Automated Consultation for the Diagnosis of Interplanetary Telecommunications

A.G. Quan, U.M. Schwuttke, J.S. Herstein, and D.J. Atkinson

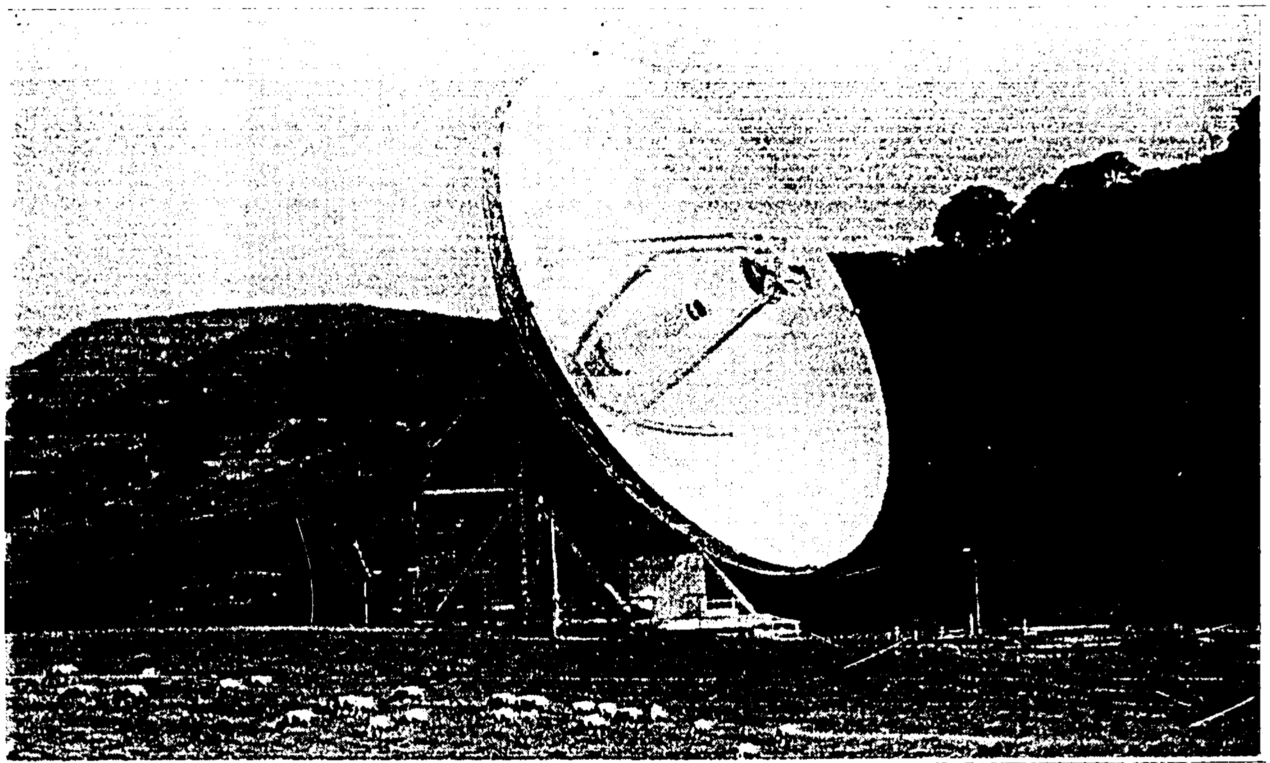
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109
alan@puente.jpl.nasa.gov

submitted to the AAAI Conference on Innovative Applications of Artificial Intelligence (IAAI-94)

INTRODUCTION

SHARP (Spacecraft Health Automated Reasoning Program) is a knowledge-based consultant for diagnosing problems in NASA's Deep Space Network (DSN) telecommunications link in near real-time. It has been under development for seven years, gradually transitioning from a research prototype to an operational system.This paper describes most recent version, which is an operational system used by spacecraft mission controllers; previous versions are described elsewhere [Atkinson 1992, Martin 1990]. The transition of this system to an operational environment involved the design and implementation of a generic architecture and a some application specific customization for its first intended long-term end-user, NASA's Mars Observer mission. Subsequent loss of the Mars Observer spacecraft provided an opportunity to validate claims for the generality of this tool, with an immediate port to the Galileo mission, (launched in 1989 and scheduled to enter Jupiter orbit in 1995). This port was achieved with less than ten percent of the effort required for the original Mars Observer implementation, as only mission specific components of the system were affected.

SHARP has been developed for use by the multimission control team at the Jet Propulsion Laboratory (JPL), for detection and analysis of spacecraft and ground system problems. The telecommunication (Telecom) downlink provides a means of communication between the spacecraft and ground, and includes the Telecom subsystem on the spacecraft and components of the DSN. Telemetry data arrives at a DSN station such as the one shown in Figure 1. Stations are located in the U.S, Australia, and Spain. Data from these stations is sent to JPL for monitoring and analysis. There are two kinds of spacecraft telemetry data: the science data, which contains actual data obtained from science instruments on the spacecraft, and the engineering data which provides the health and status of the spacecraft subsystems. There is also a third kind of telemetry



¹igure 1. A 34-meter high-efficiency antenna at the DSN Station in Canberra, Australia.

known as monitor data, which provides information about the DSN. SHARP concerns itself only with the latter two types of data.

SHARP assists the mission controller responsible for quick-look Telecom analysis in routine operations by automatically detecting degradation in the Telecom link, notifying the mission controller, isolating the cause through interactive consultation, and recommending corrective action. In past interplanetary missions, these capabilities have been provided by a full-time human analyst with expertise in spacecraft telecommunications. More recent missions have not been able to fund full-time telecommunications support. As a result, there has been an increased need for automated and intelligent tools that can substitute for a human specialist by providing interactive consultation to augment the general knowledge of the mission controllers. SHARP answers this need by implementing domain expertise and embedding artificial intelligence techniques for diagnosing anomalies in a near-real-time monitoring system.

THE APPLICATION DOMAIN: MISSION CONTROL AND TELECOM LINK ANALYSIS

The responsibility of the Mission Control Team is to coordinate and control spacecraft flight operations, transmitting commands to the spacecraft and performing real-time monitoring of spacecraft performance, the telecommunications link, and the Ground Data System. The Mission Control Team is also responsible for correcting spacecraft problems by issuing commands to the spacecraft. Telecommunication anomalies occur frequently, interrupting the transmission of data from the spacecraft, Detection and correction of the anomalies requires coordinated monitoring and diagnosis of both the spacecraft and Deep Space Network telecommunications systems. Without SHARP, this activity would still be performed manually, requiring a mission control team member to first detect that a problem exists by monitoring key telemetry values and noticing when they stray outside predetermined tolerances. Once the telemetry symptoms of the problem are discovered, a Telecom domain expert is consulted. The expert attempts to diagnose the cause of the anomaly based upon past experience, detailed knowledge of the telecommunications system, and examination of other related telemetry data as well as non-telemetric data such as the schedule of spacecraft events. After the cause of the anomaly is deduced, the expert must then decide what actions to take in order to correct it, if possible. This process can be rather time consuming, depending on the nature of the anomaly, the availability of the Telecom expert, and whether the initial corrective action chosen by the expert is effective.

Dependence upon a domain expert for anomaly diagnosis and recovery has two major drawbacks. First, this requires that an expert be on call 24 hours a day, since anomalies can occur at any time. The timeliness of an expert's response to a problem can be critical for recovering important data, or in the extreme case, saving a spacecraft. Secondly, when the experts retire, their skills are lost. The accumulated expertise of mission operations personnel is an important resource which should be preserved rather than recreated every time a senior engineer leaves the flight project. SHARP is designed to serve as an automated Telecom consultant for mission controllers, offering the problem-solving ability of a human expert for well-understood telecommunications anomalies. In this capacity, SHARP allows fast response time in anomaly diagnosis and correction, and preservation of mission expertise for the entire duration of a mission.

DESIGN AND IMPLEMENTATION

SHARP implements Telecom domain expertise and applies artificial intelligence (Al) techniques for diagnosing anomalies in the Telecom downlink. This is accomplished by examining relevant telemetry data from both the DSN and the spacecraft. If an anomaly is detected, SHARP uses knowledge-based analysis to diagnose the problem and provide a recommendation for corrective action. The SHARP system is composed of three primary modules, as shown in Figure 2: a telemetry communication interface to read telemetry from the ground data subsystem, a rule-based expert system to diagnose anomalies, and a graphical user interface (GUI) to display telemetry alarms and diagnostic messages.

THE TELEMETRY COMMUNICATION INTERFACE

The telemetry communication interface reads telemetry from the ground data system (GDS) and passes it to the GUI and knowledge base. It is composed of a SHARP-to-GDS telemetry server, an expert system message interface layer, and a GUI

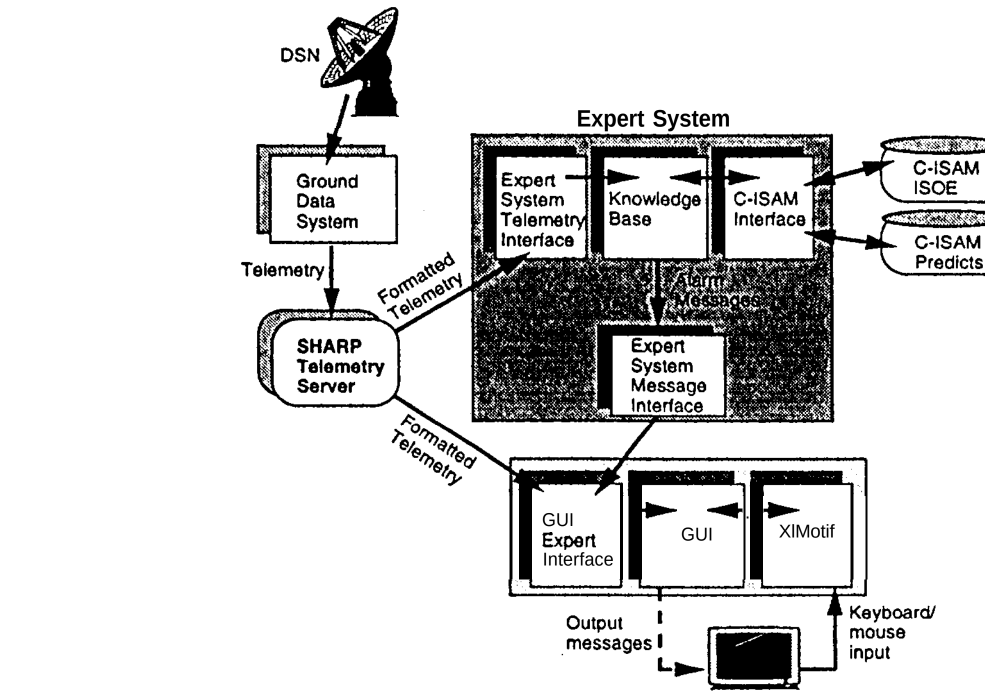


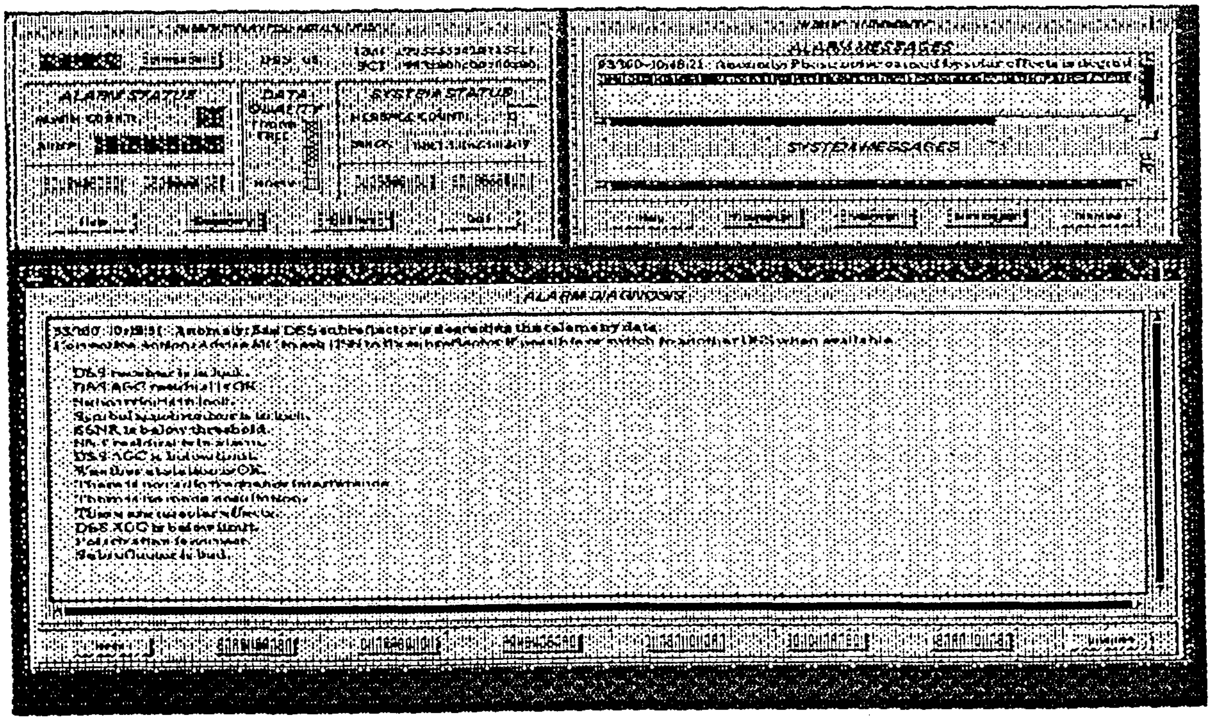
Figure 2. The high-levelarchitecture of the SHARP system.

message interface layer. The telemetry server obtains raw telemetry data in ASCII from the GDS via a UNIX pipe and converts it to a format for use by the knowledge base and GUI. The data retrieved from the GDS is in the form of text lines which are written to either a first-in-first-out queue (FIFO) or a disk file. The formatted lines can then be distributed to a maximum of five clients. The telemetry interface accepts UNIX socket connections from the expert system and the user interface ailowing interprocess communication. When the knowledge base requests data, the expert system message interface layer reads the formatted telemetry data from the server and passes it to the expert system. The formatted telemetry can also be sent directly to the GUI through a connection to the SHARP-to-GDS telemetry server.

THE GRAPHICAL USER INTERFACE

The GUI provides the interface between SHARP and the end-user, and has been designed according to the needs to the mission control end-user who is required to perform a large number of functions in addition to monitoring the Telecom link between ground and spacecraft, The user interface is a mouse driven, Motif-based GUI consisting of the three resizable windows shown in Figure 3. These include a status window containing alarm and message status counters and a graphical data quality indicator, an alarm and message summary window, and a utility window for viewing diagnostic messages, telemetry values, and other information. The status window acts as the main window and is the only window visible at start-up. To conserve screen real estate, the window begins in a reduced form displaying Summary and Utility push buttons to pop Up corresponding additional windows, a Show All button to display all windows, and a Status button to toggle back and forth between the main window's compressed and expanded states.

When an anomaly is detected by the expert system, the Status button turns red. The color change is accompanied by a beep to alert the user. When the status button is pressed, the window expands to display the Alarm and System Status counters and the data quality indicator. Each counter consists of two fields, one displaying the time of the most recent reset, and the other the number of anomalies the expert system has detected since the last time the counter was reset. When the expert system detects an anomaly or sends a status message to the GUI, the counter fields will turn either red in the Alarm Status display to signal an anomaly or yellow in the System Status display to signal a system message and the corresponding count field will be updated by one. The user can manually reset a single status counter by selecting the Reset button, thereby resetting the counter to zero, resetting the counter time to the current system time, and converting the counter fields to their normal color. The Ack (acknowledge) buttons change the counter fields back to their normal color only. The Reset and Ack buttons in the Alarm Status display will also change the Status button at the top of the display to its



93/160 10:48:51 : Anomaly: [illegible]
[illegible]

DSS receiver is in lock.
[illegible]
[illegible]
[illegible]
SSNR is below threshold.
[illegible]
DSS AGC is below limit.
[illegible]
[illegible]
[illegible]
[illegible]
DSS XGC is below limit.
[illegible]
[illegible]

'igue 3. The windows of the graphical user interface.

non-alarm color. The Status window also contains a graphical data quality indicator which consists of a four-part scale and slider button whose position and color indicate the quality of the telemetry currently being received (from “Noisy” to “Error Free”).

The Summary button pops up the Summary window, which is comprised of two scrollable paned windows that display brief one-line descriptions of the anomaly and informational messages, These messages are provided by the expert system. When the user double clicks on an anomaly message in the alarm message summary window, the Utility window pops up and displays a complete diagnostic message describing the anomaly and a recommendation for corrective action. The user may also view multiple diagnostic messages by selecting one or more anomaly messages and pressing the Diagnosis button: this pops up the Utility window and displays the information in the workspace. The user can also press either the Alarms or Messages buttons in the Summary window to pop up and display more detailed information pertaining to the messages in the Utility window workspace.

The Utility window is a scrollable user workspace area dedicated to displaying information of current interest to the user, such as diagnostic messages, alarm or system summary lists, or a table of telemetry data. The SOE button at the bottom of the window launches the Sequence of Events (SOE) file viewer and editor, which enable the analyst to see the scheduled spacecraft activities integrated with information from the DSN. The Data button displays a table of incoming telemetry channel data including columns for the channel type and number, the symbolic description of the channel (mnemonic), the time tag of the telemetry, the engineering value, the data number, and the predicted value (predict) for those channels for which predicts are available.

DATABASE FILES

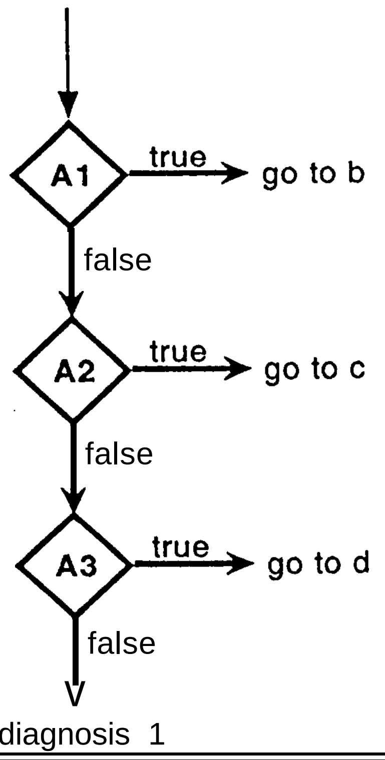
Certain data that is required by the knowledge base originates in the predict files and SOE files. Two C-ISAM (indexed sequential access method) database files are maintained, one for predict data and one for SOE data. A set of C functions invoked from the expert system access these databases to facilitate the retrieval of information. When SHARP is initialized, the SOE files in the directory of SOE data are processed to extract information concerning specific events that is used by the knowledge base, The data is compounded into a single C-ISAM SOE database file for easy retrieval when the data is requested. Also at initialization, the data in the predict files for each DSN station are transferred into a C-ISAM predict database file. Every ten seconds, the directories containing the SOE data and predict data are checked for added files that are newer than the corresponding C-ISAM file. If a newer SOE file is detected, the current C-ISAM SOE file is destroyed and re-created using all the SOE files in the directory. If a newer predict file is detected, the relevant contents of that file are added to the C-ISAM predict file.

THE KNOWLEDGE BASE

The SHARP knowledge base embodies the diagnostic logic of a Telecom expert. This domain knowledge was supplied by a senior Telecom analyst in the form of a binary decision tree. Each time a complete set of telemetry is received from the Ground Data System, the top leve node in the decision tree is evaluated. Based upon the value of the data examined at that node, one of two branches is taken to the next node in the tree. This process continues until a leaf node is encountered, at which time the anomaly causing the problem is obtained, along with a recommendation on how to fix the problem.

At each node one or more data values is examined, This data can come from one of several sources, including the telemetry, the predict database, and the SOE database. Sometimes the data is not directly available and must be derived or calculated from other data. Sometimes the data value is not available from any on-line data source (e.g., the weather condition at the DSN station). In this case the user must be asked to supply the value of the data. A pop-up window appears with a question for the user at the top of the window and a menu of possible answers below the question. The user supplies the answer by selecting the menu choice with the mouse and clicking on the "OK" button, or by simply double-clicking on the menu choice.

Whenever an expert system must enter a consultation mode to ask for data from the user, it is important to constrain the questions so that they pertain only to the current analysis, rather than asking about data from a decision tree branch that can be excluded from the search because it is irrelevant [Shortliffe 1976]. The same consideration applies to complex calculations that must be performed to transform data into a form that can be properly analyzed (e.g., Fourier analysis). In order to meet real-time constraints, the system must not perform time-consuming calculations of values that are not essential to the current analysis.



```
a(X) :- Al, b(X).
a(X) :- A2, c(X).
a(X) :- A3, d(X).
a(diagnosis1).
```

Figure 4. A generic decision tree and its Prolog representation

In a traditional forward-chained diagnostic system all or most data values must be asserted in working memory prior to the activation of any rules. Subsequently, rule firing is initiated causing relevant rules to be activated by a subset of the data in working memory [Forgy 1982]. Assertion of all data values in SHARP would require asking the user all possible questions and performing ail possible data transformations before any rule was fired. Since this would be bothersome to the user and inefficient, a forward-chained inference engine was not a viable option for SHARP.

In order to circumvent this constraint, the rules were written in Prolog, a goal-driven backward-chained language, Previous versions of SHARP employed a LISP-based inference engine and the knowledge base had a significantly different design from the current implementation. Although a forward-chained language such as NASA's CLIPS can be used to obtain data in a more efficient manner (e.g., by simulating backward-chaining) [Hayes-Roth 1983], this tends to dilute the advantages of forward-chaining [Winston 1984] as well as increasing the number of rules significantly. For this application, it was therefore preferable to simply use a backward-chained rule language.

In the Prolog knowledge base, each node of the decision tree is represented by a single rule. Figure 4 shows the relationship between a generic decision tree and the Prolog implementation of that tree. In the decision tree in figure 4, Al, A2, and A3 are decisions to be evaluated, such as "IS the telemetry value for channel X out of limit?", or "IS there bad weather at the DSN station?". B, c, and d are other subtrees of the same form as shown in the figure, that are invoked if one of the Al -A3 are true. "Diagnosis" is the default diagnosis that is concluded if Al -A3 are all false. Although only one binary decision is made at each node, there may be multiple data values that are examined in order to arrive at that decision. Thus Al -A3 each represent one or more goals to be evaluated. These goals may include calls to C functions that will return the values of telemetry, predicts, or SOE information, or ask the user for a data value.

To invoke the Prolog rules in figure 4, the goal *a(X)* is called (where X is an unbound variable). If Al, A2, or A3 is true, then *b(X)*, *c(X)*, or *d(X) is* called respectively, passing X on to be instantiated by one of the rule subtrees. When a diagnosis is determined it is unified with X, making it accessible by the top level rule that originally called *a(X),*

The Prolog knowledge base (KB) is embedded in a C program. Whenever a complete set of telemetry values is received from the GDS(approximately every 15 seconds, under ideal conditions), the main C program stores the values in global C structures for later retrieval. Then C invokes the Prolog KB by calling an interface function that calls the top level Prolog goal. The rules are evaluated in the manner described above. If a data value is needed in order to make a decision and the value is available from an on-line data source, a C function is called directly from the Prolog rule to retrieve the required value. If a decision requires a value that is not available on-line, the ruie calls a C function that consults the user for the answer by popping up a query window. At that point all telemetry processing and KB inferencing is suspended until the user responds with an answer. When the user inputs an answer, the C function that popped up the query window will return the answer to the Prolog rule that called it, and the rule will then continue on with its evaluation of the current decision node. Note that the suspension of all processing is achieved by virtue of the fact that control will not return to the Prolog rule until the C query function has terminated, and the main C function cannot continue its execution until the Prolog KB has terminated execution,

Each time a decision tree node is visited, the identifier of the rule associated with that node is recorded in the KB. If the rule eventually fails, the identifier is modified to indicate that the node is false. Upon completion of inferencing the user can request an explanation of the steps taken in the diagnosis process. In response to this request, the list of recorded nodes from the most recent diagnosis is translated into English and output to the user in the order in which the nodes were visited.

Occasionally the rules will require that some action be taken in order to attempt to fix a problem that has been diagnosed by the rules, and the result of the fix must be evaluated to determine if any other action is required. For example, if it has been determined that it is raining at the DSN station, then the user must request the station to turn on a blower to evaporate the moisture at the antenna. If the problem is fixed by the action then the KB terminates inferencing and control is returned to the main C function. If the problem is still not fixed by the action, then a different corrective action must be attempted, or the KB will determine that the problem cannot be currently fixed. The only way for the KB to determine whether the problem has been fixed is by examining the latest telemetry *after the action has been performed and the effects of the action are visible in telemetry.* Thus while KB processing is suspended pending the execution of the action, the telemetry must continue to flow through the system, without being evaluated, until the effects of the action (if any) show up in telemetry.

When this type of action must be performed, the Prolog rule first suspends KB inferencing by adding (asserting) two facts to the KB, one to indicate that the KB is in a suspended state, and the other to specify the goal to be called upon resumption of processing. Next a C function is called which pops up a window with a message asking the user to perform the required action. Unlike the C function for simple user-queries, the action-request function spawns a new process to pop up the window, which executes independently from the other SHARP processes. The action-request function can then terminate and return control to the calling Prolog rule while the pop-up action-request window is still displayed and waiting for user input. The Prolog rule then completes execution and control returns to the main C function. When the action has been performed and enough time has passed so that its effect will be visible in telemetry, the user clicks on the "OK" button in the action-request window, signifying that the KB can resume inferencing.

While the KB is in the suspended state, the main C function resumes normal processing, reading telemetry from the GDS and calling Prolog whenever a complete telemetry set is received. It is important that KB inferencing is not restarted until the state of the Telecom system is changed by the action that has been requested. When the top-level Prolog rule is called while in the KB suspended state (as determined by the previously asserted fact) the rule first calls a C function to see if the user has responded to the action request. If the user has responded, indicating that the action has been performed, then the the previously asserted goal (to be executed upon KB resumption) is retrieved from the KB. The goal is then called, which has the effect of returning inferencing to the point where execution had left off just prior to the suspension of the KB. At that point the current telemetry values should reflect the effects of the requested action, so the KB can continue its diagnosis using the latest available data.

If the user has not yet responded to the action-request window, then the top-level rule does nothing further and returns control to the main C function. This causes the latest set of telemetry to be ignored by the KB, because it does not yet show the effects of the action to be performed, and thus does not reflect the state of the Telecom system that the rules expect to exist upon resumption of inferencing.

BENEFITS OF SHARP IN MISSION OPERATIONS

SHARP enables faster response time in the detection and diagnosis of Telecom anomalies, due to two factors. First, since SHARP automates the knowledge of an expert Telecom analyst, the Telecom expert will no longer need to be consulted for problems that are within SHARP's domain. This will eliminate the time needed to contact the expert and apprise him of the situation. Secondly, the actual time required to detect the anomaly and then to diagnose the cause of the problem and recommend how to fix it, will be much faster than any human operator or domain expert could perform.

Faster response time will result in increased safety for the spacecraft, since any interruption in communication with the spacecraft could allow serious spacecraft anomalies to go undetected and uncorrected, possibly leading to the loss of the spacecraft, as occurred in the case of the Mars Observer. Also, even in the absence of serious spacecraft anomalies, faster response time means that any downtime of the Telecom link will be minimized, thus minimizing the loss of both science and engineering data transmitted from the spacecraft.

SHARP will also enable reduction in operational staff size to take place with minimal impact. Past interplanetary missions have staffed the function now carried out by SHARP with a full-time Telecom expert. More recent missions have been under significant pressure to reduce costs, causing them to eiiminate a variety of previously funded positions, including dedicated Telecom support. They have relied instead on ad-hoc support, drawing from the institutional pooi of expertise on an as-needed basis. SHARP reduces their dependence on this sometimes cumbersome approach, enabling mission controllers to derive on-line consultation from SHARP for ail but the most unusual Telecom problems.

Also, since SHARP automates Telecom anomaly detection knowiedge, routine monitoring can be performed by operators whose primary expertise is not in Telecom.

This means that a single person can monitor both Telecom and other non-Telecom subsystems, allowing additional reductions in Telecom staff size. For Galileo this is important because a single operator will sometimes be required to monitor multiple subsystems concurrently due to staffing limitations.

Another benefit of SHARP is that it will help to train inexperienced analysts by taking them step by step through the diagnosis process, via the knowledge base explanations. For the experienced analyst SHARP will serve as a reminder of what parameters are of importance in a particular analysis, and what actions are appropriate in order to solve a given problem. A final benefit will be the preservation of Telecom expertise in the SHARP knowledge base, after key Telecom experts have retired or moved on to other missions.

LESSONS LEARNED IN THE TRANSITION TO OPERATIONAL USE

As discussed previously, SHARP was originally developed for the Mars Observer mission. One week after the delivery of SHARP ail communication with the spacecraft was inexplicably lost. At that time, the mission control team had not yet had a chance to fully integrate the application into their monitoring routine. However, during SHARP'S brief period of exposure to operations personnel we gained some insight as to the potential areas where SHARP could be improved.

One area for improvement is the interactive consultation mode. When the system was connected to the real-time telemetry source it became apparent that the volume of data, (approximately one telemetry set per fifteen seconds) could cause frequent activation of pop-up user queries during anomalous situations, This is because the decision tree is invoked each time a telemetry set is received. In the original implementation, there were a total of eighteen possible questions in the KB that could be asked of the user, not counting the action request queries. Of course not all of these questions, or even a majority of them, would ever be asked during the diagnosis of a single telemetry set. But even one question every few minutes would be bothersome to the user since it would require constant attention. Pop-up queries suspend further inferencing until the user responds, and any new incoming telemetry is saved to a queue. The queue will hold about one hour's worth of telemetry. When the queue reaches its full capacity, the next telemetry set causes the system to clear the queue by simply discarding the oldest thirty minutes of data. So if the user leaves the workstation for an hour or more after a question has been posed without answering the question, when they return they will have at least thirty minutes of telemetry backed up in the queue which will have to be processed. The actual SHARP processing time for the queued data will be much faster than thirty minutes, but if there is an average of one question per twenty telemetry sets say (about one question for every five minutes of data), then there will be six questions that the user has to respond to before the system will begin processing the most current telemetry. Clearly this is a burden on the user that should be avoided.

We are making adjustments to the Galileo knowledge base that will reduce the number of user queries. Some of the data values that were not available in telemetry for Mars Observer are now available in Galileo data, and a few other unavailable values can now be calculated from on-line data. This will allow the total number of possible questions to be reduced to ten. Two of these are about conditions that will persist over time (bad weather and precipitation), so they need only be asked at most once during the extended time interval that such conditions would continue to exist (the exact length of the intervals is to be determined). In addition, instead of analyzing every telemetry set, and possibly invoking user queries as often as every fifteen seconds, we will sample the telemetry so that the knowledge-based detection and diagnosis is invoked once every two minutes. After beta-testing is complete, the interval between samples will be increased if monitoring constraints allow, or decreased if anomaly response time is inadequate. A longer time between telemetry sampling will require fewer invocations of the KB, which will decrease the likelihood that the user will be queried for data.

Another problem in querying the user was noticed during testing. If an anomaly appeared, successive sets of telemetry contained the same data, and therefore invoked the same path in the decision tree until the problem was fixed. If there were any user queries in that path, the user would have been asked the same questions repeatedly.To prevent this from happening, a redundant proof-tree path detection step was added to the rules. When a diagnosis is obtained, the path of decision nodes that led to that diagnosis, along with the key parameter value that was checked at each node, is now stored in the KB as a Prolog list. Whenever a user-query is required, the rules first check to see if the current partial node path is a leading subset of the path from the last diagnosis, and the key parameter values of corresponding nodes are equal. If so, then the current anomaly is assumed to be the same as the last anomaly, the user is not queried, and the KB aborts and returns control back to the top-level C function without any further processing. This avoids asking the user redundant questions about an anomaly that has already been diagnosed, and it improves system throughput by preventing unnecessary inferencing.

OTHER NASA SYSTEMS FOR MONITORING AND DIAGNOSIS

Other monitoring and diagnosis systems have been developed within NASA for a number of related applications. The evolution of each system has been driven predominantly by the specific domain requirements, resulting in both substantial and subtle differences.

The MARVEL system, aiso developed at JPL [Schwuttke 1992, Schwuttke 1994], features multiple distributed processes, including up to five cooperating expert systems. it has been operational on a network of UNIX workstations since 1989. MARVEL is used by both the Voyager and Galileo missions, providing automated monitoring and data-driven diagnosis of these spacecraft at both the subsystem and system levels and providing a variety of additional features such as trend analysis and automatic report generation that contribute to productivity enhancement and workforce reductions. The two applications of MARVEL are each fairly application specific, with approximately eighty percent of each system being tailored to unique mission requirements. The automated monitoring and hierarchical alarming in MARVEL is currently being implemented in a highly generic, user-customizable product.

The Real-Time Data Systems (RTDS) at Johnson Space Center uses heuristic, associative reasoning to support automated monitoring and diagnosis in the Space Shuttle Control Center [Muratore 1989, Heindel 1990]. RTDS is an application environment that provides telemetry data acquisition, management, and display as well as diagnosis functions. It has been on-line since 1986, focussing predominantly on real-time monitoring, and consisting of a distributed set of applications in a large network of workstation. RTDS has a capacity for monitoring over 4000 telemetry parameters, As a result of the overwhelming real-time demands on this system, there has been significantly less effort on automated diagnosis in RTDS than in some of the other NASA applications, as diagnostic functions tend to be the most computationally expensive, This system has recently been adopted as a standard for future automated monitoring and diagnosis stations in the Space Shuttle Control Center.

The CLEAR and GenSAA expert systems [Hughes 1991] support mission operations at Goddard Space Flight Center. CLEAR is a forward-chaining rule-based system that detects anomalies in the telecommunications link between the COBE and TDRSS spacecraft. Detected anomalies are portrayed graphically to the human operator. GenSAA is a general shell that was influenced by CLEAR. In addition to the forward-chaining rule-based systems approach used in CLEAR, GenSAA provides a workbench environment for developing applications as well as tools for linking the system with real-time data sources.

SUMMARY

SHARP automates the knowledge of an expert in the domain of spacecraft telecommunications. The knowledge base uses backward chaining to detect and diagnose Telecom anomalies by examining telemetry and other on-line data, and by obtaining information from the user through an interactive consultation mode. Some unexpected problems with the consultation mode were discovered during testing, but the latest version of SHARP for Galileo mitigates these problems to some extent. SHARP makes workforce reductions less painful by reducing the dependence on Telecom domain experts, and by allowing mission analysts from other subsystem domains (e.g., attitude control, power) to perform Telecom monitoring and analysis. SHARP also provides faster diagnosis and correction of anomalies than previously possible..

ACKNOWLEDGMENT

The work described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology under a contract with the National Aeronautics and Space Administration. The authors wish to acknowledge the efforts of Scott Burleigh on the telemetry interface and the innovation of Mark James on prior versions of SHARP. The authors appreciate support from the NASA Office of Advanced Concepts and Technology and JPL's Multimission Operations Support Office.

REFERENCES

D. Atkinson and M. James, "Spacecraft Health Automated Reasoning Prototype: Applications and Recent Progress," Proceedings of First International Conference on Fielded Applications of Intelligent Software Technologies (Toulouse-Labege, France, 1992).

C. Forgy. "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem." Artificial Intelligence 19, 1982, 17-37.

F. Hayes-Roth. *Building Expert Systems.* Reading Mass.: Addison-Wesley, 1983, pp. 172-215.

T. Heindel and J. Muratore, "Advanced Automation in Space Shuttle Mission Control," Proceedings of the Second International Symposium on Space Information Systems, American Institute of Aeronautics and Astronautics (Pasadena, CA, 1990), pp. 641-650.

P. Hughes and E. Luczak, "GenSAA: Advancing Satellite Monitoring with Expert Systems," AIAA Computing in Aerospace 8: A collection of Technical Papers, Vol. 1, (American Institute of Aeronautics and Astronautics, Baltimore, MD, 1991), pp. 293-303.

R. Martin (cd.), D. Atkinson, M. James, D. Lawson, H. Porta. "Spacecraft Health Automated Reasoning Prototype (SHARP): A Report on SHARP and the Voyager Neptune Encounter". Publication 90-21. Jet Propulsion Laboratory. Pasadena, CA. August 1990.

J. Muratore, T. Heindel, T. Murphy, A. Rasmussen and R. McFarland. "Space Shuttle Telemetry Monitoring by Expert Systems in Mission Control," Innovative Applications of Artificial intelligence, H. Schorr and A. Rappaport, Editors, (AAAI Press, 1989), pp. 3-14.

U.M. Schwuttke, A.G. Quan, R. Angelino, et al, "MARVEL: A distributed Real-time Monitoring and Analysis Application" Innovative Applications of Artificial intelligence 4, A. Scott and P. Klahr, Editors, (AAAI Press, 1991), pp. 89-106.

U.M. Schwuttke, A.G. Quan, and J.R. Veregge, "Cooperating Expert Systems for the Next Generation of Real-time Monitoring Applications ", to appear in the Proceedings of the International Conference on Expert Systems for Development, March 1994.

E.H. Shortliffe, and B.G. Buchanan. *Computer-based medics/ consultation: MYCIN. New York:* American Elsevier, 1976.

P. Winston. *Artificial Intelligence.* Reading Mass: Addison-Wesley, 1984, pp. 152-153.