPD)

IMPOS-TRACK BY EARLIER-SIGHTING RULE FIRES.
CONCLUSION:  T1 IS AN IMPOSSIBLE TRACK OF P3.

IMPOS-TRACK BY EARLIER-SIGHTING RULE FIRES.
CONCLUSION:  T2 IS AN IMPOSSIBLE TRACK OF P3.

IMPOS-TRACK BY LEAD-POSITION RULE FIRES.  CONCLUSION:  T1
IS LEADING A TASK GROUP, SO IS NOT THE TRACK OF AN OILER
OR AMMUNITION SHIP, AND THEREFORE IS AN IMPOSSIBLE-TRACK OF P4.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T1 and P3 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T1
 IS AN IMPOSSIBLE TRACK OF P3.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T2 and P3 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T2
 IS AN IMPOSSIBLE TRACK OF P3.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T1 and P4 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T1
 IS AN IMPOSSIBLE TRACK OF P4.
NIL
_(RUNPD)
NIL
_(DESCRIBEMSG3)
```

A report of a signal intercept is handled by the ESM Data Processor, which calculates that the emitter is within the region and that the bearing data are consistent with the positions of both T3 and T4. The ESM Data Processor sends the following. [Already in the data base is a file of platform classes that can emit signal-type K1, and one of these classes is the Krivak. Also, a file of platform general-types that can emit signal-type K1 is in the data base.]

MESSAGE M3

- o S1 is an emission; S1 is emitted-in-region
- o K1 is the emitter-type of S1

```
  o  B1 is the bearing-data of S1

  o  B1 is bearing-consistent with T3

  o  B1 is bearing-consistent with T4

NIL
_(ENTERMSG M3)


MESSAGE ENTERED
NIL
_(RUNPD)

REF MEMBER RULE FIRES.  CONCLUSION:
S1 IS A MEMBER OF REF (THE REGION'S EMISSION FILE);
E0317 IS THE OR-FILE OF S1.

IMPOS-EMITTER BY PLATFORM CLASS RULE FIRES.  CONCLUSION:
P1 IS AN IMPOSSIBLE EMITTER OF SIGNAL S1 BECAUSE A KARA
-CLASS SHIP DOES NOT CARRY THAT
EMITTER TYPE.

IMPOS-EMITTER BY PLATFORM CLASS RULE FIRES.  CONCLUSION:
P3 IS AN IMPOSSIBLE EMITTER OF SIGNAL S1 BECAUSE A KAZBEK
-CLASS SHIP DOES NOT CARRY THAT
EMITTER TYPE.

IMPOS-EMITTER BY PLATFORM CLASS RULE FIRES.  CONCLUSION:
P4 IS AN IMPOSSIBLE EMITTER OF SIGNAL S1 BECAUSE A KAMMO
-CLASS SHIP DOES NOT CARRY THAT
EMITTER TYPE.

IMPOS-EMITTER BY PLATFORM-GENERAL-TYPE RULE FIRES.
CONCLUSION:  P5 IS AN IMPOSSIBLE-EMITTER OF SIGNAL S1
 BECAUSE A SHIP OF GENERAL-TYPE COMMERCIAL DOES NOT CARRY
THAT EMITTER TYPE.

IMPOS-EMITTER BY PLATFORM-GENERAL-TYPE RULE FIRES.
CONCLUSION:  P6 IS AN IMPOSSIBLE-EMITTER OF SIGNAL S1
 BECAUSE A SHIP OF GENERAL-TYPE COMMERCIAL DOES NOT CARRY
THAT EMITTER TYPE.

IMPOS-EMITTER BY BEARING RULE FIRES.  CONCLUSION:
T1 IS AN IMPOSSIBLE-EMITTER OF S1.

IMPOS-EMITTER BY BEARING RULE FIRES.  CONCLUSION:
T2 IS AN IMPOSSIBLE-EMITTER OF S1.

IMPOS-EMITTER BY BEARING RULE FIRES.  CONCLUSION:
T5 IS AN IMPOSSIBLE-EMITTER OF S1.
```

```
IMPOS-EMITTER BY BEARING RULE FIRES.  CONCLUSION:
T6 IS AN IMPOSSIBLE-EMITTER OF S1.

EMISSION OR-FILE MEMBER RULE FIRES.  CONCLUSION:
P2 IS A MEMBER OF THE OR-FILE OF THE SIGNAL S1.

COMPLETE EMISSION OR-FILE RULE FIRES.  CONCLUSION:  THE OR-FILE OF
SIGNAL S1
 IS COMPLETE BECAUSE THE FILE OF SURFACE
PLATFORMS IS COMPLETE AND THE EMITTER IS ON A SURFACE SHIP.

AND-THEN-THERE-WAS-ONE PLATFORM-EMITTER RULE FIRES:
CONCLUSION:  P2 IS THE PLATFORM-EMITTER OF S1
BECAUSE ALL OTHER PLATFORMS WERE ELIMINATED
AS POSSIBLE EMITTERS OF THAT SIGNAL.

IMPOS-TRACK BY PLATFORM-EMISSION ASSOCIATION RULE FIRES.
CONCLUSION:  T1 IS AN IMPOSSIBLE-TRACK OF P2,
SINCE THE SIGNAL EMITTED BY P2 COULD NOT HAVE COME FROM TRACK T1.

IMPOS-TRACK BY PLATFORM-EMISSION ASSOCIATION RULE FIRES.
CONCLUSION:  T2 IS AN IMPOSSIBLE-TRACK OF P2,
SINCE THE SIGNAL EMITTED BY P2 COULD NOT HAVE COME FROM TRACK T2.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T1 and P2 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T1
 IS AN IMPOSSIBLE TRACK OF P2.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T2 and P2 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T2
 IS AN IMPOSSIBLE TRACK OF P2.
NIL
_(RUNPD)

AND-THEN-THERE-WAS-ONE PLATFORM RULE FIRES.  CONCLUSION:
T1 IS THE TRACK OF P1!!

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T2 IS AN IMPOSSIBLE-TRACK OF P1 BECAUSE T1 IS THE TRACK OF P1.

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T3 IS AN IMPOSSIBLE-TRACK OF P1 BECAUSE T1 IS THE TRACK OF P1.

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T4 IS AN IMPOSSIBLE-TRACK OF P1 BECAUSE T1 IS THE TRACK OF P1.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T2 and P1 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T2
 IS AN IMPOSSIBLE TRACK OF P1.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T3 and P1 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T3
```

```
 IS AN IMPOSSIBLE TRACK OF P1.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T4 and P1 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T4
 IS AN IMPOSSIBLE TRACK OF P1.
NIL
_(RUNPD)

AND-THEN-THERE-WAS-ONE PLATFORM RULE FIRES.  CONCLUSION:
T2 IS THE TRACK OF P4!!

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T3 IS AN IMPOSSIBLE-TRACK OF P4 BECAUSE T2 IS THE TRACK OF P4.

IMPOS-TRACK BY PLATFORM-ELIM RULE FIRES.  CONCLUSION:
T4 IS AN IMPOSSIBLE-TRACK OF P4 BECAUSE T2 IS THE TRACK OF P4.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T3 and P4 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T3
 IS AN IMPOSSIBLE TRACK OF P4.

ORFILE REDUCTION RULE FIRES.  CONCLUSION:
T4 and P4 ARE REMOVED FROM EACH OTHERS' ORFILES BECAUSE T4
 IS AN IMPOSSIBLE TRACK OF P4.
NIL
_(ORFILE 'T1)

The OR-file of T1 is complete;
its members are:  (P1).
      NIL
_(ORFILE 'T2)

The OR-file of T2 is complete;
its members are:  (P4).
      NIL
_(ORFILE 'T3)

The OR-file of T3 is complete;
its members are:  (P2 P3).
      NIL
_(ORFILE 'T4)

The OR-file of T4 is complete;
its members are:  (P2 P3).
      NIL
_(ORFILE 'T5)

The OR-file of T5 is complete;
its members are:  (P5).
      NIL
_(ORFILE 'T6)
```

```
The OR-file of T6 is complete;
its members are:  (P6).
      NIL
_(ORFILE 'P1)

The OR-file of P1 is complete;
its members are:  (T1).
      NIL
_(ORFILE 'P2)

The OR-file of P2 is complete;
its members are:  (T3 T4).
      NIL
_(ORFILE 'P3)

The OR-file of P3 is complete;
its members are:  (T3 T4).
      NIL
_(ORFILE 'P4)

The OR-file of P4 is complete;
its members are:  (T2).
      NIL

_
(PLATFORMS)
```

- P1 is CG155, a Kara class guided missile cruiser
- P2 is DDG233, a Krivak class guided missile destroyer
- P3 is A07, an oiler
- P4 is AE12, an ammunition ship
- P5 and P6 are known merchants

```
NIL
_(LOGOUT)
KILLED JOB 39, USER GDILLARD, ACCT ACCAT, TTY 41, AT  1/28/80 0748
  USED 0:1:27 IN 0:19:29
```

SUMMARY OF EXAMPLE 2

Message 1: [from track correlation subsystem] Six tracks originating from a satellite radar map are reported; tracks T5 and T6 have already been paired with merchants P5 and P6, respectively.

Message 2: [from track geometry subsystem] T1 and T2 are unreachable from an earlier position of oiler P3. Also, the leader of the task group is T1, implying that T1 is not the track of replenishment ship P3 or P4.

Message 3: [from ESM data processor] An emission of type K1 is emitted from within the region. The source of the emission could only be T3 or T4. (Production rules then eliminate platforms P1 and P3-P6 as possible emitters of signal type K1, and conclude that T3 or T4 is the track of P2.)

	MEMBERS OF TRACK'S OR-FILE		
TRACK	AFTER MESSAGE 1	AFTER MESSAGE 2	AFTER MESSAGE 3
T1	P1 P2 P3 P4	P1 P2	P1
T2	P1 P2 P3 P4	P1 P2 P4	P4
T3	P1 P2 P3 P4	P1 P2 P3 P4	P2 P3
T4	P1 P2 P3 P4	P1 P2 P3 P4	P2 P3
T5	P5	P5	P5
T6	P6	P6	P6

I.4 PTAPS Rules in STAMMER1 Syntax

RTF Member Rule

Conditions:

```
((GETS TR (RETRIEVE2 'IS* 'TRACK)) (RETRIEVE3B 'INREGION 'IS TR)
(UNLESS ( RETRIEVE3B TR 'MEMBER 'RTF)) (SETQ N (GENSYM 'F)))
```

Actions:

```
((ASSERT TR 'MEMBER 'RTF) (ASSERT 'RTF 'MEMBER* TR) (ASSERT N 'ORFILE
TR) ( ASSERT TR 'ORFILE* N))
```

Track-Category Member Rule

Conditions:

```
((GETS TR (RETRIEVE2 'MEMBER 'RTF)) (GETS CTG (RETRIEVE2 'WCATEGORY
TR)) (GETS TF (RETRIEVE2 'CATEGSUBSET 'RTF)) (RETRIEVE3B CTG 'CATEGORY
TF) (UNLESS ( RETRIEVE3B TR 'MEMBER TF)))
```

Actions:

```
((ASSERT TR 'MEMBER TF) (ASSERT TF 'MEMBER* TR))
```

Impos-Track by Acoustic-Data Rule

Conditions:

```
((GETS TR (RETRIEVE2 'MEMBER 'RTF)) (RETRIEVE3B 'SUBSURF 'WCATEGORY
TR) (GETS AD (RETRIEVE2 'ACOUSDATA TR)) (GETS ICF (RETRIEVE2
'IMPOSCLASSFILE AD)) (GETS P (RETRIEVE2 'MEMBER 'RPF)) (UNLESS
(RETRIEVE3B TR 'IMPOSTRACK P)) (GETS CLS ( RETRIEVE2 'CLASS P))
(RETRIEVE3B CLS 'MEMBER ICF))
```

Actions:

```
((ASSERT TR 'IMPOSTRACK P) (ASSERT P 'IMPOSTRACK* TR))
```

OR-File Member Rule

Conditions:

```
((GETS P (RETRIEVE2 'MEMBER 'RPF)) (GETS TR (RETRIEVE2 'MEMBER 'RTF))
(UNLESS ( RETRIEVE3B TR 'IMPOSTRACK P)) (GETS ORF (RETRIEVE2 'ORFILE
P)) (GETS FRO ( RETRIEVE2 'ORFILE TR)) (UNLESS (RETRIEVE3B TR 'MEMBER
ORF)))
```

Actions:

((ASSERT TR 'MEMBER ORF) (ASSERT ORF 'MEMBER* TR) (ASSERT P 'MEMBER FRO) (ASSERT FRO 'MEMBER* P))

Complete Track-OR-File Rule

Conditions:

((GETS PF (RETRIEVE2 'IS* 'PLATFORMFILE)) (RETRIEVE3B 'COMPLETE 'STATUS PF) (GETS TF (RETRIEVE2 'CORRESPFILE PF)) (GETS TR (RETRIEVE2 'MEMBER TF)) (GETS FRO (RETRIEVE2 'ORFILE TR)) (UNLESS (RETRIEVE3B 'COMPLETE 'STATUS FRO)))

Actions:

((ASSERT 'COMPLETE 'STATUS FRO) (ASSERT FRO 'STATUS* 'COMPLETE))

And-Then-There-Was-One Platform Rule

Conditions:

((GETS TR (RETRIEVE2 'MEMBER 'RTF)) (GETS FRO (RETRIEVE2 'ORFILE TR)) (RETRIEVE3B 'COMPLETE 'STATUS FRO) (EQP (LENGTH (RETRIEVE2 'MEMBER FRO)) 1) (GETS P (RETRIEVE2 'MEMBER FRO)) (UNLESS (RETRIEVE3B TR 'TRACK P)))

Actions:

((ASSERT TR 'TRACK P) (ASSERT P 'TRACK* TR))

Impos-Track by Platform-Elim Rule

Conditions:

((GETS P (RETRIEVE2 'MEMBER 'RPF)) (GETS AT (RETRIEVE2 'TRACK P)) (GETS TR (RETRIEVE2 'MEMBER 'RTF)) (UNLESS (RETRIEVE3B TR 'IMPOSTRACK P)) (UNLESS (EQ TR AT)))

Actions:

((ASSERT TR 'IMPOSTRACK P) (ASSERT P 'IMPOSTRACK* TR))

OR-File Reduction Rule

Conditions:

((GETS P (RETRIEVE2 'MEMBER 'RPF)) (GETS ORF (RETRIEVE2 'ORFILE P)) (GETS TR (RETRIEVE2 'MEMBER ORF)) (RETRIEVE3B TR 'IMPOSTRACK P) (GETS FRO (RETRIEVE2 'ORFILE TR)))

Actions:

((ERASE1 (CAR (RETRIEVE3B TR 'MEMBER ORF))) (ERASE1 (CAR (RETRIEVE3B ORF 'MEMBER* TR))) (ERASE1 (CAR (RETRIEVE3B P 'MEMBER FRO))) (ERASE1 (CAR (RETRIEVE3B FRO 'MEMBER* P))))

Complete Platform-OR-File Rule

Conditions:

((GETS TF (RETRIEVE2 'IS* 'TRACKFILE)) (RETRIEVE3B 'COMPLETE 'STATUS TF) (GETS PF (RETRIEVE2 'CORRESPFILE* TF)) (GETS P (RETRIEVE2 'MEMBER PF)) (GETS ORF (RETRIEVE2 'ORFILE P)) (UNLESS (RETRIEVE3B 'COMPLETE 'STATUS ORF)))

Actions:

((ASSERT 'COMPLETE 'STATUS ORF) (ASSERT ORF 'STATUS* 'COMPLETE))

Complete Track-File Rule

Conditions:

((GETS PF (RETRIEVE2 'IS* 'PLATFORMFILE)) (RETRIEVE3B 'COMPLETE 'STATUS PF) (GETS TF (RETRIEVE2 'CORRESPFILE PF)) (EQP (LENGTH (RETRIEVE2 'MEMBER PF)) (LENGTH (RETRIEVE2 'MEMBER TF))) (UNLESS (RETRIEVE3B 'COMPLETE 'STATUS TF)))

Actions:

((ASSERT 'COMPLETE 'STATUS TF) (ASSERT TF 'STATUS* 'COMPLETE))

[The remaining rules are not needed for the Two-Submarine scenario.]

Impos-Track by Earlier-Sighting Rule

Conditions:

((GETS P (RETRIEVE2 'MEMBER 'RPF)) (GETS TR (RETRIEVE2 'MEMBER 'RTF)) (UNLESS (RETRIEVE3B TR 'IMPOSTRACK P)) (UNLESS (RETRIEVE3B TR 'TRACK P)) (RETRIEVE3B TR 'UNREACHABLE P))

Actions:

((ASSERT TR 'IMPOSTRACK P) (ASSERT P 'IMPOSTRACK* TR))

Impos-Track by Lead-Position Rule

Conditions:

((GETS TR (RETRIEVE2 'MEMBER 'RTF)) (RETRIEVE3B 'LEADPOSITION 'FUNCTION TR) (GETS P (RETRIEVE2 'MEMBER 'RPF)) (UNLESS (RETRIEVE3B TR 'IMPOSTRACK P)) (OR (RETRIEVE3B 'OILER 'GENTYPE P) (RETRIEVE3B

'AMMO 'GENTYPE P)))

Actions:

((ASSERT TR 'IMPOSTRACK P) (ASSERT P 'IMPOSTRACK* TR))

REF Member Rule

Conditions:

((GETS S (RETRIEVE2 'IS* 'EMISSION)) (UNLESS (RETRIEVE3B S 'MEMBER 'REF)) (RETRIEVE3B S 'IS* 'EMITTEDINREGION) (SETQ N (GENSYM 'E)))

Actions:

((ASSERT S 'MEMBER 'REF) (ASSERT 'REF 'MEMBER* S) (ASSERT N 'ORFILE S) (ASSERT S 'ORFILE* N))

Impos-Emitter by Platform Class Rule

Conditions:

((GETS S (RETRIEVE2 'MEMBER 'REF)) (GETS P (RETRIEVE2 'MEMBER 'RPF)) (UNLESS (RETRIEVE3B P 'IMPOSEMITTER S)) (GETS CL (RETRIEVE2 'CLASS P)) (GETS K (RETRIEVE2 'EMITTERTYPE S)) (GETS CORF (RETRIEVE2 'CLASSORFILE K)) (UNLESS (RETRIEVE3B CL 'MEMBER CORF)))

Actions:

((ASSERT P 'IMPOSEMITTER S) (ASSERT S 'IMPOSEMITTER* P))

Emission OR-File Member Rule

Conditions:

((GETS S (RETRIEVE2 'MEMBER 'REF)) (GETS P (RETRIEVE2 'MEMBER 'RPF)) (UNLESS (RETRIEVE3B P 'IMPOSEMITTER S)) (GETS SORF (RETRIEVE2 'ORFILE S)) (UNLESS (RETRIEVE3B P 'MEMBER SORF)))

Actions:

((ASSERT P 'MEMBER SORF) (ASSERT SORF 'MEMBER* P))

Complete Emission OR-File Rule

Conditions:

((GETS S (RETRIEVE2 'MEMBER 'REF)) (GETS PF (RETRIEVE2 'CATEGSUBSET 'RPF)) (RETRIEVE3B 'SURFACE 'CATEGORY PF) (RETRIEVE3B 'COMPLETE 'STATUS PF) (GETS K (RETRIEVE2 'EMITTERTYPE S)) (GETS SORF (RETRIEVE2 'ORFILE S)) (UNLESS (RETRIEVE3B 'COMPLETE 'STATUS SORF)))

Actions:

```
((ASSERT 'COMPLETE 'STATUS SORF) (ASSERT SORF 'STATUS* 'COMPLETE))
```

And-Then-There-Was-One Platform-Emitter Rule

Conditions:

```
((GETS S (RETRIEVE2 'MEMBER 'REF)) (GETS SORF (RETRIEVE2 'ORFILE S)) (
RETRIEVE3B 'COMPLETE 'STATUS SORF) (EQP (LENGTH (RETRIEVE2 'MEMBER
SORF)) 1) ( GETS P (RETRIEVE2 'MEMBER SORF)) (UNLESS (RETRIEVE3B P
'PLTFMEMITTER S)))
```

Actions:

```
((ASSERT P 'PLTFMEMITTER S) (ASSERT S 'PLTFMEMITTER* P))
```

Impos-Emitter by Bearing Rule

Conditions:

```
((GETS S (RETRIEVE2 'MEMBER 'REF)) (GETS TR (RETRIEVE2 'MEMBER 'RTF))
(UNLESS ( RETRIEVE3B TR 'IMPOSEMITTER S)) (GETS B (RETRIEVE2
'BEARINGDATA S)) (UNLESS ( RETRIEVE3B B 'BEARINGCONSISTENT TR)))
```

Actions:

```
((ASSERT TR 'IMPOSEMITTER S) (ASSERT S 'IMPOSEMITTER* TR))
```

Impos-Track by Platform-Emission Association Rule

Conditions:

```
((GETS S (RETRIEVE2 'MEMBER 'REF)) (GETS P (RETRIEVE2 'PLTFMEMITTER
S)) (GETS TR (RETRIEVE2 'MEMBER 'RTF)) (UNLESS (RETRIEVE3B TR
'IMPOSTRACK P)) (RETRIEVE3B TR 'IMPOSEMITTER S))
```

Actions:

```
((ASSERT TR 'IMPOSTRACK P) (ASSERT P 'IMPOSTRACK* TR))
```

Emission OR-File Reduction Rule

Conditions:

```
((GETS S (RETRIEVE2 'MEMBER 'REF)) (GETS SORF (RETRIEVE2 'ORFILE S))
(GETS P ( RETRIEVE2 'MEMBER SORF)) (RETRIEVE3B P 'IMPOSEMITTER S))
```

Actions:

```
((ERASE1 (CAR (RETRIEVE3B P 'MEMBER SORF))) (ERASE1 (CAR (RETRIEVE3B
SORF 'MEMBER* P))))
```

Impos-Emitter by Platform-General-Type Rule

Conditions:

```
((GETS S (RETRIEVE2 'MEMBER 'REF)) (GETS P (RETRIEVE2 'MEMBER 'RPF))
(UNLESS ( RETRIEVE3B P 'IMPOSEMITTER S)) (GETS GTY (RETRIEVE2 'GENTYPE
P)) (GETS K ( RETRIEVE2 'EMITTERTYPE S)) (GETS GORF (RETRIEVE2
'GENTYPEORFILE K)) (UNLESS ( RETRIEVE3B GTY 'MEMBER GORF)))
```

Actions:

```
((ASSERT P 'IMPOSEMITTER S) (ASSERT S 'IMPOSEMITTER* P))
```

Impos-Track by Track-Elim Rule

Conditions:

```
((GETS TR (RETRIEVE2 'MEMBER 'RTF)) (GETS AP (RETRIEVE2 'TRACK* TR))
(GETS P ( RETRIEVE2 'MEMBER 'RPF)) (UNLESS (RETRIEVE3B TR 'IMPOSTRACK
P)) (UNLESS (EQ P AP)))
```

Actions:

```
((ASSERT TR 'IMPOSTRACK P) (ASSERT P 'IMPOSTRACK* TR))
```

II. PTAPS in STAMMER2 -- Two-Submarine Scenario

```
#CONNection.to BBNA is complete.#

 BBN-SYSTEM-A, TOPS-20 Monitor 5B(3056)
@LOGIN RDILLARD
 Job 47 on TTY334 4-Sep-80 17:11:11
 End of LOGIN.CMD.3
@term no page
@stammer2
Type (STAMMER) to begin.
(<PMORRIS>STAMMER2.EXE.6 . PS1:<LISP>LISP.EXE.1330)
_load(ptaps-oracles)
(ORACLES reset)
(ORACLEFNS reset)
<RDILLARD>PTAPS-ORACLES..3
_(stammer)
Welcome to version 2.5 of the STAMMER TSA system.
Memory file? (Default is MEMORY.): ptaps-memory
Memory initialized.
Rulefile? (Default is RULES.):ptaps-rules
Rules loaded
What file would you like to take messages from?
(Default is SCENE.ICE): ptaps-msgfile
Are you running on a Tektronix?no
Do you have a Tektronix available for display? no

A0015: PF1 is  a PLATFORMFILE.
A0016: PF1 is  a CATEGSUBSET of RPF.
A0017: SUBSURF is  a CATEGORY of PF1.
A0018: TF1 is  a TRACKFILE.
A0019: TF1 is  a CATEGSUBSET of RTF.
A0020: SUBSURF is  a CATEGORY of TF1.
A0021: TF1 is  a CORRESPFILE of PF1.
A0022: COMPLETE is  a STATUS of PF1.
A0023: P1 is  a MEMBER of RPF.
A0024: P1 is  a MEMBER of PF1.
A0025: ORF1 is  a ORFILE of P1.
A0026: P2 is  a MEMBER of RPF.
A0027: P2 is  a MEMBER of PF1.
A0028: ORF2 is  a ORFILE of P2.

Question?  WHY is A0015
The information came directly from a message.
Question?  TELL me about P1
A0025: ORF1 is  a ORFILE of P1.
A0024: P1 is  a MEMBER of PF1.
A0023: P1 is  a MEMBER of RPF.
A0009: P1 is a SUB.
A0008: SUBSURF is  a CATEGORY of P1.
A0007: P1 is a DELTA.
A0006: P1 is HOSTILE.
```

```
A0005: P1 is a KNOWNPLATFORM.
Question?  WHY is A0005
That assertion is part of the technical data base
Question?  TELL me about P2
A0028: ORF2 is  a ORFILE of P2.
A0027: P2 is  a MEMBER of PF1.
A0026: P2 is  a MEMBER of RPF.
A0014: P2 is a SUB.
A0013: SUBSURF is  a CATEGORY of P2.
A0012: P2 is a ECHOII.
A0011: P2 is HOSTILE.
A0010: P2 is  a KNOWNPLATFORM.
Question?  Quit
Leaving EXPLAIN

     Message received from external source.
     Something detected at (28.1 50.0)  Time: 5
     Associated with track T1

Question?  Quit
Leaving EXPLAIN

A0033: T1 is  a TRACK.
A0034: SUBSURF is  a WCATEGORY of T1.
A0035: T1 is  a INREGION.

A0047: P2 is  a MEMBER of F0036.
A0046: T1 is  a MEMBER of ORF2.
A0045: P1 is  a MEMBER of F0036.
A0044: T1 is  a MEMBER of ORF1.
A0043: COMPLETE is  a STATUS of F0036.
A0042: F0036 is  a ORFILE of T1.
A0039: T1 is  a MEMBER of TF1.
A0038: T1 is  a MEMBER of RTF.
Question?  WHY is A0033
The information came directly from a message.
Question?  WHOSE CLASS is ECHOII
P2
Question?  WHAT IS THE CLASS OF P1
DELTA
Question?  TELL me about F0036
A0047: P2 is  a MEMBER of F0036.
A0045: P1 is  a MEMBER of F0036.
A0043: COMPLETE is  a STATUS of F0036.
A0042: F0036 is  a ORFILE of T1.
A0037: F0036 is  a ORFNUM of T1.
Question?  WHY is A0037
That assertion was computed by the oracle ORFNUM
Question?  WHY is A0047
STAMMER applied the rule(s)
ORFILEMEMB
Question?  WHY is A0046
```

```
STAMMER applied the rule(s)
ORFILEMEMB
Question?  PRINT the rule ORFILEMEMB
If a track t is not an impossible-track of a platform p, then p and t
are put into each others' OR-files.
Question?  WHY is A0044
STAMMER applied the rule(s)
ORFILEMEMB
Question?  WHY is A0043
STAMMER applied the rule(s)
COMPL.TRKORF
Question?  PRINT the rule COMPL.TRKORF
The OR-file of a track is complete if the track is a member of a track
file whose corresponding platform file is complete.
Question?  WHY is A0042
STAMMER applied the rule(s)
RTFMB
Question?  WHY is A0038
STAMMER applied the rule(s)
RTFMB
Question?  PRINT the rule RTFMB
Each new track is put into RTF, the region's track file.
Question?  WHY is A0039
STAMMER applied the rule(s)
TRKCTGMEMB
Question?  PRINT the rule TRKCTGMEMB
Each new track is put into a track file for tracks of that category.
Question?  CODE for the rule ORFILEMEMB
CONDITIONS:
((MEMBER RPF *P)
 (MEMBER RTF *TR)
 (*UNLESS* (IMPOSTRACK *P *TR))
 (ORFILE *P *ORF)
 (ORFILE *TR *FRO))

ACTIONS:
((MEMBER *ORF *TR)
 (MEMBER *FRO *P))

Question?  CODE for the rule COMPL.TRKORF
CONDITIONS:
((PLATFORMFILE *PF)
 (STATUS *PF COMPLETE)
 (CORRESPFILE *PF *TF)
 (MEMBER *TF *TR)
 (ORFILE *TR *FRO)
 (*UNLESS* (STATUS *FRO COMPLETE)))

ACTIONS:
((STATUS *FRO COMPLETE))

Question?  CODE for the rule TRKCTGMEMB
```

```
CONDITIONS:
((MEMBER RTF *TR)
 (WCATEGORY *TR *CTG)
 (CATEGSUBSET RTF *TF)
 (CATEGORY *TF *CTG))

ACTIONS:
((MEMBER *TF *TR))

Question?  CODE for the rule RTFMB
CONDITIONS:
((TRACK *TR)
 (INREGION *TR)
 (ORFNUM *TR *N))

ACTIONS:
((MEMBER RTF *TR)
 (ORFILE *TR *N))

Question?  Quit
Leaving EXPLAIN

     Message received from external source.
     Something detected at (26.0 53.0)  Time: 20
     Associated with track T2

Question?  Quit
Leaving EXPLAIN

A0052: T2 is  a TRACK.
A0053: SUBSURF is  a WCATEGORY of T2.
A0054: T2 is  a INREGION.

A0066: P2 is  a MEMBER of F0055.
A0065: T2 is  a MEMBER of ORF2.
A0064: P1 is  a MEMBER of F0055.
A0063: T2 is  a MEMBER of ORF1.
A0062: COMPLETE is  a STATUS of F0055.
A0061: F0055 is  a ORFILE of T2.
A0058: T2 is  a MEMBER of TF1.
A0057: T2 is  a MEMBER of RTF.
Question?  WHY is A0053
The information came directly from a message.
Question?  WHY is A0066
STAMMER applied the rule(s)
ORFILEMEMB
Question?  WHY is A0062
STAMMER applied the rule(s)
COMPL.TRKORF
Question?  WHY is A0061
STAMMER applied the rule(s)
RTFMB
```

```
Question?  WHY is A0058
STAMMER applied the rule(s)
TRKCTGMEMB
Question?  Quit
Leaving EXPLAIN

     Message received from external source.
     Something detected at (28.0 50.0)  Time: 35
     Associated with track T1

Question?  Quit
Leaving EXPLAIN

A0071: AD1 is  a ACOUSDATA of T1.
A0072: ICF is  a IMPOSCLASSFILE of AD1.
A0073: DELTA is  a MEMBER of ICF.

A0045: P1 is not known to be  a MEMBER of F0036.
A0044: T1 is not  a MEMBER of ORF1.
A0066: P2 is not known to be a MEMBER of F0055.
A0065: T2 is not  a MEMBER of ORF2.
A0079: T2 is  a TRACK of P1.
A0060: T2 is  a IMPOSTRACK of P2.
A0075: T1 is  a TRACK of P2.
A0040: T1 is  a IMPOSTRACK of P1.
Question?  WHY is A0045
STAMMER applied the rule(s)
ORFILEREDUC ORFILEMEMB
Question?  WHY is A0044
STAMMER applied the rule(s)
ORFILEREDUC ORFILEMEMB
Question?  PRINT the rule ORFILEREDUC
A track and a platform are removed from each others' OR-files if the
track is found to be an impossible-track of that platform.
Question?  CODE for the rule ORFILEREDUC
CONDITIONS:
((MEMBER RPF *P)
 (ORFILE *P *ORF)
 (MEMBER *ORF *TR)
 (IMPOSTRACK *P *TR)
 (ORFILE *TR *FRO))

ACTIONS:
((MEMBER *ORF *TR)
 (MEMBER *FRO *P))

Question?  HOW does rule ORFILEREDUC apply to  A0044
The rule was applied with the assertions

A0023: P1 is  a MEMBER of RPF.

A0025: ORF1 is  a ORFILE of P1.
```

```
A0044: T1 is somewhat likely to be  a MEMBER of ORF1.
       (condition is no longer true)

A0040: T1 is  a IMPOSTRACK of P1.

A0042: F0036 is  a ORFILE of T1.

Question?  WHY is A0079
STAMMER applied the rule(s)
THEN1PLATF
Question?  PRINT the rule THEN1PLATF
If a track has an OR-file that is complete and contains only one
platform, then it is a track of that platform.
Question?  CODE for the rule THEN1PLATF
CONDITIONS:
((MEMBER RTF *TR)
 (ORFILE *TR *FRO)
 (STATUS *FRO COMPLETE)
 (IMPOSTRACK *PR *TR)
 (MEMBERCOUNT1 *FRO)
 (MEMBER *FRO *P))

ACTIONS:
((TRACK *P *TR))

Question?  HOW does rule THEN1PLATF apply to  A0079
The rule was applied with the assertions

A0057: T2 is  a MEMBER of RTF.

A0061: F0055 is  a ORFILE of T2.

A0062: COMPLETE is  a STATUS of F0055.

A0060: T2 is  a IMPOSTRACK of P2.

A0078: F0055 is  a MEMBERCOUNT1.

A0064: P1 is  a MEMBER of F0055.

Question?  WHY is A0060
STAMMER applied the rule(s)
IMPOS.PLTF.ELIM
Question?  PRINT the rule IMPOS.PLTF.ELIM
A track is an impossible-track of a platform if it is the track of
another platform.
Question?  CODE for the rule IMPOS.PLTF.ELIM
CONDITIONS:
((MEMBER RPF *P)
 (TRACK *P *AT)
 (MEMBER RTF *TR)
```

```
 (*UNLESS* (IMPOSTRACK *P *TR))
 (*UNLESS* (SAME-AS *TR *AT)))

ACTIONS:
((IMPOSTRACK *P *TR))

Question?  HOW does rule IMPOS.PLTF.ELIM apply to  A0060
The rule was applied with the assertions

A0026: P2 is  a MEMBER of RPF.

A0075: T1 is  a TRACK of P2.

A0057: T2 is  a MEMBER of RTF.

A0060: T2 is not known to be  a IMPOSTRACK of P2.
       (no longer valid)

A0077: T2 is not the same as T1.

Question?  WHY is A0060
STAMMER applied the rule(s)
IMPOS.PLTF.ELIM
Question?  WHY is A0075
STAMMER applied the rule(s)
THEN1PLATF
Question?  WHY is A0040
STAMMER applied the rule(s)
IMPOSTRK.ACOUS
Question?  PRINT the rule IMPOSTRK.ACOUS
If the acoustic signature associated with a track cannot result from a
platform of a certain class, then the track is an impossible-track of
every platform of that class.
Question?  CODE for the rule IMPOSTRK.ACOUS
CONDITIONS:
((MEMBER RTF *TR)
 (WCATEGORY *TR SUBSURF)
 (ACOUSDATA *TR *AD)
 (IMPOSCLASSFILE *AD *ICF)
 (MEMBER RPF *P)
 (CLASS *P *CLS)
 (MEMBER *ICF *CLS))

ACTIONS:
((IMPOSTRACK *P *TR))

Question?  TELL me about P1
A0079: T2 is  a TRACK of P1.
A0064: P1 is  a MEMBER of F0055.
A0059: T2 is not known to be  a IMPOSTRACK of P1.
A0045: P1 is not known to be  a MEMBER of F0036.
A0040: T1 is  a IMPOSTRACK of P1.
```

```
A0025: ORF1 is  a ORFILE of P1.
A0024: P1 is  a MEMBER of PF1.
A0023: P1 is  a MEMBER of RPF.
A0009: P1 is a SUB.
A0008: SUBSURF is  a CATEGORY of P1.
A0007: P1 is a DELTA.
A0006: P1 is HOSTILE.
A0005: P1 is a KNOWNPLATFORM.
Question?  TELL me about P2
A0075: T1 is  a TRACK of P2.
A0066: P2 is not known to be  a MEMBER of F0055.
A0060: T2 is  a IMPOSTRACK of P2.
A0047: P2 is  a MEMBER of F0036.
A0041: T1 is not known to be  a IMPOSTRACK of P2.
A0028: ORF2 is  a ORFILE of P2.
A0027: P2 is  a MEMBER of PF1.
A0026: P2 is  a MEMBER of RPF.
A0014: P2 is a SUB.
A0013: SUBSURF is  a CATEGORY of P2.
A0012: P2 is a ECHOII.
A0011: P2 is HOSTILE.
A0010: P2 is  a KNOWNPLATFORM.
Question?  Quit
Leaving EXPLAIN
Cleaning up, please be patient and watch the mysterious output.

 TOPS-20 Command processor 4(546)
 <RBECHTAL>TAB.SET-NO-SPACES.6 [OK]
 STAMMER.LOG.1 [OK]
BBNA FTP User process 4(33)
* Connection opened.
 Assuming 36-bit connections, Paged transfers.
< USC-ISI FTP Server 1.44.11.0 - at Thu 4-Sep-80 14:32-PDT
* < Login completed.
*
*
 to remote-file < Store of <RLPT>ACCAT-TIP-13600002.STAMMER-TRACE;1;P77
-GEN started.
< Transfer completed.
**
 TEMP.FILE.1 [OK]
 STAMMER.LOG.1 [OK]
 PS:<RDILLARD> [8 pages freed]
 Thank you for your interest in the STAMMER system.
NIL
_load(rule-conf.lsp)
<RDILLARD>RULE-CONF.LSP.4
_(rule-confidences)
RTFMB:  1.0
TRKCTGMEMB:  1.0
IMPOSTRK.ACOUS:  1.0
ORFILEMEMB:  1.0
```

```
COMPL.TRKORF:  1.0
THEN1PLATF:  1.0
IMPOS.PLTF.ELIM:  1.0
ORFILEREDUC:  -1.0
NIL
_(logout)
@logo
Killed Job 47, User RDILLARD, Account 9160, TTY 334,
  at  4-Sep-80 17:44:46
  Used 4.42 KA-equivalent cpu minutes  in 0:33:35
Remote disconnect of 1
#disconnect
#
#quit
```

III. PTAPS in ROSIE -- Two-Submarine Scenario

```
#connection.to rand-ai is complete.#

 Rand-AI  Information Systems Laboratory, TOPS-20 Monitor 3A(2013)
@LOGIN NOSC
 Job 6 on TTY271 14-Apr-80 08:55:38
@<ROSIE>ROSIE
****ATTENTION USER NOSC:
this sysout is initialized for user ROSIE.
To reinitialize, type GREET()
>enable history;
>load rules.ptaps;
>display every platform;
    PLATFORM#1
        INSTANCE-1 is (PLATFORM)
        CLASS is DELTA
        CATEGORY is SUBSURF
        ORFILE is ORFILE#1
        IMPOSTRKFILE is IMPOSTRKFILE#1
    PLATFORM#2
        INSTANCE-1 is (PLATFORM)
        CLASS is ECHOII
        CATEGORY is SUBSURF
        ORFILE is ORFILE#2
        IMPOSTRKFILE is IMPOSTRKFILE#2
>display region1;
    REGION1
        LOCATION is PERSIAN-GULF
        STATUS is CONSTANTLY-MONITORED
        MAIN-FILES is (RPF RTF)
        SUBSET-FILES is (PLATFORMFILE#1 TRACKFILE#1)
>Create a track whose category is subsurf and whose region is
>region1;
>run;

The new track
"
    TRACK#1
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is ORFILE#3

"is put into RTF, the region's track file,
and an OR-file is opened for it:
"
    ORFILE#3
        INSTANCE-1 is (ORFILE)
    "
The new track
"
```

```
    TRACK#1
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is ORFILE#3
    "becomes a member of the trackfile of that category

"
    "
The platform and the track
"
    PLATFORM#1
        INSTANCE-1 is (PLATFORM)
        CLASS is DELTA
        CATEGORY is SUBSURF
        ORFILE is (TRACK#1)
        IMPOSTRKFILE is IMPOSTRKFILE#1
    TRACK#1
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is (PLATFORM#1)
    "are made members of each others' OR-files.

"
    "
The platform and the track
"
    PLATFORM#2
        INSTANCE-1 is (PLATFORM)
        CLASS is ECHOII
        CATEGORY is SUBSURF
        ORFILE is (TRACK#1)
        IMPOSTRKFILE is IMPOSTRKFILE#2
    TRACK#1
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is (PLATFORM#1 PLATFORM#2)
    "are made members of each others' OR-files.

"
    "
The OR-file of the track
"
    TRACK#1
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is (PLATFORM#1 PLATFORM#2)
        ORFILE-STATUS is COMPLETE
```

```
"is complete because the track is a member of a track-file
whose corresponding platform-file is complete.
"
>what put track#1 into RTF?
A: At cycle #32 RTFMEMB fired.
The rule is:

IF THERE IS A TRACK TR WHICH IS NOT IN RTF
AND WHOSE REGION IS REGION1
THEN FOR EACH CASE PUT TR INTO RTF
AND CREATE AN ORFILE FRO
AND SET THE ORFILE OF TR TO FRO
AND DISPLAY "
The new track
"
AND DISPLAY TR
AND DISPLAY
"is put into RTF, the region's track file,
and an OR-file is opened for it:
"
AND DISPLAY FRO ;
     The TRACK TR was TRACK#1
>display every trackfile whose category is subsurf;
    TRACKFILE#1
        INSTANCE-1 is (TRACKFILE)
        CATEGORY is SUBSURF
        BIN is (TRACK#1)
>what put track#1 into the bin of trackfile#1?
B: At cycle #34 TRKCTGMEMB fired.
The rule is:

IF THERE IS A TRACK TR WHICH IS IN RTF
AND THERE IS A TRACKFILE TF
AND TR IS NOT IN THE BIN OF TF
AND THE CATEGORY OF TR IS THE CATEGORY OF TF
THEN FOR EACH CASE PUT TR INTO THE BIN OF TF
AND DISPLAY "
The new track
"
AND DISPLAY TR
AND DISPLAY "becomes a member of the trackfile of that category

" ;
     The TRACKFILE TF was TRACKFILE#1
     The TRACK TR was TRACK#1
>what put track#1 into the orfile of platform#1?
C: At cycle #38 ORFMEMB fired.
The rule is:

IF THERE IS A PLATFORM P WHICH IS IN RPF
AND THERE IS A TRACK TR WHICH IS IN RTF
AND TR IS NOT IN THE IMPOSTRKFILE OF P
```

```
AND TR IS NOT IN THE ORFILE OF P
THEN FOR EACH CASE PUT TR INTO THE ORFILE OF P
AND PUT P INTO THE ORFILE OF TR
AND DISPLAY "
The platform and the track
"
AND DISPLAY P
AND DISPLAY TR
AND DISPLAY "are made members of each others' OR-files.

" ;
     The TRACK TR was TRACK#1
     The PLATFORM P was PLATFORM#2
>why?
D.1: RPF was (PLATFORM#2 PLATFORM#1)
D.2: RTF was (TRACK#1)
D.3: The IMPOSTRKFILE of PLATFORM#2 was IMPOSTRKFILE#2
D.4: The ORFILE of PLATFORM#2 was ORFILE#2
>why d.3?
E: At cycle #18 a typed-in rule fired.
The rule is:
CREATE A PLATFORM P2 WHOSE CLASS IS ECHOII
AND WHOSE CATEGORY IS SUBSURF
AND WHICH IS IN RPF
AND CREATE AN ORFILE ORF2
AND SET THE ORFILE OF P2 TO ORF2
AND CREATE AN IMPOSTRKFILE IPF2
AND SET THE IMPOSTRKFILE OF P2 TO IPF2 ;
>what set the orfile-status of track#1?
G: At cycle #46 COMPL.TRKORF fired.
The rule is:

IF THERE IS A PLATFORMFILE PF WHOSE STATUS IS COMPLETE
AND THERE IS A CORRESPFILE CF OF PF
AND THERE IS A TRACK TR WHICH IS IN THE BIN OF CF
AND THE ORFILE-STATUS OF TR IS NOT KNOWN
THEN FOR EACH CASE SET THE ORFILE-STATUS OF TR TO COMPLETE
AND DISPLAY "
The OR-file of the track
"
AND DISPLAY TR
AND DISPLAY
"is complete because the track is a member of a track-file
whose corresponding platform-file is complete.
" ;
     The CORRESPFILE CF was TRACKFILE#1
     The PLATFORMFILE PF was PLATFORMFILE#1
     The TRACK TR was TRACK#1
>why?
H.1: The STATUS of PLATFORMFILE#1 was COMPLETE
H.2: The BIN of TRACKFILE#1 was (TRACK#1)
H.3: The ORFILE-STATUS of TRACK#1 was  not known
```

```
>Create a track whose category is subsurf and whose region is
>region1;
>run;

The new track
"
    TRACK#2
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is ORFILE#4

"is put into RTF, the region's track file,
and an OR-file is opened for it:
"
    ORFILE#4
        INSTANCE-1 is (ORFILE)
    "
The new track
"
    TRACK#2
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is ORFILE#4
    "becomes a member of the trackfile of that category

"
    "
The platform and the track
"
    PLATFORM#1
        INSTANCE-1 is (PLATFORM)
        CLASS is DELTA
        CATEGORY is SUBSURF
        ORFILE is (TRACK#1 TRACK#2)
        IMPOSTRKFILE is IMPOSTRKFILE#1
    TRACK#2
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is (PLATFORM#1)
    "are made members of each others' OR-files.

"
    "
The platform and the track
"
    PLATFORM#2
        INSTANCE-1 is (PLATFORM)
        CLASS is ECHOII
        CATEGORY is SUBSURF
```

```
            ORFILE is (TRACK#1 TRACK#2)
            IMPOSTRKFILE is IMPOSTRKFILE#2
        TRACK#2
            INSTANCE-1 is (TRACK)
            CATEGORY is SUBSURF
            REGION is REGION1
            ORFILE is (PLATFORM#1 PLATFORM#2)
        "are made members of each others' OR-files.

    "
        "
    The OR-file of the track
    "
        TRACK#2
            INSTANCE-1 is (TRACK)
            CATEGORY is SUBSURF
            REGION is REGION1
            ORFILE is (PLATFORM#1 PLATFORM#2)
            ORFILE-STATUS is COMPLETE

    "is complete because the track is a member of a track-file
    whose corresponding platform-file is complete.
    "
    >create an acousdata ad and set the acousdata of track#1 to ad
    >and create an imposclassfile icf
    >and set the imposclassfile of ad to icf
    >and put Delta into the imposclassfile of ad;
    >run;

    Because of its acoustic signature, the track
    "
        TRACK#1
            INSTANCE-1 is (TRACK)
            CATEGORY is SUBSURF
            REGION is REGION1
            ORFILE is (PLATFORM#1 PLATFORM#2)
            ORFILE-STATUS is COMPLETE
            ACOUSDATA is ACOUSDATA#1
        "is an impossible-track of the platform
    "
        PLATFORM#1
            INSTANCE-1 is (PLATFORM)
            CLASS is DELTA
            CATEGORY is SUBSURF
            ORFILE is (TRACK#1 TRACK#2)
            IMPOSTRKFILE is (TRACK#1)
        "
    The track and the platform
    "
        TRACK#1
            INSTANCE-1 is (TRACK)
            CATEGORY is SUBSURF
```

```
        REGION is REGION1
        ORFILE is (PLATFORM#2)
        ORFILE-STATUS is COMPLETE
        ACOUSDATA is ACOUSDATA#1
    PLATFORM#1
        INSTANCE-1 is (PLATFORM)
        CLASS is DELTA
        CATEGORY is SUBSURF
        ORFILE is (TRACK#2)
        IMPOSTRKFILE is (TRACK#1)

"are removed from each others' OR-files because the
track was found to be an impossible track of that platform.

"
    "
The And-then-there-was-one platform rule fires.
The track
"
    TRACK#1
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is (PLATFORM#2)
        ORFILE-STATUS is COMPLETE
        ACOUSDATA is ACOUSDATA#1
    "is the track of the platform
"
    PLATFORM#2
        INSTANCE-1 is (PLATFORM)
        CLASS is ECHOII
        CATEGORY is SUBSURF
        ORFILE is (TRACK#1 TRACK#2)
        IMPOSTRKFILE is IMPOSTRKFILE#2
        TRACK is TRACK#1
    "
The track
"
    TRACK#2
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is (PLATFORM#1 PLATFORM#2)
        ORFILE-STATUS is COMPLETE
    "is an impossible track of the platform
"
    PLATFORM#2
        INSTANCE-1 is (PLATFORM)
        CLASS is ECHOII
        CATEGORY is SUBSURF
        ORFILE is (TRACK#1 TRACK#2)
        IMPOSTRKFILE is (TRACK#2)
```

```
        TRACK is TRACK#1
    "because it is the track of another platform.

"
    "
The track and the platform
"
    TRACK#2
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is (PLATFORM#1)
        ORFILE-STATUS is COMPLETE
    PLATFORM#2
        INSTANCE-1 is (PLATFORM)
        CLASS is ECHOII
        CATEGORY is SUBSURF
        ORFILE is (TRACK#1)
        IMPOSTRKFILE is (TRACK#2)
        TRACK is TRACK#1

"are removed from each others' OR-files because the
track was found to be an impossible track of that platform.

"
    "
The And-then-there-was-one platform rule fires.
The track
"
    TRACK#2
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is (PLATFORM#1)
        ORFILE-STATUS is COMPLETE
    "is the track of the platform
"
    PLATFORM#1
        INSTANCE-1 is (PLATFORM)
        CLASS is DELTA
        CATEGORY is SUBSURF
        ORFILE is (TRACK#2)
        IMPOSTRKFILE is (TRACK#1)
        TRACK is TRACK#2
>what put track#1 into the impostrkfile of platform#1?
I: At cycle #85 IMPOSTRK.ACOUS fired.
The rule is:

IF THERE IS A TRACKFILE TF WHOSE CATEGORY IS SUBSURF
AND THERE IS A TRACK TR WHICH IS IN THE BIN OF TF
AND THERE IS A PLATFORMFILE PF WHOSE CATEGORY IS SUBSURF
AND THERE IS A PLATFORM P WHICH IS IN THE BIN OF PF
```

```
AND THERE IS AN ACOUSDATA AD OF TR
AND THE CLASS OF P IS IN THE IMPOSCLASSFILE OF AD
AND TR IS NOT IN THE IMPOSTRKFILE OF P
THEN FOR EACH CASE PUT TR INTO THE IMPOSTRKFILE OF P
AND DISPLAY "
Because of its acoustic signature, the track
"
AND DISPLAY TR
AND DISPLAY "is an impossible-track of the platform
"
AND DISPLAY P ;
     The ACOUSDATA AD was ACOUSDATA#1
     The PLATFORM P was PLATFORM#1
     The PLATFORMFILE PF was PLATFORMFILE#1
     The TRACK TR was TRACK#1
     The TRACKFILE TF was TRACKFILE#1
>why?
J.1: The CATEGORY of TRACKFILE#1 was SUBSURF
J.2: The BIN of TRACKFILE#1 was (TRACK#1 TRACK#2)
J.3: The CATEGORY of PLATFORMFILE#1 was SUBSURF
J.4: The BIN of PLATFORMFILE#1 was (PLATFORM#1 PLATFORM#2)
J.5: The CLASS of PLATFORM#1 was DELTA
J.6: The IMPOSCLASSFILE of ACOUSDATA#1 was (DELTA)
J.7: The IMPOSTRKFILE of PLATFORM#1 was IMPOSTRKFILE#1
>what removed track#1 from the orfile of platform#1?
REMOVED => REMOVE ? yes

Syntax error after WHAT REMOVE
NIL
>show orfilereduc;
The rule is:

IF THERE IS A PLATFORM P WHICH IS IN RPF
AND THERE IS A TRACK TR WHICH IS IN THE ORFILE OF P
AND TR IS IN THE IMPOSTRKFILE OF P
THEN FOR EACH CASE REMOVE TR FROM THE ORFILE OF P
AND REMOVE P FROM THE ORFILE OF TR
AND DISPLAY "
The track and the platform
"
AND DISPLAY TR
AND DISPLAY P
AND DISPLAY
"are removed from each others' OR-files because the
track was found to be an impossible track of that platform.

" ;
>what set the track of platform#2?
K: At cycle #97 THEN1PLATF fired.
The rule is:

IF THERE IS A TRACK TR WHICH IS IN RTF
```

```
AND THE ORFILE-STATUS OF TR IS COMPLETE
AND LENGTH ( ORFILE OF TR ) IS 1
AND THERE IS A PLATFORM P WHICH IS IN THE ORFILE OF TR
AND THE TRACK OF P IS NOT KNOWN
THEN FOR EACH CASE SET THE TRACK OF P TO TR
AND DISPLAY "
The And-then-there-was-one platform rule fires.
The track
"
AND DISPLAY TR
AND DISPLAY "is the track of the platform
"
AND DISPLAY P ;
     The TRACK TR was TRACK#1
     The PLATFORM P was PLATFORM#2
>why?
L.1: RTF was (TRACK#1 TRACK#2)
L.2: The ORFILE-STATUS of TRACK#1 was COMPLETE
L.3: The ORFILE of TRACK#1 was (PLATFORM#2)
L.4: The TRACK of PLATFORM#2 was  not known
>what put track#2 into the impostrkfile of platform#2?
M: At cycle #103 IMPOS.PLTF.ELIM fired.
The rule is:

IF THERE IS A PLATFORM P WHICH IS IN RPF
AND THERE IS A TRACK ATR WHICH IS THE TRACK OF P
AND THERE IS A TRACK TR WHICH IS IN RTF
AND TR IS NOT ATR
AND TR IS NOT IN THE IMPOSTRKFILE OF P
THEN FOR EACH CASE PUT TR INTO THE IMPOSTRKFILE OF P
AND DISPLAY "
The track
"
AND DISPLAY TR
AND DISPLAY "is an impossible track of the platform
"
AND DISPLAY P
AND DISPLAY "because it is the track of another platform.

" ;
     The TRACK TR was TRACK#2
     The TRACK ATR was TRACK#1
     The PLATFORM P was PLATFORM#2
>display every track;
    TRACK#1
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is (PLATFORM#2)
        ORFILE-STATUS is COMPLETE
        ACOUSDATA is ACOUSDATA#1
    TRACK#2
```

```
        INSTANCE-1 is (TRACK)
        CATEGORY is SUBSURF
        REGION is REGION1
        ORFILE is (PLATFORM#1)
        ORFILE-STATUS is COMPLETE
>display every platform;
    PLATFORM#1
        INSTANCE-1 is (PLATFORM)
        CLASS is DELTA
        CATEGORY is SUBSURF
        ORFILE is (TRACK#2)
        IMPOSTRKFILE is (TRACK#1)
        TRACK is TRACK#2
    PLATFORM#2
        INSTANCE-1 is (PLATFORM)
        CLASS is ECHOII
        CATEGORY is SUBSURF
        ORFILE is (TRACK#1)
        IMPOSTRKFILE is (TRACK#2)
        TRACK is TRACK#1
>display every trackfile;
    RTF
        INSTANCE-1 is (TRACKFILE)
    TRACKFILE#1
        INSTANCE-1 is (TRACKFILE)
        CATEGORY is SUBSURF
        BIN is (TRACK#1 TRACK#2)
>display every platformfile;
    RPF
        INSTANCE-1 is (PLATFORMFILE)
    PLATFORMFILE#1
        INSTANCE-1 is (PLATFORMFILE)
        CATEGORY is SUBSURF
        STATUS is COMPLETE
        CORRESPFILE is TRACKFILE#1
        BIN is (PLATFORM#1 PLATFORM#2)
>bye;
((PARSE . 4.03) (RUN . 39.808) (RULES . 33))
_(logout)
@logout
Killed Job 6, User NOSC, Account , TTY 271,
  at 14-Apr-80 09:17:30,  Used 0:1:26 in 0:21:52
```

REFERENCES

1. NOSC TD 288, Higher Order Logic for Platform Identification in a Production System, by R. A. Dillard, 17 October 1979.

2. NOSC TR 364, New Methodologies for Automated Data Fusion, by R. A. Dillard, September 1978.

3. NOSC TD 324, Natural Language Processing Applied to Navy Tactical Messages, by Davis M. Keirsey (Systems Development Corporation), February 1980.

4. Dillard, R. A., Text-Understanding Techniques Applied to Partly Formatted Navy Tactical Messages, NOSC TD, in preparation.

5. NOSC TD 252, STAMMER: System for Tactical Assessment of Multisource Messages, Even Radar, by R. J. Bechtel and P. H. Morris (Systems Development Corporation), May 1979.

6. NOSC TD 298, Vol 1 and 2, STAMMER 2: A Production System for Tactical Situation Assessment, by D. C. McCall (NOSC), P. H. Morris, D. F. Kibler, and R. J. Bechtel (Systems Development Corporation), October 1979.

7. RAND Corporation report N-1158-1-ARPA, Design for a Rule-Oriented System for Implementing Expertise, by D. A. Waterman, R. H. Anderson, F. Hayes-Roth, P. Klahr, G. Martins, and S. J. Rosenscheim, May 1979.