

511-32
38274

Sequence-of-Events-Driven Automation of the Deep Space Network

R. Hill, Jr., K. Fayyad, C. Smyth, T. Santos, and R. Chen
Applications Development Section

S. Chien
Information Systems Technology Section

R. Bevan
Communications Systems and Research Section

In February 1995, sequence-of-events (SOE)-driven automation technology was demonstrated for a Voyager telemetry downlink track at DSS 13. This demonstration entailed automated generation of an operations procedure (in the form of a temporal dependency network) from project SOE information using artificial intelligence planning technology and automated execution of the temporal dependency network using the link monitor and control operator assistant system. This article describes the overall approach to SOE-driven automation that was demonstrated, identifies gaps in SOE definitions and project profiles that hamper automation, and provides detailed measurements of the knowledge engineering effort required for automation.

I. Introduction

The Voyager 1 spacecraft is cruising at 17.5 km/s toward the outer edge of the solar system. Though its onboard systems are mostly asleep during this phase of its mission, Voyager's health metrics are continually sent to Earth via a telemetry signal radiated by its 40-W transmitter. It will take 8 hours at the speed of light for the signal to reach its destination, Earth, a billion miles away. Upon arrival, the telemetry signal is received by an extremely sensitive ground communications system, NASA's Deep Space Network (DSN), where it is recorded, processed, and sent to the mission operations and Voyager project engineers, who assess the health of the spacecraft based on the contents of the signal.

The type of activity just described occurs daily for dozens of different spacecraft and projects. Though the process of sending signals from a spacecraft to Earth is conceptually simple, in reality there are many Earth-side challenges, both technical and institutional, that must be addressed before a spacecraft's signal is acquired and transformed into useful information. The purpose of this article is to identify some of these challenges and propose ways of automating the process of capturing data from a spacecraft. In particular, this article focuses on how to automatically transform a flight project service request into an executable set of DSN operations that fulfill the request. To understand in more concrete terms the processes we propose to improve, consider first the following example of the typical activities that currently occur

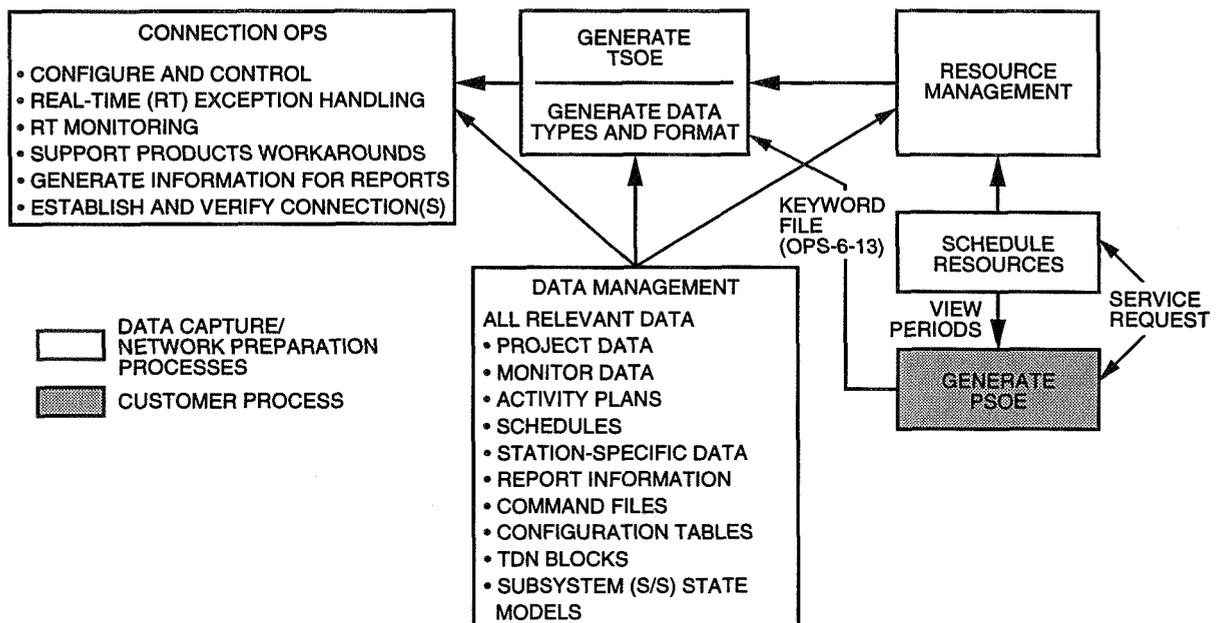


Fig. 1. Partial process map for network preparation and data capture processes.

when the DSN provides a service in response to a project request. Figure 1 shows a partial process map of the steps taken to fulfill the service request.¹

A team within the Voyager flight project decides to perform a periodic check of the health status of the Voyager 1 spacecraft. Since the spacecraft's health metrics are sent via a telemetry signal, a project representative submits a service request to the Telecommunications and Mission Operation Directorate (TMOD) (see Fig. 1) for a telemetry downlink activity to be performed, and the time, day, and equipment are negotiated with a representative from the DSN. Once the activity has been scheduled (Fig. 1, "schedule resources"), the Voyager team develops a project sequence of events (PSOE),² which is a time-ordered description of the events that should take place during the activity (Fig. 1, "generate PSOE"). The PSOE is written in a high-level language³ that provides a set of keywords and parameters to specify DSN functions and spacecraft events. Some of the keywords used in the SOE are actions that must be performed by the DSN, e.g., configure the receiver or acquire a downlink. Other keywords describe actions that will be taken by the spacecraft, e.g., a change in bit rate or the modulation index during the track activity. Once completed, the Voyager team sends the SOE to DSN operations, whose job it is to implement the requested activities.

After DSN operations receives the SOE keyword file, which is a high-level description of the events and activities that must be performed, operations personnel supplement the PSOE with additional information

¹ This is a modified version of the full process maps found in *Final Report of the Reengineering Team (RET) for Services Fulfillment (Operations)* (internal document), Telecommunications and Missions Operations Directorate (TMOD), Jet Propulsion Laboratory, Pasadena, California, March 14, 1995.

² This is a simplified description of the actual process. In reality, a project develops an integrated PSOE describing all of the activities planned for its spacecraft for a 2-week period. The integrated SOE is usually developed using the project's own language and later converted into the DSN keyword format described above. There are currently some efforts within the DSN to standardize the set of services it offers to its customers. With all of the translation required to go from the project language to the DSN keyword format, it seems desirable to develop an SOE language or tool that can be reused by multiple flight projects and that also would eliminate the need to translate into the DSN format. This is currently being addressed by the SOE reengineering team.

³ *Flight Project Interface to the DSN for Sequence of Events Generation*, DSN Document 820-13, OPS-6-13 (internal document), Jet Propulsion Laboratory, Pasadena, California, August 31, 1995.

concerning resource allocation and activity-specific configuration (Fig. 1, “generate TSOE” and “generate data types and formats”).⁴ The TSOE (which stands for Telecommunications and Mission Operations Directorate Ground Network sequence of events) is sent to the deep-space communications complex, where operators read and transform its contents into a set of operations (ops) procedures, which, when executed, will satisfy the goals stated in the TSOE (Fig. 1, “connection ops”). Based on the keywords in the TSOE, the operator chooses procedures to accomplish the goals of the activity and executes them by manually typing and sending command directives to the subsystems in the assigned communications link. The execution of the issued directives is monitored by the operator, who closes the control loop by determining whether the directive was successful or not. By “successful” execution, we mean that the directive had its intended effect on the subsystem. Thus, to successfully transform the TSOE from a list of time-tagged keywords into an operational procedure, the following steps are performed:

- (1) Determine the appropriate directive(s) and parameters to achieve the current keyword goal
- (2) Determine that the necessary conditions required for the execution of the directive are satisfied (i.e., verify preconditions of the directive)
- (3) Execute the directive (manual and computer-initiated activities)
- (4) Verify that the directive was correctly received by the subsystem
- (5) Verify that the desired subsystem state has changed (verify successful accomplishment of the postconditions of the directive)

These are the steps taken under normal circumstances. In addition, the operator monitors the overall progress of the activity and the health status of the subsystems by viewing a host of different computer displays. Exceptions must be handled in real time by the operator, who invokes recovery procedures when there are equipment anomalies or failures. Thus, from the start to the finish, the process of capturing data from a spacecraft currently involves many manual steps.⁵

II. Automated DSN Operations—The Concept

While current DSN operations are extremely labor- and knowledge-intensive, the DSN Technology Program has been investing in the groundwork for a new model of operations that automates major portions of the DSN data capture and network preparation processes. In this new model, generic services are electronically requested of the DSN by a project. These services are then provided to the project in a manner completely transparent to the project. Within the DSN services fulfillment process, routine operations will be completely automated, greatly enhancing the flexibility and responsiveness of the DSN. Furthermore, many of the costly, knowledge- and labor-intensive processes will be automated, resulting in a more efficient, streamlined DSN operations structure.

The Network Automation Work Area, funded by the DSN Technology Program, has been developing and demonstrating technology to fulfill this vision of automated data capture and network preparation. This work and these demonstrations can be described in terms of three areas (see Fig. 2): resource allocation—assignment of global DSN resources necessary to fulfill the requests; procedure

⁴ Again, this is a simplified account of the actual process; the project SOE keyword file requires additional information before it is sent to the deep-space tracking station for implementation. The SOE also triggers the generation of predict and support data needed by various subsystems during the activity, e.g., the antenna needs pointing predicts so that it will be oriented to receive the spacecraft signal.

⁵ The complete process map is in DSN Document 820-13, OPS-6-13, op cit.

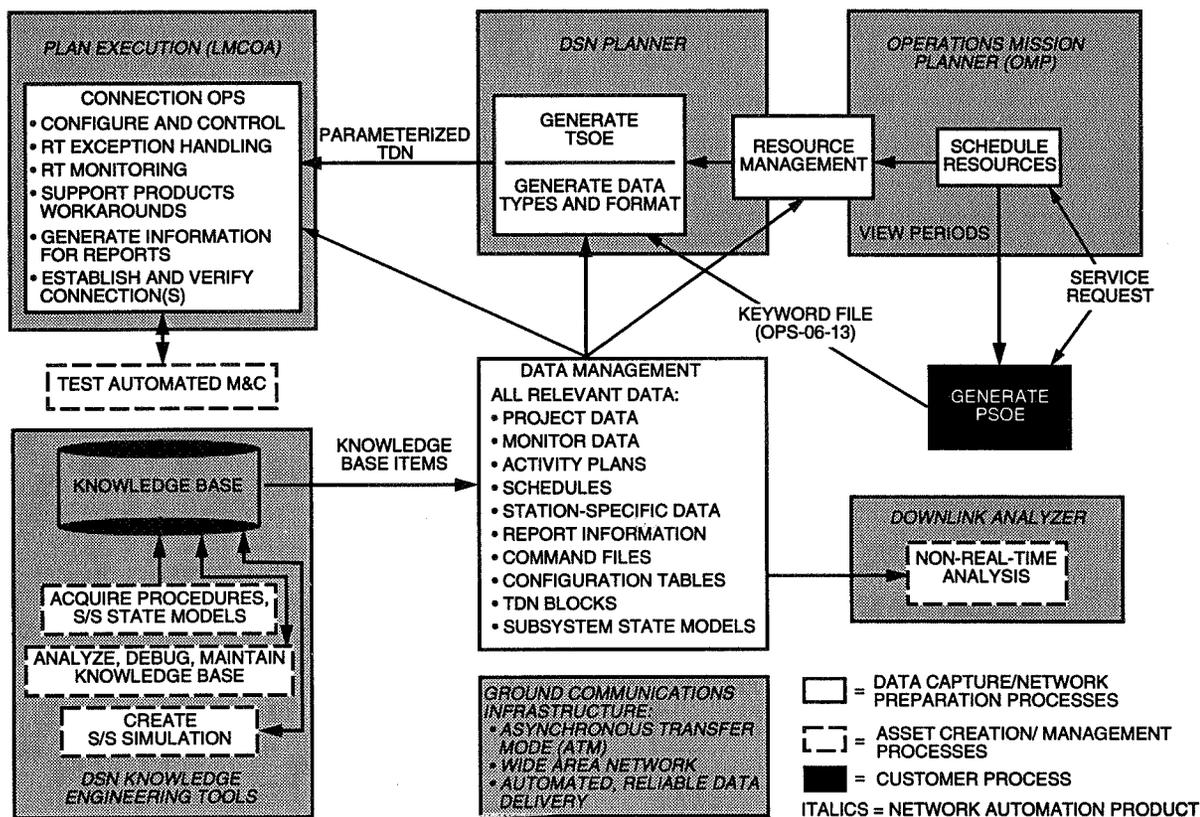


Fig. 2. Automating the DSN's data capture and network preparation processes.

generation—generation of appropriate executable procedures to fulfill specific service requests in light of the allocated equipment; and procedure execution—executing the appropriate procedure and ensuring that the appropriate services are correctly fulfilled. Below we outline in further detail ongoing efforts in each of these areas.

In the resource allocation area, the OMP-26 scheduling system has been fielded to support scheduling of the 26-m DSN antenna subnetwork. Current efforts involve extending the OMP-26 system to schedule further classes of DSN assets (extension beyond the 26-m subnet); expansion of OMP-26 coverage to the real-time area, where changing requests and equipment availability force modifications of the operational schedule; and further extension and maturation of the OMP-26 automated scheduling capability.

In the procedure generation area, we have been adapting planning technology that has been proven and fielded in science data processing applications to the problem of automated generation of antenna operations procedures. This work involves development of the DSN planner (DPLAN) system, which generates an executable antenna operations procedure, called a temporal dependency network (TDN) [3], from a high-level track specification and equipment assignment.

In the procedure execution area, the link monitor control operator assistant (LMCOA) system is being further refined and generalized to support automated tracking using DSN antennas and subsystems. The LMCOA system provides closed-loop control, execution monitoring, and error recovery to provide robust procedure execution for data capture. LMCOA has been used for the Ka-Band Antenna Performance (KaAP) experiments, the Jupiter Patrol, and the phased-array experiment at DSS 13.

III. SOE-Driven Automation Demonstration

A. Overview and Objectives

In February 1995, a comprehensive demonstration was conducted to validate the concept of end-to-end SOE-driven automation. In this demonstration, the procedure generation and procedure execution elements of automation were demonstrated by performing a Voyager telemetry downlink track using the 34-m beam-waveguide (BWG) antenna at DSS 13 in Goldstone, California. There were four major objectives to this demonstration:

- (1) To demonstrate that an executable operations procedure could be generated using the DPLAN automated planning system [7].
- (2) To demonstrate that the TDN generated by DPLAN could be correctly executed by the LMCOA procedure execution system to successfully perform the requested data capture process.
- (3) To further understand the institutional and infrastructure difficulties in fielding technologies for (1) and (2). Specifically, what types of information are currently missing from SOEs and affiliated request information that would be needed to automatically generate, parameterize, and execute TDNs? Additionally, how much effort in knowledge engineering was required to endow DPLAN and LMCOA with the capability for this track? How much effort would it take to extend to a related track for a different project? To a different type of track?
- (4) To acquire both qualitative and quantitative data on the amount of effort required to perform knowledge engineering to field systems for tasks (1) and (2) above [4,6,8,10].

The remainder of this section outlines at a high level how each of these capabilities was demonstrated and describes the interrelations between different elements of the demonstration. The remaining sections in this article describe the underlying technologies and representations.

The first goal of the end-to-end demonstration was to show automated generation of DSN operations procedures. In this element of the demonstration, DPLAN used the PSOE, equipment allocation, and a project profile file to construct a TDN that could be executed to perform the track. Generating the TDN involves

- (1) Determination of the correct TDN blocks (to achieve the tracking goals in the context of the particular equipment assignment and project profile)
- (2) Determination of the correct dependencies among blocks (e.g., block A must be completed before block B)
- (3) Determination of the correct parameters for each block (in the context of the tracking goals, equipment assignment, and project profile)

Some elements of TDN parameterization are deferred until plan execution time (to be performed by LMCOA). DPLAN uses information about how to achieve track goals using generic TDN blocks in the context of various equipment configurations and project parameters. This modular representation of operations procedures allows for a more compact knowledge base, removing the need to maintain a complete set of complex end-to-end TDNs to cover all combinations of track types, equipment configurations, and project classes—a set that may number in the hundreds. Each TDN block is a procedure executable by

the procedure execution engine, LMCOA, and consists of a sequence of subsystem directives tailored to achieve a higher-level goal in context of a specific type of track.

The second goal of the end-to-end demonstration was to exercise the automated procedure execution capability of the LMCOA system on the TDN produced by the DPLAN. This involved demonstrating the data capture capability of the LMCOA for a series of Voyager 2 downlink telemetry passes at DSS 13. While LMCOA has been in use since the fall of 1994 for KaAP and Jupiter Patrol tracking activities, tracking Voyager 2 represented the first time LMCOA had been used to track a spacecraft. Thus, one point of the demonstration was to show the generality of the LMCOA approach to both spacecraft and non-spacecraft tracking activities.

The third goal of the end-to-end demonstration was to characterize and quantify the effort required to implement the automation architecture. This effort can be attributed to two factors. The first part of the implementation effort involved determining elements of information and infrastructure that would impede implementation of an end-to-end automation system as demonstrated. This goal was furthered by requiring all types of project, spacecraft state, and sequence-related information to be contained in the project SOE and project profile. Because the project SOE is fixed, all mission information would need to be contained in the project profile. Thus, examining the types of information in the project profile provided valuable insight into information missing from the project SOEs. The second part of the effort was to determine how much knowledge engineering was required to implement the automation architecture. A careful accounting was made of the knowledge engineering effort required to construct and validate the DPLAN knowledge base (used in generating the TDN), the TDN blocks themselves, the parameterization of the TDN blocks by both DPLAN and LMCOA, and the directives that make up the TDN blocks used in the demonstration.

We think that the labor estimates we derived in constructing the TDNs represent a realistic estimate of the true effort required to construct operation TDNs. The labor estimates reflect generation, manual evaluation, and operation validation of the TDNs. All three of these steps are required to produce a TDN capable of providing reliable operations support. Other TDNs that have been constructed by other groups, while carefully constructed and manually verified, have not yet been validated by usage in an automation system.

B. Demonstration Context: Deep Space Station 13

Located at Goldstone, California, Deep Space Station 13 (DSS 13) is the deep-space tracking station dedicated to research and development activities of the Deep Space Network. DSS 13 has functional elements (see Fig. 3) comparable to DSN tracking stations: a 34-m BWG antenna, which is controlled by the antenna pointing subsystem; feed horns for S-band (2.3-GHz), X-band (8.45-GHz), and Ka-band (32-GHz) signals; a total power radiometer (TPR) for measuring broadband signal strength; a weather subsystem for tracking wind, humidity, and other atmospheric conditions affecting signal acquisition; a microwave switch controller subsystem; and a receiver, the TP-13 telemetry processor. These subsystems are connected to a local area network (LAN) along with a monitor and control (M&C) subsystem. The subsystems communicate with the M&C subsystem via a set of communications services implemented using manufacturing message specification (MMS) protocols. The subsystems communicate with one another by setting, sending, and reading MMS-variable structures.

The TP-13 receiver was integrated into the M&C LAN to support the Voyager telemetry downlink demonstration. Since most of the recent tracking activities at DSS 13 have not involved spacecraft, the telemetry processor was not at the station. Integrating TP-13 into DSS 13 involved writing an MMS-based interface for TP-13. At the same time, the TP-13 software was ported to the OS/2 operating system, requiring reimplementing of much of the low-level communication between the TP-13 software and hardware.

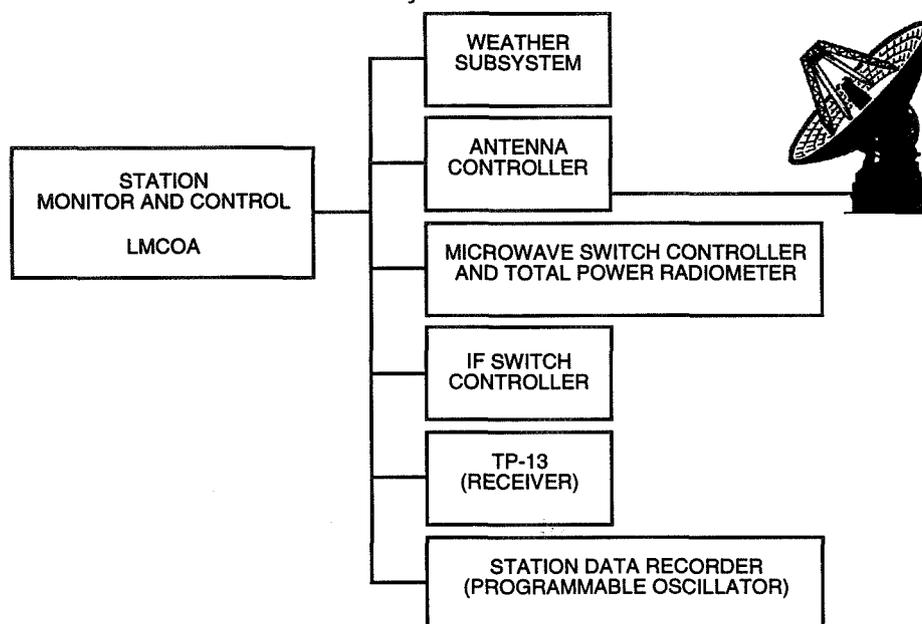


Fig. 3. DSS 13 configuration.

The station monitor and control (SMC) subsystem provides the human operator with an interface to the subsystems shown in Fig. 3. As the name suggests, the SMC subsystem is used by the operator to monitor the status of the subsystems and to exercise control over them during a track activity. The LMCOA provides the capability to automatically execute operations procedures encoded in TDNs. In the SOE-driven automation demonstration, the DPLAN automatically generated the operations procedure in TDN format, and the LMCOA executed it.

IV. Goal 1: Automated Procedure Generation

As part of the Voyager track demonstration, the capability to generate an executable operations procedure from high-level track information was demonstrated. The DPLAN planner uses high-level track information to determine appropriate steps, order constraints on these steps, and determine parameters of the steps to achieve the high-level track goals given the equipment allocation. In generating the TDN, the planner uses information from several sources:

- (1) Project SOE—The project SOE specifies events from the mission/project perspective. As a result, the project SOE contains a great deal of information regarding the spacecraft state that is relevant to the DSN track, as well as a large amount of spacecraft information unrelated to DSN operations. Relevant information specified in the project SOE includes such items as the one-way light time (OWLT) to the spacecraft, notifications of the beginning and ending times of tracks, spacecraft data transmission bit-rate changes, modulation index changes, and carrier and subcarrier frequency changes. An excerpt from a Voyager SOE is shown in Fig. 4.
- (2) Project profile—This file specifies project-specific information regarding frequencies and pass types. For example, the project SOE might specify a high frequency, and the project profile would specify the exact frequency used. The project profile might also specify other signal parameters and default track types. An excerpt from a Voyager project profile file is presented in Fig. 5.

- (3) LMCOA SOE (LSOE)—Using the project profile information, the project SOE is parsed into a more manageable form specifying only the information directly relevant to the specific pass. A sample LSOE for a Voyager telemetry downlink pass is shown in Fig. 6.
- (4) TDN knowledge base (KB)—The TDN knowledge base stores information on the TDN blocks available for DPLAN and LMCOA to use. This knowledge base includes information regarding preconditions, postconditions, directives, and other aspects of the TDN blocks. It also includes information on how to expand the block parameters and name into the actual flat file entry in a TDN.
- (5) Equipment configuration—This details the types of equipment available and unique identifiers to be used to specify the exact pieces of equipment to be used in the track. These include the antenna, antenna controller, the receiver, etc.

```

*ID=DSNISOE52B1      S/C=032 BEG=94 360 094000      END=95 001 221500
I032 42 00022 360 201500 ACE NCT S/C ANT      STATUS, RCP/00.100/Y, HGA, RCP/00.100/Y, HGA
I032 42 00022 360 201500 ACE NCT      S/C TXR      STATUS, OFF, EXC1/TWT2/18,
LOW,EXC1/TWT2/18
I032 42 00022 360 201500 ACE NCT      S/C RCVR      STATUS, 2/S, ON
I032 42 00022 360 201500 ACE NCT      S/C RNG      MODE, OFF, OFF, ON, OFF
I032 42 00022 360 201500 ACE NCT      S/C TLM X-    MOD, 160CD, LOW, 61DG
I032 42 00022 360 201500      D42      BOT
I032 42 00022 360 201500      D42      ACD D/L, 1-WAY, X
I032 42 00031 361 095000      D42      EOT

```

Fig. 4. Sample Voyager project SOE.

```

#      switch IF3 to S-band (5), IF4 to S-band (7)
IF3 S
IF4 X
#      LOW and HIGH subcarrier frequencies (kHz), from 870-14
SUB_LOW 22500
SUB_HIGH 360
PRED_POWER 25.6
SKY_FREQ 300000000
CARRIER_FREQ 300000000
CARRIER_UPDATE_RATE 500
CARRIER_BANDWIDTH 1
CARRIER_LOOP II
PC_NO 87.27
SUB_UPDATE_RATE 100
SUB_BANDWIDTH 1
SUB_LOOP II
PD_NO 275.42
SYMBOL_UPDATE_RATE 100
SYMBOL_BANDWIDTH 1
SYMBOL_LOOP II
ES_NO 0.871

```

Fig. 5. Excerpt from Voyager project profile file.

```

95/005/16:30:00 CONFIG: IF3=X_KA; POLAR=RCP; SKY_FREQ=300000000; PRED_POWER=25.6;
PO_FILE="/u/cm/lmcoa/lmcoa/src/kb/xv2_05.po";
95/005/16:30:00 TPR_PRECAL: BAND=X;
95/005/16:30:00 SOURCE: SRC=VGR2;
95/005/16:30:00 CARRIER: PC_NO=87.27; FREQ=300000000; BANDWIDTH=1.0; UPDATE_RATE=500;
LOOP=2;
95/005/16:30:00 SUBCARRIER: PD_NO=275.42; FREQ=22500; BANDWIDTH=1.0; UPDATE_RATE=100;
LOOP=2;
95/005/16:30:00 SYMBOL: ES_NO=0.871; BANDWIDTH=1.0; UPDATE_RATE=100; LOOP=2;
95/005/16:30:00 ACQ_D/L: MODE=1-WAY; BAND=X;
95/005/16:30:00 TELEM: BIT_RATE=160; CODE_FLAG=1; MOD_INDEX=61;
95/005/17:15:00 TELEM: BIT_RATE=2800; CODE_FLAG=1;
95/005/17:15:00 TELEM: MOD_INDEX=72;
95/005/18:30:00 TELEM: BIT_RATE=160; CODE_FLAG=1;
95/005/18:30:00 TELEM: MOD_INDEX=61;
95/005/20:00:00 END_OF_TRACK.

```

Fig. 6. Voyager downlink telemetry pass LSOE.

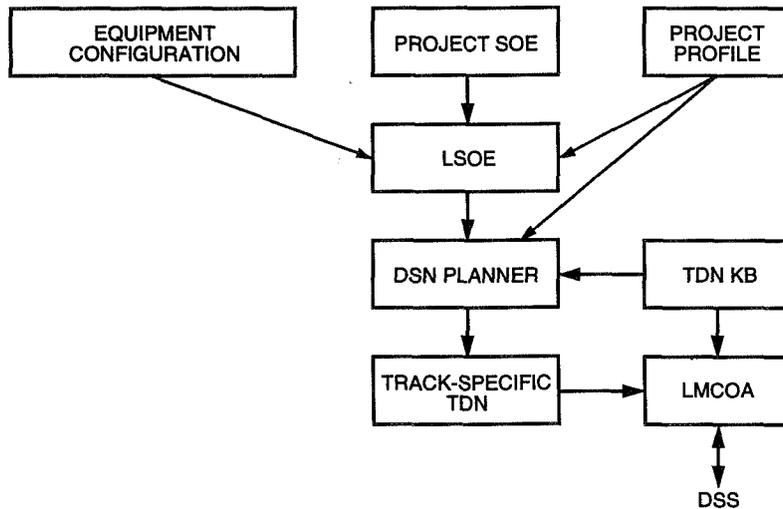


Fig. 7. SOE-driven automation architecture.

The overall flow of information in our approach to SOE-driven automation is shown in Fig. 7. In this approach, the project SOE is parsed into an LSOE that contains only that information relevant to the track of interest. DPLAN uses the LSOE and the project profile to determine the parameterized goals of the track. A sample set of track goals is shown in Fig. 8. This set of goals and a set of contextual facts derived from the LSOE are the planning inputs. The contextual facts are basically the planning system's representation of the LSOE information.

DPLAN reduces the high-level track goals into executable steps by applying knowledge about how to achieve specific combinations of track goals in the context of specific equipment combinations. This information is represented in the form of task reduction rules, which detail how a set of high-level goals can be reduced into a set of lower-level goals in a particular problem-solving context. Each task reduction rule rigorously details the scope of its expertise in terms of track and equipment combinations. This information on the scope of applicability of the rule can be considered in terms of a track goal

```

(PO_required track1)
(spacecraft_mode_changes track1)
(track_goal spacecraft_track telemetry track1)
(track_goal spacecraft_track downlink track1)
(track_goal decode_data)
(station-used track1 DSS13)

```

Fig. 8. Sample tracking goals.

hierarchy and equipment goal hierarchy, where the rule applies to all contexts below the rule in the relevant hierarchy (all specializations of its scope).

Figure 9 shows a partial tracking-goal hierarchy involving the goals for telemetry, commanding, and ranging. Figure 10 shows a partial track hierarchy for antennas. A rule might specify how to achieve a tracking goal for a particular type of antenna. For example, a rule might specify how to achieve the telemetry tracking goal for a 34-m BWG antenna. Alternatively, a rule might specify a constraint on how to achieve telemetry when using high-efficiency antennas (HEFs). For example, a rule might specify that two receiver calibration steps, A and B, might have to be ordered so that A is always before B. By representing the track, equipment, and other hierarchies, the scope of various pieces of knowledge regarding antenna track activities can be naturally and easily represented.

Using this problem specification, DPLAN then uses task reduction planning techniques, also called a hierarchical task network (HTN) [2,9], and operator-based planning techniques [11] to produce a parameterized track-specific TDN to be used to conduct the track. The actual planner used to generate the TDN is a modified version of the task reduction planning component of the multimission Video Image Communication and Retrieval (VICAR) planner system [1].⁶ This track-specific TDN and the LSOE can then be used by LMCOA to operate the actual antenna to conduct the requested antenna track.

For example, the task reduction rule shown in Fig. 11 states that in the equipment context of DSS 13 and the tracking goal context of a downlink track, the abstract task of precalibration can be achieved by performing the lower-level tasks of inspecting the subsystems, connecting the subsystems, configuring the TPR, loading antenna predicts, and configuring the receiver. Furthermore, the subsystems must be inspected before connecting the subsystems, and so on. Also, some of these tasks are composite tasks and will be expanded later into more detailed tasks. For example, configuring the TPR involves configuring the IF switch, configuring the microwave switch controller (UWC)/TPR for precalibration, and performing the actual TPR precalibration.

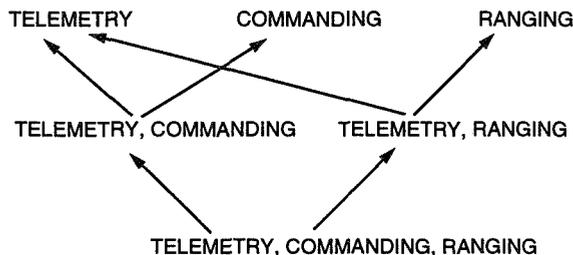


Fig. 9. Partial track hierarchy.

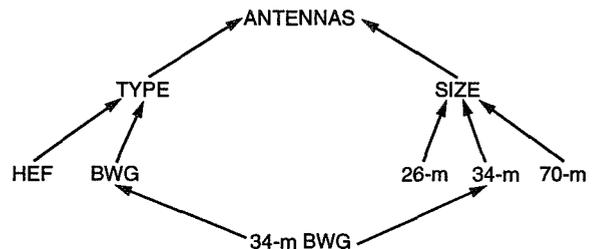


Fig. 10. Example equipment hierarchy—antenna types.

⁶ S. Chien, R. Hill, and K. Fayyad, "Why Real-World Planning Is Difficult," working notes of the 1994 Fall Symposium on Planning and Learning: On to Real Applications, New Orleans, Louisiana, November 1994.

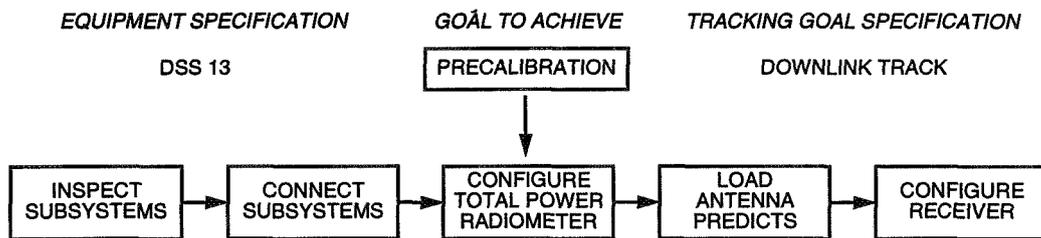


Fig. 11. Sample task reduction.

Next consider the task reduction rule shown in Fig. 12. It states that in the equipment context of DSS 13 and in any tracking goal context, the abstract goal of including a programmable oscillator (PO) can be achieved by adding the "load PO files" step followed by the "configure Doppler tuner" step. Additionally, these steps must be ordered with respect to the "connect subsystems" and "load antenna predicts" steps as indicated.

In the context of specific tracks, generic configuration blocks will be specialized as appropriate to the details of the track. For example, in the equipment context of DSS 13, if the track context is downlink, telemetry, with symbol decoding requested, the task reduction rule in Fig. 13 states the specific receiver configuration block to use to configure the receiver.

Considerable effort in computing the final TDN is devoted to determining the correct parameters (params) for blocks in the TDN. For example, Table 1 shows a configuration table used to determine the IF switch parameter for the TPR precalibration step. Depending on the communication bands used in the track, differing bands will be assigned to each of the communication pathways in the UWC/TPR. Based on the bands being used, the TPR IF switch parameter is also determined. This parameter setting is also determined during the decomposition process, so a correctly parameterized TDN can be constructed.

The application of decomposition rules continues until all of the activity blocks in the TDN are grounded—that is to say, known blocks in the TDN KB can be used to instantiate the activities. This fully instantiated TDN can then be used with the LSOE by LMCOA to perform the actual track. Shown in Fig. 14 is the TDN for a Voyager downlink telemetry track using the PO for DSS 13.

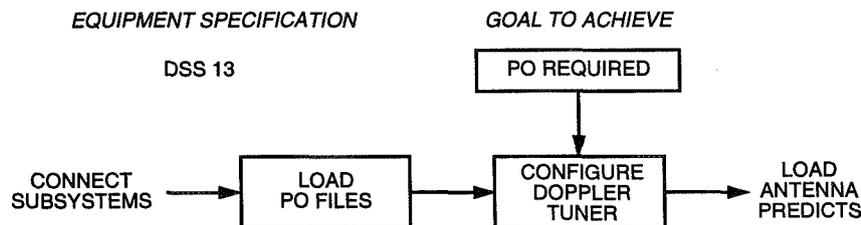


Fig. 12. Augmentation of the TDN required by an additional PO track goal.

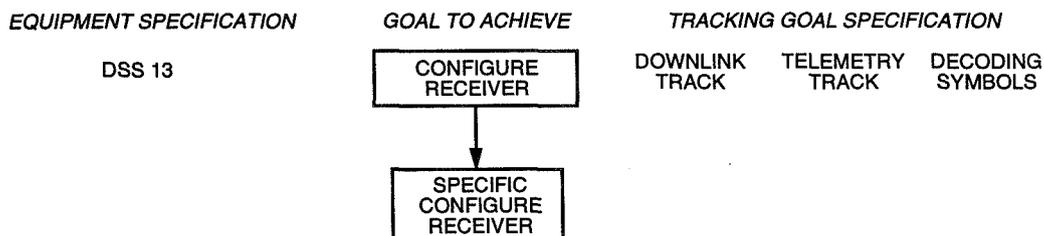


Fig. 13. A specialized task reduction.

Table 1. TPR IF switch parameter determination.

| Frequency | IF1 | IF2 | Parameter |
|-----------------|-----|-----|-----------|
| S-band | * | — | 1 |
| X-band | * | — | 1 |
| Ka-band | — | * | 2 |
| Q-band (45 GHz) | — | * | 2 |
| S- and X-band | * | * | 3 |
| X- and Ka-band | * | * | 3 |

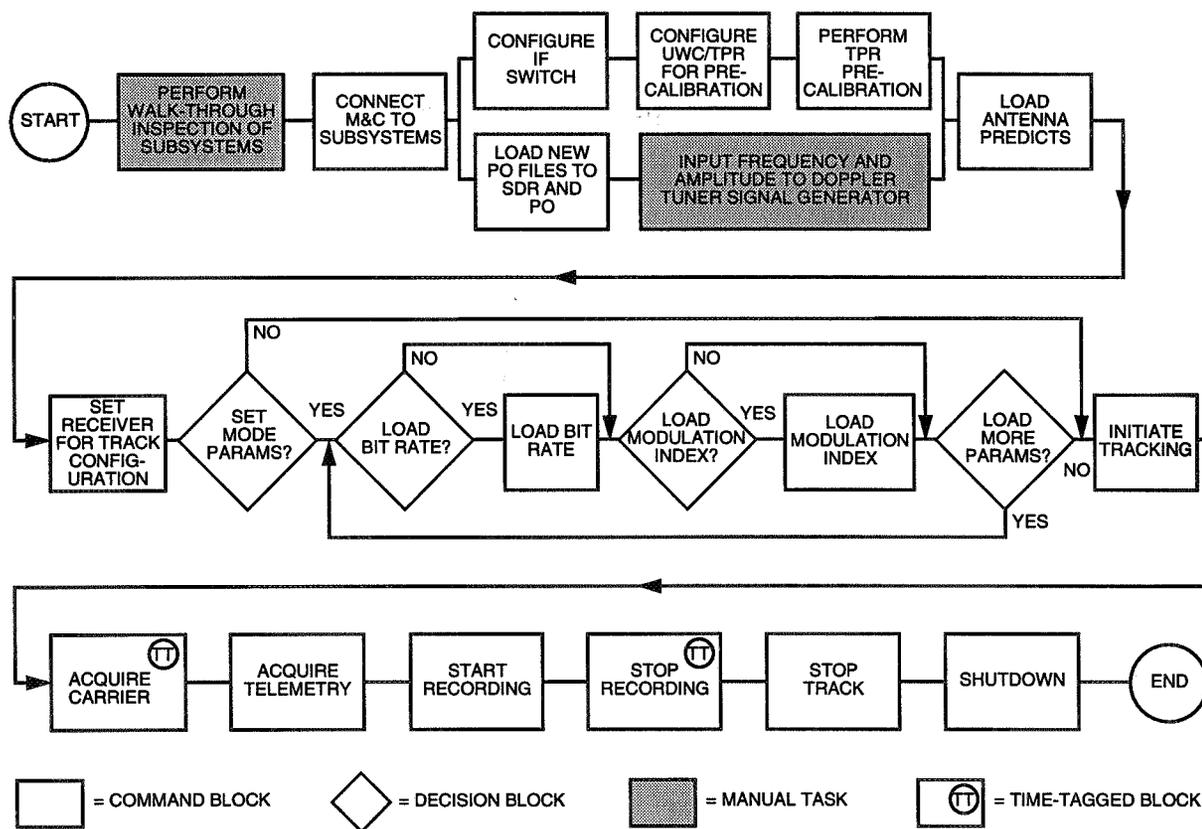


Fig. 14. The TDN for a Voyager telemetry downlink track using the DSS-13 antenna (including telemetry mode changes).

V. Goal 2: Automated Procedure Execution

The previous section described how DPLAN automatically composes a procedure, called a TDN, from a knowledge base of procedure components, called TDN blocks, based on a request for a DSN service. In this section, we describe how a TDN is executed at a deep-space tracking station, DSS 13, using the LMCOA. As the name suggests, the purpose of the Link Monitor and Control Operator Assistant is to assist DSN operators in performing tasks on the M&C subsystem. M&C tasks involve monitoring, configuring, operating, and troubleshooting deep-space communications equipment. The goal of automating the execution of procedures is to drastically reduce the amount of manual operations needed to support a DSN tracking activity by automating the execution of the operations procedures

for an activity. The benefits of achieving this goal include reducing the operator's workload, thereby allowing him or her to do other tasks, and reducing the amount of time required to perform a track.

In order to understand better how LMCOA works, a short description of the operational context in the DSS-13 M&C subsystem is given first. Then we give an overview of LMCOA's architecture along with a description of how the TDN is executed. Finally, we discuss the results of the demonstration in which LMCOA executed the TDN for the Voyager telemetry downlink activity.

A. Monitor and Control

The M&C subsystem is the centralized site for monitoring and controlling the subsystems that comprise a deep-space communications link. The M&C console displays subsystem status data to the operator and also provides a graphical user interface for controlling the subsystems; it provides all the control functions needed for an operator to perform a tracking activity. For instance, Fig. 15 shows the M&C interface for performing the boresight procedure on the antenna subsystem. The state of the antenna is displayed as a table of values for azimuth, elevation, and offset. Control over the subsystem is exercised by pressing graphical user interface (GUI) buttons, such as the "MiniCal" button shown in the display. When the M&C subsystem is used without LMCOA to perform a tracking activity, the operator manually executes a sequence of actions (i.e., pushing buttons, typing data into data entry fields, switching displays, etc.) to satisfy the requirements of the activity. All control is exercised by the operator, who must also monitor the state of the subsystems in the communications link in order to close the control loop [5].

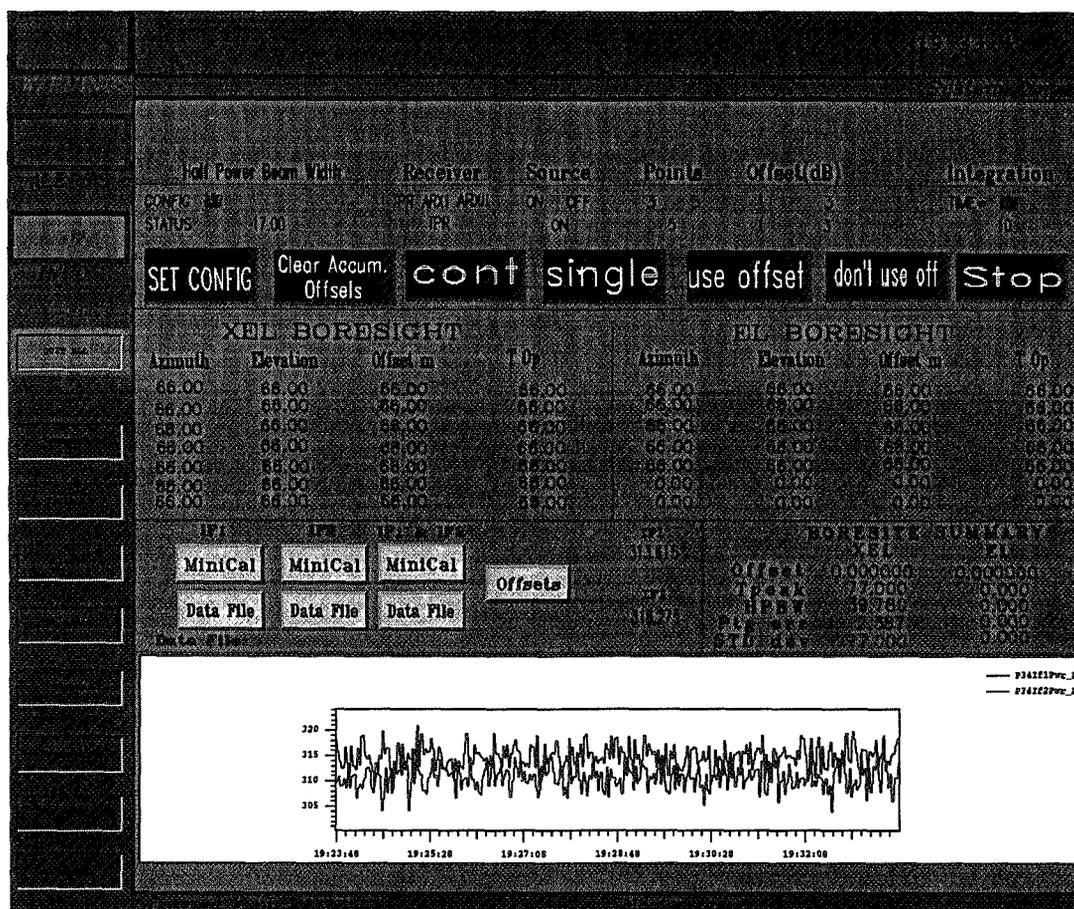


Fig. 15. The M&C subsystem user interface for the boresight procedure.

B. Link Monitor and Control Operator Assistant

There are several obvious drawbacks to operating the M&C subsystem manually. Certain DSN operations require continuous attendance by an operator over long periods of time, and some operations are highly repetitive and require large amounts of data entry. For instance, it is not unusual to conduct a KaAP track lasting 8 hours. During a KaAP track, the procedure called "load sidereal predicts" is repeated many times (see the KaAP TDN in Fig. 16 for an end-to-end view of the track); the "load sidereal predicts" procedure requires inputs by the operator each time it is conducted. We estimate that an 8-hour KaAP track requires about 900 operator inputs overall, if the track is performed manually. For this same track, under nominal conditions, LMCOA reduces the number of operator inputs to less than 10. Since people tend to be more error prone than computers on simple repetitive tasks, it makes sense to assign these tasks to LMCOA, freeing the operator for the task of handling exceptions, which requires more intelligence and knowledge.

The LMCOA performs the operations procedures for a tracking activity by executing a TDN, which is a procedure that is automatically generated by DPLAN, as described in the last section. DPLAN composes the TDN so that it contains the procedures (TDN blocks) needed for a specific tracking activity, and it orders them according to its knowledge of the dependencies that are defined among the blocks as well

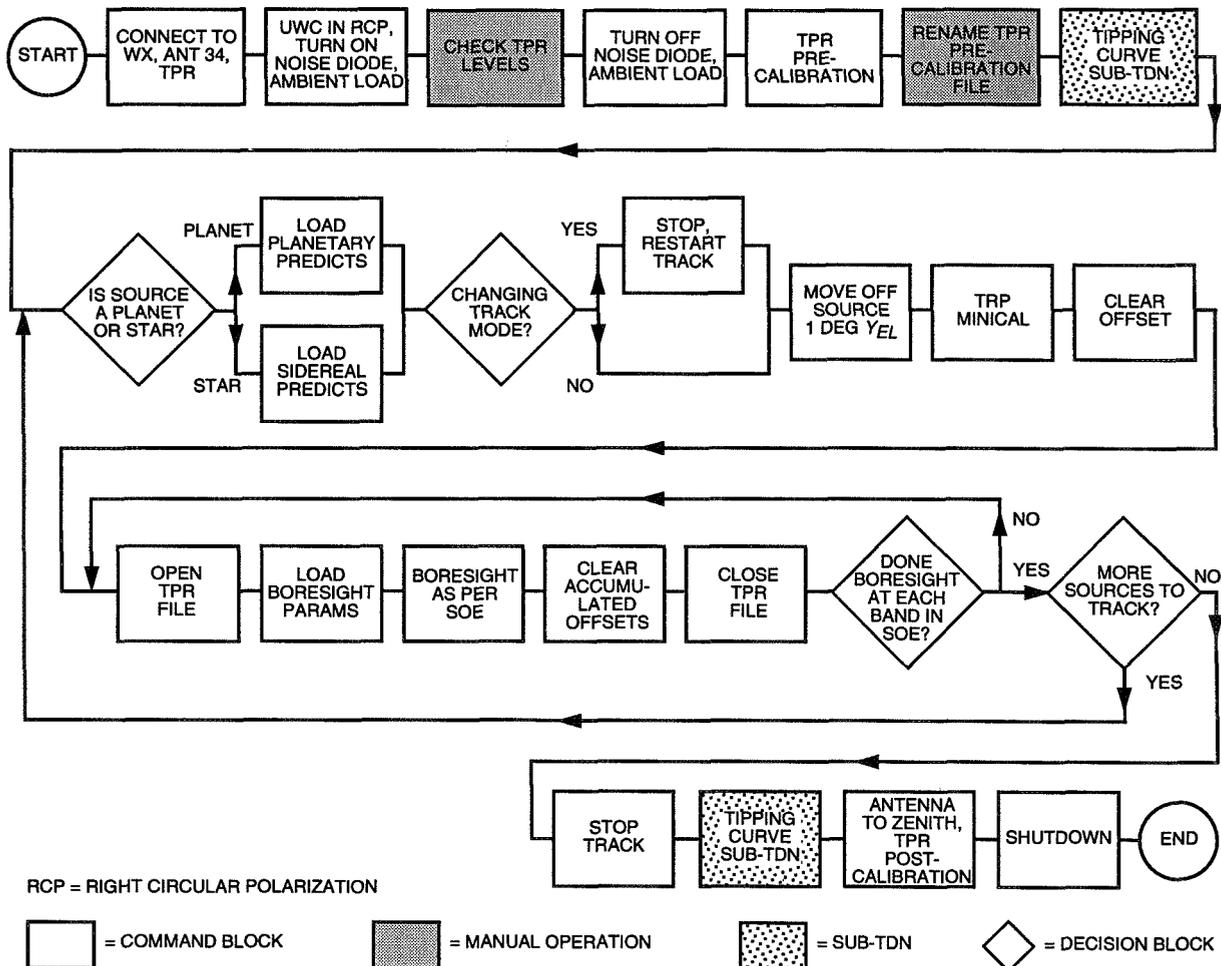


Fig. 16. A KaAP track TDN.

as by what it knows about the pre- and post-conditions of the blocks. The knowledge about interblock dependencies and about block pre- and post-conditions is passed to the LMCOA, whose task it is to execute the end-to-end procedure. The LMCOA receives the TDN in the form of a directed graph, where the precedence relations are specified by the nodes and arcs of the network. The blocks in the graph are partially ordered, meaning that some blocks may be executed in parallel. Temporal knowledge is also encoded in the TDN, which includes both absolute (e.g., “acquire the spacecraft at time 02:30:45”) and relative (e.g., “perform step y 5 min after step x ”) temporal constraints. Conditional branches in the network are performed only under certain conditions. Optional paths are those that are not essential to the operation but may, for example, provide a higher level of confidence in the data resulting from a plan if performed. More details about TDNs are provided in [3].

To execute a TDN, LMCOA performs the following functions (see Fig. 17):

- (1) Load the parameterized TDN into the execution manager
- (2) Determine which TDN blocks are eligible for execution and spawn a process for each TDN block
- (3) Check whether the preconditions of each TDN block have been satisfied
- (4) Once the preconditions are satisfied, issue the TDN block commands
- (5) Verify that the commands had their intended effects on the equipment

The operator interacts with the LMCOA by watching the execution of the TDN; the operator can pause or skip portions of the TDN, check the status of commands within individual blocks, and provide inputs where they are required. When part of a TDN fails, the LMCOA supports manual recovery by the operator by highlighting the point of failure and providing information about the preconditions or postconditions that failed to be satisfied.

C. Demonstration Results

Before LMCOA could perform a Voyager telemetry downlink activity, a number of steps had to be taken:

- (1) Knowledge was acquired about the operations procedures for performing this type of activity. This included collecting the commands for the activity; identifying each command’s parameters and its source in the SOE and project profile files; identifying each command’s preconditions and postconditions and tying them to specific equipment state variables; making the commands into TDN block procedures; defining the precedence relations among the blocks; and displaying the status of the TDN’s execution to the operator. Though much of this knowledge had been previously captured for the subsystems at DSS 13, some TDN blocks had to be modified because the Voyager telemetry downlink activity is different from the previously performed activities. In addition, as described earlier in this article, a new subsystem, TP-13, was installed at DSS 13 for the demonstration. Thus, a significant amount of knowledge about TP-13 had to be acquired.
- (2) Portions of LMCOA had to be modified to accommodate the knowledge acquired for the Voyager telemetry downlink activity. In particular, the user interface had to be written for the new TDN, and rules were written in the situation manager. One of the long-term

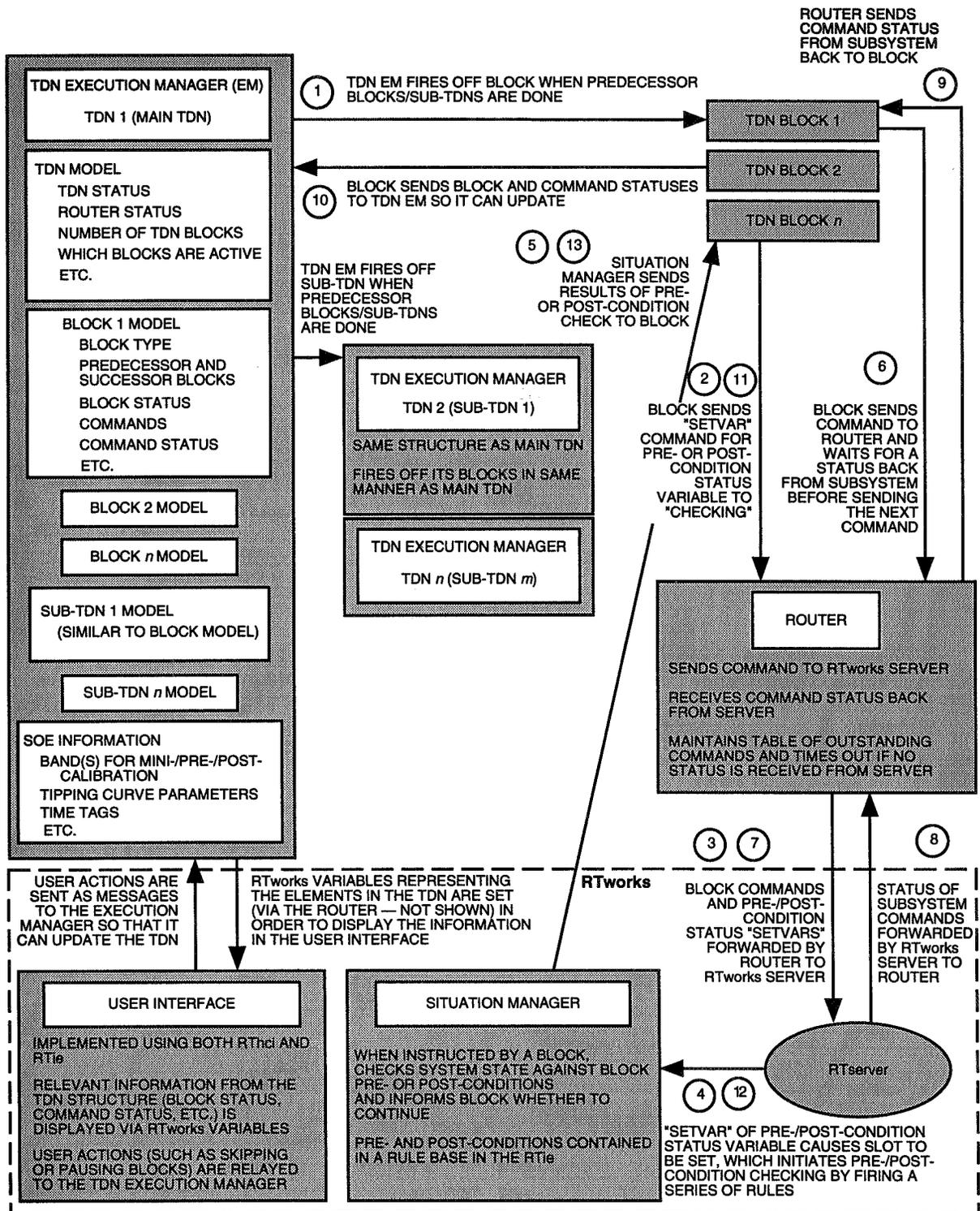


Fig. 17. LMCOA architecture.

goals for LMCOA is to generalize the architecture so that the TDN for any activity can be loaded and run without making modifications to any code, but for this demonstration, these modifications were necessary.

- (3) New MMS state variables had to be defined for the TP-13 subsystem so that it could communicate with the M&C subsystem and LMCOA. The MMS variable names are used by all of the subsystems on the M&C LAN, meaning that relatively minor modifications had to be made to TP-13, M&C, and LMCOA.
- (4) A TP-13 simulator was developed and installed in the network automation laboratory so that the communications interface could be tested between TP-13, M&C, and LMCOA. In addition, the TDN was debugged by executing it against the TP-13 simulator. The preconditions, postconditions, and time tags on TDN blocks often require a significant amount of testing before the TDN is validated. Such testing also helps identify missing knowledge.

Once the preceding activities had been completed, the modified versions of LMCOA and TP-13 were installed at DSS 13 and the demonstration was conducted over the course of several weeks.

VI. Knowledge Engineering

The SOE-driven automation demonstration uses a knowledge base to store data about the domain and, as such, requires that knowledge engineering be performed in order to acquire and encode the knowledge. In this section, we describe the level and types of effort required to acquire, encode, and validate the knowledge base used in this demonstration. The levels of effort can be summarized as follows and are in units of 8-hour workdays. It took about 36 workdays to acquire the knowledge, almost the same amount of time to encode the knowledge, but three times as much time to validate the knowledge base. Table 2 summarizes the knowledge engineering effort. Details about each phase of this effort are described below.

Table 2. Knowledge engineering effort.

| Effort | Time, days |
|------------------------------------|------------|
| Acquire the knowledge | 36 |
| Background information | 14 |
| Reuse blocks from other TDNs | 4 |
| Interview experts | 11 |
| Produce TDN documentation | 7 |
| Encode the knowledge | 45 |
| Project SOE parser | 10 |
| M&C subsystem extended | 11 |
| Adapt LMCOA to Voyager TDN | 15 |
| Write, revise TDN flat file | 4 |
| Decomposition rules, syntax editor | 5 |
| Validate the knowledge base | 127 |
| TDN, pre- and post-conditions | 51 |
| M&C subsystem extensions | 19 |
| LMCOA adapted for Voyager TDN | 24 |
| Project SOE parser | 7 |
| Goldstone testing | 24 |
| DPLAN | 2 |

A. Acquiring the Knowledge

The majority of knowledge represented in the system consists of the project SOE definition, the DPLAN knowledge base (used in generating the TDN), the TDN blocks themselves, the parameterization of the TDN blocks by both DPLAN and LMCOA, and the directives that make up the TDN blocks used in the demonstration. Much of this information comes from subsystem knowledge, for example, in defining the TDN blocks, identifying and understanding the subsystem directives within the blocks, and knowing how and when to parameterize subsystem directives within the blocks. Five subsystems were utilized for this TDN, two of which had never been used by LMCOA, and one that required more development and testing to incorporate it into the current TDN. Acquiring this knowledge took about 36 workdays.

The information was obtained by several methods, including reviewing documents and learning about the existing software systems, interviewing experts familiar with a particular part of the domain, documenting the knowledge, and participating in status meetings. Fourteen days were spent acquiring background information on DSN operations, SOEs, the M&C subsystem at DSS 13, and the LMCOA at DSS 13.

A small but very valuable amount of time was spent reviewing the existing TDNs for DSS 13. A significant amount of knowledge was directly extracted from both the Ka-Band Link Experiment (KaBLE) and KaAP TDNs for use in the Voyager TDN. The operational unit of a TDN, namely a block, has proven advantageous in our previous work on TDNs. The reuse of blocks between different operations procedures is one key advantage. Twelve of the 22 blocks, or 55 percent, in the Voyager TDN came directly from the KaBLE and KaAP TDNs. Our plans for the next generation LMCOA include the use of a relational database to store TDNs, blocks, and their contents. A block that can be used in more than one TDN needs to be represented only once in the database. Changes could be made to one or more of these reusable blocks without having to make significant, if any, changes to each TDN requiring the block. Therefore, to answer the question put forth in the overview and objectives section, the effort to extend this demonstration to a different type of track can be greatly reduced, depending on the reuse of TDN blocks from existing TDNs.

Most of the time, acquiring pre- and post-conditions occurred during the validation phase, due to the way the procedural knowledge is captured from operations personnel. Therefore, details about acquiring pre- and post-conditions are presented later in the validation section.

Eleven days were spent interviewing experts in the areas of DSN operations in general, DSS-13 operations for a spacecraft telemetry downlink pass, subsystem M&C, and project SOE definition. In addition, members of the Voyager project were interviewed in order to obtain parameters specific to Voyager that were required for parameterizing the TDN. Seven more days were spent generating graphical and written documentation of the TDN, and four days were spent participating in status meetings.

B. Encoding the Knowledge

The next step in the knowledge engineering process is to encode the knowledge into the right format in order for it to be machine readable. Due to the existing LMCOA, automated M&C subsystem, and planner, the majority of knowledge representations were designed and already used in previous efforts. In this effort, the majority of the encoding time was spent putting the acquired knowledge into a known format. Encoding the knowledge covers writing the TDN flat file, which specifies the Voyager TDN and is loaded by the LMCO; adapting the LMCOA to Voyager; writing the parser for the project SOE; extending the DSS-13 M&C subsystem to automate required subsystems; and developing DPLAN. Some highlights of the encoding effort are noted below.

The DSS-13 M&C subsystem was expanded to include two new subsystems that were not required in other TDNs developed for the LMCOA. This included developing simple subsystem simulators for testing purposes. The integration of TP-13 into the M&C environment took the most amount of time.

The changes made to the LMCOA for the new Voyager TDN took 15 days. Some of this time was spent extending the definition of the LSOE to account for new items in the Voyager SOE. As other types of tracks and more subsystems are handled by the LMCOA, the definition of the LSOE will broaden to include operations (e.g., boresight, modulation index changes) required by different tracks.

C. Validating the Knowledge Base

This phase of the knowledge engineering required about three times the level of effort of either the acquisition or encoding phases. This phase includes on-site testing at Goldstone in the operational environment; testing and developing simulators in the DSN advanced systems laboratory; and validating the TDN, pre- and post-conditions, and other software modules. The validation phase took 127 workdays.

Almost half of this time, 51 days, was spent validating the TDN and, especially, the pre- and post-conditions on TDN blocks. Validating the TDN involves making sure that the operations procedure is accurate. One reason this is a time-consuming task is that different experts have different ways of performing a part of the procedure or, over time, they revise and/or refine the operations procedure. This has been a constant in our experience developing TDNs. It is difficult to come to consensus on a single way to perform an operations procedure. After a TDN is in place and can be executed by the LMCOA, the operators can "see" the procedure more clearly and refine it. In any case, these reasons all point out the need to be able to modify TDN blocks and pre- and post-conditions. The format of the operations procedure must be simple so that it will be easy for developers and operations personnel to understand and maintain these procedures.

Pre- and post-conditions were extremely time consuming to validate. From our previous experience building TDNs, we have found that operations personnel do not usually think in terms of pre- and post-conditions. The need for pre- and post-conditions is much more obvious after an initial demonstration of the baseline TDN. At this time, the operators observe subsystem actions occurring automatically in the TDN and identify when some actions occur before sufficient subsystem states have been reached. These states are then implemented as preconditions. Another reason why operators are not familiar with the concept of preconditions is related to how they operate the manual M&C environment. In this environment, the operators often have a lot of equipment preconfigured. A detailed question and answer session between the TDN developer and the operator proved useful for identifying what portions of the preconfiguration are preconditions for existing blocks in the TDN. Postconditions were more difficult for the operations personnel and developers to determine.

The two main reasons for the large amount of time validating pre- and post-conditions are (1) the complexity of pre- and post-conditions and (2) subsystem support for them. Pre- and post-conditions can be very complex and, therefore, difficult to encode to have the desired effect. For example, a pre- or post-condition based on absolute or relative time can complicate the implementation of that condition. In an early version of the Voyager TDN, two TDN blocks occur in sequence, "start recording" and then "stop recording." The "start" recording block just tells the appropriate subsystem to start recording. The block then completes. The "stop recording" block does not execute until the appropriate time has been reached, according to the time in the SOE. Until this time, it appears that no blocks are executing in the LMCOA. That is true; however, the subsystems are recording data during this time. In order for the user interface to show that some activity was occurring, a postcondition was put on the "start recording" block. According to this postcondition, the "start recording" block will finish executing when the time has come to stop recording. Therefore, during the long data recording phase of the pass, the start recording block in the LMCOA remains in the executing state.

Subsystem support is also required in order to make use of pre- and post-conditions. In this and previous TDNs, one or more subsystems did not provide status information that needed to be checked by a pre- or post-condition to the M&C subsystem. For example, in the Voyager TDN, files were downloaded to the Station Data Recorder (SDR). We had to modify this subsystem to send a "finished downloading" status back to M&C so that a postcondition could automatically test when downloading was completed.

Extensions to the M&C subsystem took 19 days to complete. The majority of this time was spent incorporating TP-13, the receiver, into the M&C subsystem. New variables were added to the subsystem, and communications problems had to be overcome. This time also included writing subsystem simulators for testing M&C and the LMCOA in the DSN advanced systems laboratory.

A significant amount of time, 24 days, was spent at DSS 13. Extensive testing and debugging that could not be performed in the DSN advanced systems laboratory environment were performed at Goldstone. Much of this time was spent incorporating TP-13 into the DSS-13 M&C environment. Some of the difficulties encountered resulted in modifying sections of the knowledge base related to the M&C subsystem as well as the LMCOA.

Finally, here are the levels of effort for debugging other software modules: Debugging the SOE parser took 7 days, and debugging DPLAN took 2 days. Another 24 days were required to debug the Voyager version of the LMCOA.

VII. Discussion and Summary

At the request of the TMOD Services Fulfillment Reengineering Team, a team from the network automation work area demonstrated SOE-driven automation for a Voyager telemetry downlink track at DSS 13 in February 1995. This SOE-driven automation demonstration accomplished the following significant results:

- (1) Demonstrated the ability to automatically generate procedures (TDNs) based on a project sequence of events (PSOE) using an artificial intelligence planning system called DPLAN
- (2) Demonstrated the capability for automated procedure (TDN) execution using the LMCOA system; this capability has been demonstrated before but never on a spacecraft-related track
- (3) Identified a set of gaps in the SOE definitions and project profiles that make it difficult to automatically generate operations procedures
- (4) Produced detailed measurements of the amount of effort required for knowledge engineering for necessary DPLAN and LMCOA knowledge bases
- (5) Showed that validating a TDN requires a lot of time and effort before it can be released for operational use
- (6) Pointed out the need for subsystems to report their status so that postcondition checking can be done more explicitly
- (7) Demonstrated that reusing TDNs can save a significant amount of effort in knowledge acquisition, knowledge engineering, and validation

Acknowledgments

We wish to thank Les Deutsch and the TMOD Services Fulfillment Reengineering Team (RET) for challenging us to demonstrate the technology described in this

article; the SOE-driven automation demonstration on the Voyager track was performed at their request and funding for portions of this work was provided from the RET account. We also acknowledge Chad Edwards and the DSN Technology Program for the ongoing support of these efforts. Finally, we wish to give credit to Lynne Cooper, who was the Research and Technology Operating Plan (RTOP) manager who first envisioned and demonstrated the TDN and LMCOA concept and who was a member of the TMOD RET.

References

- [1] S. Chien, "Automated Synthesis of Complex Image Processing Procedures for a Large-Scale Image Database," *Proceedings of the First IEEE International Conference on Image Processing*, vol. 3, Austin, Texas, pp. 796–800, November 1994.
- [2] K. Erol, J. Hendler, and D. Nau, "UMCP: A Sound and Complete Procedure for Hierarchical Task Network Planning," *Proceedings of the Second International Conference on AI Planning Systems*, Chicago, Illinois, pp. 249–254, June 1994.
- [3] K. E. Fayyad and L. P. Cooper, "Representing Operations Procedures Using Temporal Dependency Networks," *SpaceOPS '92*, Pasadena, California, pp. 541–546, November 1992.
- [4] K. Fayyad, R. Hill, and E. J. Wyatt, "Knowledge Engineering for Temporal Dependency Networks as Operations Procedures," *Proceedings of AIAA Computing in Aerospace 9 Conference*, San Diego, California, pp. 1040–1047, October 19–21, 1993.
- [5] W. R. Heuser and L. Cooper, "An OSI Architecture for the DSN," *SpaceOps '92*, Pasadena, California, pp. 97–101, November 1992.
- [6] R. W. Hill, Jr., K. Fayyad, T. Santos, and K. Sturdevant, "Knowledge Acquisition and Reactive Planning for the Deep Space Network," *Proceedings of the 1994 AAAI Fall Symposium on Planning and Learning: On to Real Applications*, New Orleans, Louisiana, pp. 2–4, 1994.
- [7] R. W. Hill, Jr., S. Chien, C. Smyth, and K. Fayyad, "Planning for Deep Space Network Operations," *Proceedings of the 1995 AAAI Spring Symposium on Integrated Planning*, Palo Alto, California, pp. 51–56, 1995.
- [8] R. Hill and W. L. Johnson, "Situated Plan Attribution for Intelligent Tutoring," *Proceedings of the Twelfth National Conference on Artificial Intelligence*, Seattle, Washington, pp. 499–505, July 31–August 4, 1994.
- [9] A. Lansky, *Localized Planning With Diverse Plan Construction Methods*, Technical Report FIA-93-17, Ames Research Center, Moffett Field, California, June 1993.
- [10] A. Munro, M. C. Johnson, D. S. Surmon, and J. L. Wogulis, "Attribute-Centered Simulation Authoring for Instruction," *Proceedings of AI-ED 93, World Conference on Artificial Intelligence in Education*, Edinburgh, Scotland, pp. 23–27, August 23–27, 1993.
- [11] J. S. Pemberthy and D. S. Weld, "UCPOP: A Sound Complete, Partial Order Planner for ADL," *Proceedings of the Third International Conference on Knowledge Representation and Reasoning*, Boston, Massachusetts, pp. 103–114, October 1992.