KNOWLEDGE ENGINEERING FOR TEMPORAL DEPENDENCY NETWORKS AS OPERATIONS PROCEDURES

Kristina E. Fayyad
Randall W. Hill, Jr.
E. Jay Wyatt

Jet Propulsion laboratory
California Institute of Technology
4800 oak Grove Drive
Pasadena, CA 91109

Abstract

'1 'he usefulness of a knowledge based system is highly dependent upon the implementation of the knowledge base which drives that system. The knowledge acquisition and engineering process is a recognized bottleneck in the development and deployment of knowledge based systems. This paper presents a case stud y oft he kno wledge engineering process employed to support the Link Monitor & Control Operator Assistant (LMCOA). The LMCOA is a prototype system which automates the configuration, calibration, test, and operation (referred to as *precalibration*) of the communications, data processing, metric data, antenna, and other equipment used to support space-ground communicant ions with deep space spacecraft in NASA's Deep Space Network (DSN). The primary knowledge base in the LMCOA is the Temporal Dependency Network (TDN) - a directed graph which provides a procedural representation of the precalibration operation. The TDN incorporates precedence, temporal, and state constraints and uses several supporting knowledge bases and databases. The paper provides a brief background on the DSN, and describes the evolution of the TDN and supporting knowledge bases, the process used for knowledge engineering, and an analysis of the successes -- and problems -- of the knowledge engineering effort.

Introduction

DSN Link Monitor & Control (LMC) operations consist primarily of executing procedures to configure, calibrate, test, and operate a communications link between an interplanetary spacecraft and the ground station. Currently, LMC operators are responsible for integrating procedures into an end-to-end series of steps. In this article, we address the use of a Temporal Dependency Network (TDN) for specifying operations procedures. The TDN incorporates the insight of operations, engineering, and science personnel to improve mission operations. An operational test of this concept as implemented in the LMC Operator Assistant (1. MCOA) prototype was performed in early 1993.[5] The domain selected is for Galileo Very Long Baseline Interferometry (VLBI) Delta Differential One-way Ranging (DDOR) precalibration using the 70m antenna at the Goldstone Deep Space Communications Complex (GDSCC) in California. '1-he extensibility of this representation to other domains is now being analyzed. This paper analyzes the knowledge engineering effort required during the development of the LMCOA prototype. in the first section, knowledge engineering is defined followed by a discussion of the initial, high level domain analysis. The next section presents the low level domain analysis as wc]] as the definition of a TDN and the information required to build it. Finally, results arc presented as well as recommended tools to facilitate building TDNs.

Knowledge Engineering

As shown in Figure 1 6 our approach to knowledge engineering is an interative process which contains the following steps. The development of the TDN representation followed these steps: acquire the know] edge, organize it, analyze and develop conceptual models, review models with experts, implement the model and knowledge base, and review performance. There were two major iterations through the process. The first corresponds to a domain analysis, where the knowledge engineering was at a high level and resulted in the design of the TDN. The second pass was used to refine the TDN representation, add additional knowledge structures needed to support real-time operations, and build the knowledge base used to support the VLBI DDOR TDN.

DSN operations personnel identified LMC operations, and specifically precalibration as a major problem. The intent of the initial domain analysis was to characterize this problem and identify ways of improving operations.

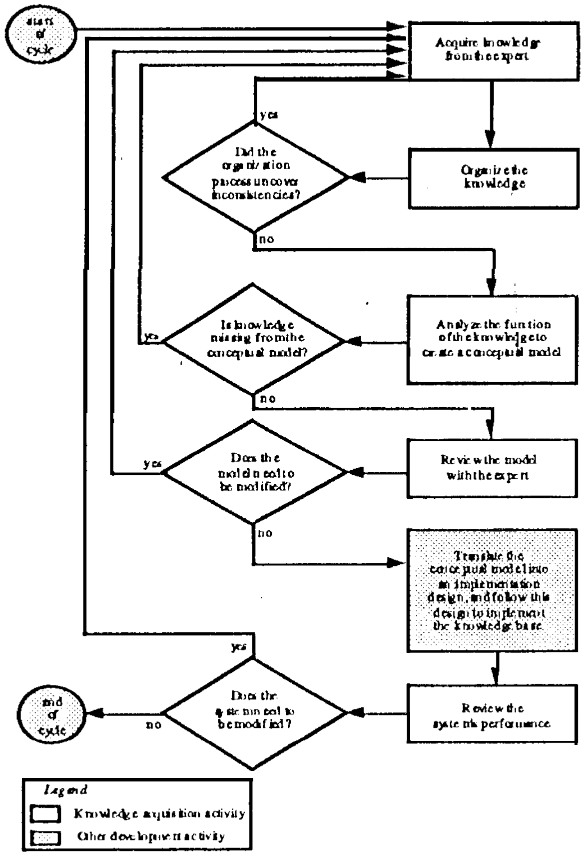

Figure 1. Detailed Investigation: knowledge acquisition and system development.[6]

High level domain analysis

The first step we took was to become familiar with how an operator performs precalibration. Precalibration is the task of creating a communications link between a DSN antenna and a spacecraft. During precalibration, the operator must type in well over 100 directives to configure, test and calibrate the subsystems and must monitor over 1000 responses (in the form of textual messages on a scrolling log) to determine the execution status of those directives.

A directive is the basic unit of control that the operator has and is the primary interface to the subsystems. The operators need to know thousands of directives from which they select the ones appropriate to perform the precalibration. The operators are responsible for determining the appropriate sequence of directives, inserting the correct parameters, and determining, with limited support from the system the state of the equipment following execution of those directives.

The knowledge acquisition effort had to address two specific issues, the first being what were the appropriate directives, their parameters, execution responses and other monitor and control information associated with a directive. The second issue was in what order did these directives need to be executed in order to successfully perform precalibration. To acquire this information, a variety of knowledge engineering techniques were used which included reviewing operations logs, interviewing operations and engineering personnel, and reviewing documentation. As a result of these activities several specific operability problems were identified with precalibration.

Currently, operators must manually enter hundreds of directives through a command-line interface. Parameters to these directives are not available on-line but rather in hard copy form which the operator must sift through. After sending a directive, the operator must check a scrolling log to see that it was executed, due to lack of closed loop control. At the same time they must also monitor the health and status of subsystems by analyzing the data in scrolling logs and several graphical displays. If operators multiplex their time to work on several subsystems at the same time, they must correlate a variety of monitor data and event messages to the specific subsystems without system support. Therefore there is a lack of assistance for supporting parallel operations. An added difficulty is that the operator must filter through alarm messages, many of which are false alarms, in order to identify significant ones. Finally, the entire representation of the operations procedure is documented in several manuals which address a specific subsystem or spacecraft, or provide a general overview of an activity, The operators do not have a procedure to follow from start to end. They must refer to the documentation and rely on their own experience.[5]

End-to-end Procedure Representation

The combination of these problems results in a system which requires large amounts of time for subsystem configuration and calibration. It is also susceptible to keyboard entry errors, and places a huge burden on the human operators to correlate and process a large amount of data. Based on this analysis, it was decided to pursue an Artificial Intelligence-based approach to improving DSN link operations. Addressing the above mentioned problems served as the initial requirements for the LMCOA. Onc of the main requirements was development of an end-to-end representation of a precalibrat ion procedure.

Analysis of operator logs showed that some directives could be issued in different orders. Interviews with operations personnel identified that there were independent sequences of directives. These two facts together implied that there was inherent parallelism in precal i brat ion. The concept of a TDN was developed which could represent a procedure with parallel activities. We define what a TDN is and how wc usc it later in this paper.

Perspectives on Precalibration

Discussions with operations, engineering, and scientific personnel lead to an understanding of the difficulties of precalibration and in particular, there was an interest in improving precalibrat ion for a V 1.111 1)DOR pass on the 70m antenna. The perspectives of several individuals regarding the details of precalibration led to the comprehensive '1'1)N representation for an end-to-end precalibration procedure.

Operations personnel arc responsible for the real-tinw activity of precalibration. They know what equipment is necessary to setup the link, how much time is required and available, and the procedure to configure the equipment in the link. The operators arc responsible for performing certain activities at particular times, based on scheduling information available to them. They also know what really works, and therefore, they maintain their own notes for how to configure the equipment. Since time is a major constraint, they also know the options available. During link monitor and control activities, they must consider safety, time, and data integrity when making decisions.

Subsystem and opera t ions engineers arc primarily interested in the equipment required for the link. Based on their intimate knowledge of the equipment, they prescribe recommended sequences of directives for configuring the equipment. These recommendat ions are bawd on equipment physical characteristics and limitations. Therefore, the engineers provided the reasons for sequence of directives, dependencies between subsystems, as wc]] as verification of the inherent parallelism in some precalibration activities.

The scientists' interest in precalibration is to ensure the best possible data resulting from a pass with a spacecraft. They prescribe what tests to perform during precalibration to ensure high quality data, Given the constraints of real-time operations, they also recommend the most important aspects of precalibration and which arc optional.

The TDN was designed to incorporate the various precedent, temporal, and state constraints required to represent these various perspectives on the precalibration procedure. The TDN is described in more detail in the next section. The result of this initial phase of domain analysis was a high level TDN for VLBI DDOR precalibration, shown in Figure 2.

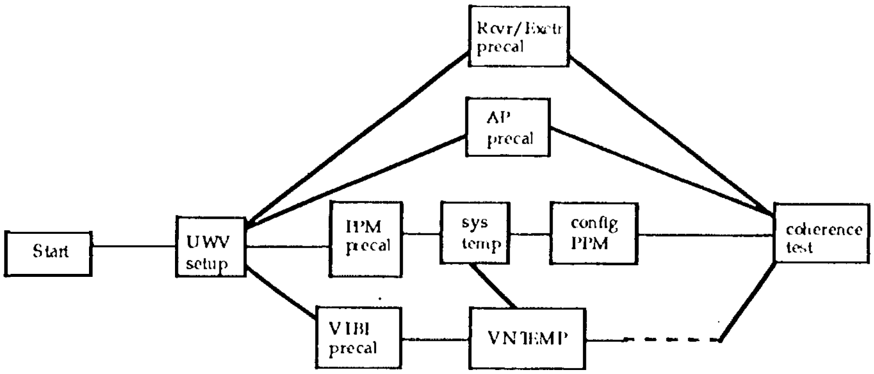

Figure 2. High level TDN.

Detailed domain analysis

In order to fully test the concept of the TDN in the LMCOA prototype and to provide semi-automated precalibration and closed loop control, it was necessary to enhance and refine the knowledge in the TDN. To provide closed loop control a rule-based module, the Situation Manager[4] (SM) was developed. The SM evaluated incoming data from subsystems and checked preconditions and postconditions in the TDN, to verify and control the execution of the TDN. In order to support closed loop control, the support data mentioned above was expanded to a separate knowledge base, the Directive Diet ionary. On top of this knowledge base, the low level TDN was built. In order to understand the TDN representation it is necessary to first understand the basic building Mock of the TDN, directives.

Directives

The primary data unit of the TDN is a directive. A directive is a control message which is sent to an individual subsystem in order to perform a specific function. The primary data fields arc the destination subsystem, the control action, and any associated parameters. To support the TDN model of procedures, an enhanced representation of a directive is used. An example of a directive definition is presented in Table 1. Each directive is represented in the Directive Dictionary as an object containing the following information.

1. Directive syntax (subsystem, message name, required and optional parameters).
2. The function of the directive: what primary and side effects it has on the subsystem; what changes it causes in any devices or subassemblies.
3. Parameter definitions: any constraints on the parameters and the support data used to determine parameter values.
4. Directive responses: the response messages sent from the subsystem to the LMC to acknowledge receipt of the directive. This is only a communications handshake and does not indicate that the directive was successfully executed.
5. Rejection notices: messages sent by the subsystem when the directive has failed to execute. (Includes syntax errors as well as real-time failures).
6. Monitor and event information: data that may be generated by the subsystem based on the actions of the directive. Specifics which parameters and user interface displays to monitor to confirm that the directive has successfully executed.
7. Preconditions: what state must the system be in before this directive can be sent.
8. Postconditions: what state the system is in when the directive has successfully executed.

Actions	Transmits predict data set CW to the ACS
Sources	I 1 og DOY-067-1991, line 189
Preconditions	Predicts available
Postconditions	Predict table is filled Predicts downloaded successfully
Responses	COMPLETED. PROCESSING DLOAD REQUEST
Rejections	COMPLETED. INVALID PREDICT SET NAME
Event notice messages	PA 14:INTERPOLATING CW SUB1 ... PA 14:ACS CONFIRM DLOAD... PA 14:ACS <time>

Table 1. Directive Example, AP DLOAD PRED CW

The information in the directive definitions is stored in a knowledge base, the Directive Dictionary. Of the above listed types of information, only a subset, dealing primarily with syntax and general responses, is available in the DSN documentation. Much of the information, such as extensive preconditions and postconditions is available only from operations personnel and engineers.

Temporal Dependency Network

A TDN is a complex object which encodes the procedural information necessary to perform a specific operational task. The primary representation of the TDN is an augmented directed graph. In the graph, each arc represents a strict precedence relationship, each node a sequence of directives which perform a subset of the overall function. The network explicitly specifies the precedence relationships between nodes, any potential parallelism, and rules for recovering from global faults. The nodes, or blocks, consist of the directives, temporal constraints, preconditions and postconditions, and local recovery information should the Mock fail. An example block is given in Figure 3.

After identifying the directives necessary to perform the given operation and any preconditions and postconditions specific to the type of pass, designing a TDN becomes an exercise in assigning directives to blocks. The TDN is the genera] representation of an operational task. An instance of the TDN is created from the general representation and is parameterized for the specific pass being performed. From this perspective, the TDN acts as a template for operations, and individual parameters (time, frequency, file names) arc filled in at execution time to perform operations.

Preconditions:
ACS finished resetting

Directive sequence:
AP CONN 14
AP ACS DELUT 299
AP ACS REQCORR

Postconditions:
ACS received connect"""
DELUT set to 299

Figure 3. Block Example

Detailed TDN

After obtaining a high level version of the procedure, the next step is to define the procedure at a lower level. i.e. down to the level of each operator directive issued and other actions performed. As a result of reviewing the high level TDN for VLBI DDOR, an expert provided detailed flow charts of the precalibration procedure including parallelism of subprocedures. This formed the basis of the low level TDN. Subsequently, more details were obtained about the configuration procedure of each subsystem and were reviewed by other experts. These knowledge engineering tasks required on the different subsystems can. be done in parallel while keeping track of dependencies to and from the other subsystem configuration procedures.

At a later time these procedures will be merged into a single TDN. e.g. While reviewing the TDN with operations personnel, it is important to find out dependencies between subsystems. For example, in Figure 2 "V NTEMP x y" is issued after getting the system temperature and threshold which is done during Precision Power Monitor (PPM) setup. Dependencies are not always explicitly identified when the procedure is sequential. However, explicit representations are needed for parallel representation of the procedure.

The information that needs to be obtained for the low level TDN consists of directives issued, directive preconditions and postconditions, displays accessed, manual operat ions performed such as making safet y pages and manual configurations and the order of all of these actions. This information can most easily be obtained by interviewing operations personnel and by referring to operations logs. SOme documentation exists which has this low level procedural information, but not necessarily all the details that are required to successfully perform a precalibration.

Data for Closed Loop Control

One of the features of the LMCOA prototype is to provide closed loop control of TDN execution. That is, it provides the operator with explicit and consistent feedback about the executing state of directives.[5] In order to provide this capability it 'is necessary to determine the responses, event notice messages, monitor data, and preconditions and postconditions associated with each directive. The details of acquiring this information and the usc of it for closed loop control is discussed below.

Subsystem Data

For each directive sent out, a directive response is returned which is an acknowledgement from the subsystem that the directive was received. Event notice messages and monitor data provide additional information on the status of the subsystem in response to a directive. The responses and event notice messages can be initially determined from analyzing operations logs and referencing Software Operators Manuals (SOMs). However, it is necessary to find out from operations personnel and engineers the usc and meaning of these messages. in some cases, the receipt of a directive response is the signal that the next directive can be sent. in other cases, the operator waits for one or more event notice messages in response to a directive in order to determine successful completion of a directive and to proceed with sending more directives.

Identifying the responses and event notice messages for each directive was not always straightforward, For example, a few very long event notice messages were truncated in both the scrolling and printed logs. To provide closed loop control in the LMCOA prototype, knowledge of the entire message is required. in such cases, the LMCOA itself assisted in some of the knowledge engineering. It was used to trap the messages directly from the LAN. These complete messages were then inserted into the knowledge base.

The SOMS provided varying degrees of information on directive responses and event notice messages depending cm the particular subsystem. In some documentation, the directive responses are completely specified with the directi ve. In other cases, only the type of response is documented, e.g. PROCESSI NG, COMPLETED while there may be more text in the actual response. Also, the documentation for different subsystems varies in how the event notice messages are presented. in some cases, they arc organized by type of action that the message is associated with. For example, the Antenna Pointing Assembly (APA) SOM organizes them according to predict, conscan, monitor, etc. in other SOMS the event notice messages arc listed alphabetically. Finally, some SOMS provide explanations of the messages and others do not. interviews with experts were needed to fill in information gaps.

The existing prototype made limited use of monitor data. Onc problem was that in the existing environment there was incomplete access to monitor data. in addition, the usc of monitor data was integrated into the prototype after major knowledge bases had already been built. However, in extensions of this prototype, an emphasis will be placed on creating the knowledge bases according to monitor data.

Preconditions and postconditions

Preconditions specify device states that must be true before the directive can execute. Postconditions specify the expected device states after the directive has successfull y executed. Precedence relationships in the TDN arc formed by ensuring that the actions required to satisfy a directive precondition occur and arc verified before that directive executes. So, if two directives arc in sequence because one depends on the successful completion of the other, these directives will be placed in separate blocks and a precedence relationship formed between them.

Directive preconditions are pushed up to the block level, so that before the block begins executing its first directive, all preconditions of all directives in that block must be satisfied. in some cases, this check is redundant because completion of the previous block is dependent upon satisfying a postcondition which satisfies the precondition of the next block. We have designed the TDN in this wa y for two reasons:

1) If a directive or block is moved to a different location in the TDN, violated precedence relations will be detected.
2) if a device fails between the end of the first block and the start of the second, wc have a way to detect the failure before proceeding.

The preconditions and postcond it ions of directives were determined by reviewing subsystem documentation and by interviewing operations personnel and engineers. By analyzing rejection messages in logs and subsystem documentation, preconditions, may become apparent. For example, the following is a rejection message associated with Al' ACS IDLE (i.e. put the ACS into IDLE state.):

REJECTED. CANNOT CHANGE MODE WITH SCAN ON

Therefore, a precondition is that SCAN is OFF.

Results and Tools

The block level TDN for VLBI DDOR precalibration was the result of assigning direct ives to blocks and incorporating the above subsystem data, preconditions., and postconditions. The LMCOA was successfully demonstrated by performing precalibration for a VLBI DDOR pass at the Goldstone 70 Meter Antenna in Goldstone, California.

Usc of the TDN itself during the knowledge engineering effort proved to be a valuable tool to communicate with experts about procedural details. Analysis of the knowledge engineering described above led to the need for other tools to facilitate this type of effort in the future.

o Access to online documentation including SOMS and monitor data documents to facilitate the construction of the knowledge bases.

o A graphical too] to facilitate building, modifying, and maintaining TDNs.

o A tool to build the Directive Dictionary, which includes links to critical documentation and software and automatic notification when ncw versions of such information is available.

o A method of arbitrating and documenting the different views and preferences of experts regarding the procedure.

Conclusion

Previous research involved investigate ing the usc of a Temporal Dependency Network (TDN) as a way of specifying LMC operations procedures that incorporate the insight of operations, engineering, and science personnel to improve mission operations. An operational test of this concept as implemented in the LMC Operator Assistant (LMCOA) was performed in early 1993. The application domain was Galileo VLBI DDOR precalibration on the 70m antenna at the GDSCC in Goldstone, California. The extensibility of this representation to other domains is now being analyzed. This paper analyzed the knowledge engineering effort required to build a TDN and recommend cd improvements to the LMCOA to facilitate the knowledge engineering process.

Acknowledgements

The work described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. The other members of the LMCOA team arc Lorrine Lee, Kathy Sturdevant, Crista Smyth, Juan Urista, and Sanguan Chow. We would like to acknowledge the contributions of Pam Wolken, Terry Dow, Dave Girdner, Dan Kicwicz and the many operations and engineering personnel at]1'1. and Goldstone Deep Space Communications complex.

References

1. Cooper, Lynne P, "operations Automation Using Temporal Dependency Networks," Technology 2001, San Jose, CA, December 3-5, 1991.

2. Cooper, Lynne P., Rajiv Desai and Elmain Martinez, "Operator Assistant to Support Deep Space Network Link Monitor and Control," SOAR Symposium, Houston, TX, 1991.

3. Fayyad, Kristina E. and Lynne P. Cooper, "Representing Operations Procedures Using Temporal Dependency Networks," Space Ops '92, Pasadena, CA, November 1992

4. Hill, Randall W., Jr. and Lorrine F. Lee, "situation Management in the Link Monitor and Control Operator Assistant (LMCOA)", SpaceOPS 92: Second international Symposium on Ground Data Systems for Space Mission Operations, Pasadena, CA, 1992,

5. Lee, Lorrine F. and Randall W. Hill, Jr., "Process Control and Recovery in the Link Monitor and Control Operator Assistant," Space Operations, Applications and Research Symposium, Johnson Space Center, Houston, TX, August, 1992.

6. Scott, A. Carlisle, Jan E. Clayton, and Elizabeth L. Gibson, A Practical Guide to Knowledge Acquisition, Addison-Wesley, 1991.

7. Urista, Juan. "Using MACH Threads to Control DSN Operational Sequences," Space Ops '92., Pasadena, CA, November 1992.