A Planning Approach to Monitor and Control For Deep Space Communications[1]

Forest Fisher, Russell Knight, Barbara Engelhardt, Steve Chien, Niko Alejandre
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive, M/S 126-347
Pasadena, CA 91109-8099
818-393-5368
Forest.Fisher@jpl.nasa.gov

Abstract— In recent years with the large increase in the number of space missions at NASA and JPL (Jet Propulsion Laboratory), the demand for deep space communications services to command and collect data from these missions has become more difficult to manage. In an attempt to increase the efficiency of operating deep space communications antennas, we are developing a prototype system to perform monitor, control, execution and recovery in order to automate the operations of the Deep Space Network (DSN) communication antenna stations.

This paper describes the application of Artificial Intelligence planning techniques for antenna track plan generation and monitor and control for a NASA Deep Space Communications Station. The described system, CLEaR (Closed Loop Execution and Recovery), will enable an antenna communications station to automatically respond to a set of tracking goals by correctly configuring the appropriate hardware and software and providing the requested communication services, while adapting itself to its dynamic environment. To perform this task, the Continuous Activity Scheduling, Planning, Execution and Replanning (CASPER) engine has been applied and extended to automatically produce antenna tracking plans that are tailored to support a set of input goals. Then during the execution of these track plans, CLEaR monitors the execution and adapts the track plan to the changing environment. In this paper, we will describe the antenna automation problem, the CASPER planning and scheduling system, how CASPER is used to generate antenna track plans and perform monitor and control during execution, and future work utilizing dynamic planning technology.

TABLE OF CONTENTS

1. INTRODUCTION

The Deep Space Network (DSN) [5] was established in 1958 and since then it has evolved into the largest and most sensitive scientific telecommunications and radio navigation network in the world. The purpose of the DSN is to support unmanned interplanetary spacecraft missions and support radio and radar astronomy observations in the exploration of the solar system and the universe. The DSN currently consists of three deep-space communications facilities placed approximately 120 degrees apart around the world: at Goldstone, in California's Mojave Desert; near Madrid, Spain; and near Canberra, Australia (figure 1). This strategic placement permits constant observation of spacecraft as the Earth rotates, and helps to make the DSN the largest and most sensitive scientific telecommunications network in the world. Each DSN complex operates a set of deep space stations consisting of: one 70-meter antenna, a collection of 34-meter antennas, one 26-meter antenna, and 11-meter antennas (figure 2). The functions of the DSN are to receive telemetry signals from spacecraft, transmit commands that control the spacecraft operating modes, generate the radio navigation data used to locate and guide the spacecraft to its destination, and acquire flight radio science, radio and radar astronomy, very long baseline interferometry, and geodynamics measurements.

From its inception, the DSN has been driven by the need to create increasingly more sensitive telecommunications devices and better techniques for navigation. The operation of the DSN communications complexes requires a high level of manual interaction with the devices in the communications link with the spacecraft. In more recent times NASA has added some new drivers to the

---

[1] 0-7803-5846-5/00/$10.00 © 2000 IEEE

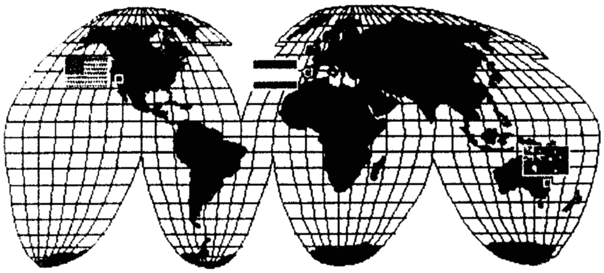

Figure 1 World Map of Deep Space Network

development of the DSN: (1) reduce the cost of operating the DSN, (2) improve the operability, reliability, and maintainability of the DSN, and (3) prepare for a new era of space exploration with the New Millennium program: support small, intelligent spacecraft requiring very few mission operations personnel [10].

In order to address these new requirements for the DSN, we have worked on antenna station automation. In this paper we describe the Closed Loop Execution and Recovery (CLEaR) system being developed to address the problem of automated track plan generation (i.e. automatically determining the necessary actions to set up a communications link between a deep space antenna and a spacecraft), monitor, control, execution and recovery for the DSN. In our approach we are utilizing artificial intelligence (AI) planning and scheduling techniques to generate the track plans, and we are utilizing a continuous planning approach to provide monitor, control, execution and recovery. Similar to many planning problems, track plan generation involves elements such as subgoaling to achieve preconditions and decomposing high-level (abstract) actions into more detailed sub-actions. However, unlike most classical planning problems, the problem of track generation is complicated by the need to reason about issues such as metric time, DSN resources and equipment states. To address this problem, we have applied the Continuous Activity Scheduling Planning Execution and Replanning (CASPER) engine, a generic framework for automated planning, scheduling, execution and replanning, to generate antenna track plans on demand [3,4].

CASPER is a soft real-time planning, scheduling and execution framework built on top of the Automated Scheduling and Planning ENvironment (ASPEN) [1,9], which in turn is a generic planning and scheduling system being developed at JPL that has been successfully applied to problems in spacecraft commanding and maintenance scheduling and antenna track plan generation. CASPER/ASPEN utilizes techniques from Artificial Intelligence planning and scheduling to automatically generate the necessary antenna command sequence based on input goals. This sequence is produced by utilizing an "iterative repair" algorithm [9,12,17], which classifies conflicts and resolves them each individually by performing one or more plan modifications. This system has been adapted to input antenna-tracking goals and automatically produce the required command sequence to set up and perform the requested communications link.

Figure 2 70-Meter Deep Space Communication Antenna

This work is one element of a far-reaching effort to upgrade and automate DSN operations building on previous work. The ASPEN Track Plan Generator, which was demonstrated in support of the Deep Space Terminal (DS-T), an autonomous prototype 34-meter deep space communications station [6,7,8], produced batch plans with limited conditionals for error recovery. CLEaR is the continuation of the automation concepts introduced during DS-T but is intended to demonstrate a greater level of automation and robustness while providing a larger class of communication services.

The rest of this paper is organized in the following manner. We begin by introducing the reader to the deep space communications domain and characterizing the current mode of operations for the DSN. We then describe how we apply artificial intelligence (AI) planning and scheduling techniques to perform autonomous monitor, control, execution and recovery functions in order to automate communication antenna stations operations. Next we describe the continuous planning technique and the CASPER system, and we conclude by discussing future work, related work and summarize how this work enables closed-loop control and automatic error recovery when executing DSN antenna tracks.

2. HOW THE DSN OPERATES

The DSN track process occurs daily for dozens of different NASA spacecraft and projects, which use the DSN to command spacecraft, as well as capture spacecraft and science data. Though the process of sending signals from a spacecraft to Earth is conceptually simple, in reality there are many earthside challenges that must be addressed before a spacecraft's signal is acquired and successfully transformed into useful information. In the remainder of this section, we outline some of the steps involved in providing tracking services and in particular discuss the problem of track plan generation.

The first step in performing a DSN track is called network preparation. Here, a project sends a request for the DSN to track a spacecraft involving specific tracking services (e.g. downlink, uplink). The DSN responds to the request by attempting to schedule the necessary resources (i.e. an antenna and other shared equipment) needed for the track. Once an equipment schedule and other necessary information has been determined, the next step is the data capture process, which is performed by operations personnel at the deep space station. During this process, operators determine the correct steps to perform the following tasks: configure the equipment for the track, perform the actual establishment of the communications link, and then perform the actual track by issuing control commands to the various subsystems comprising the link.

Throughout the track the operators continually monitor the status of the communications link and handle exceptions (e.g. the receiver looses signal lock with the spacecraft) as they occur. To perform all of these actions, human operators manually issue tens to hundreds of command directives via a computer terminal. This paper discusses the CLEaR system being developed as a prototype monitor, control (M&C) and execution system for DSN communication antenna automation.

3. MONITOR AND CONTROL THROUGH CONTINUOUS PLANNING

In the section "How the DSN Operates," we presented a high level view of DSN operations. In this section we will focus specifically on the steps involved in performing automated communication services. The system architecture we are developing, CLEaR, views the antenna station as an autonomous unit within the DSN. It is here at the antenna station, referred to a deep space station (DSS), that the CLEaR automation engine is intended to be deployed. Of course in today's world of computer networks it is not necessary for the software or even the computer to physically reside at the station itself. Regardless of the physical proximity of the automation system, the station is said to have a station controller that is responsible for determining and controlling the behavior of the station. It is this functionality that the CLEaR system provides as the primary control module of the station controller's automation software.

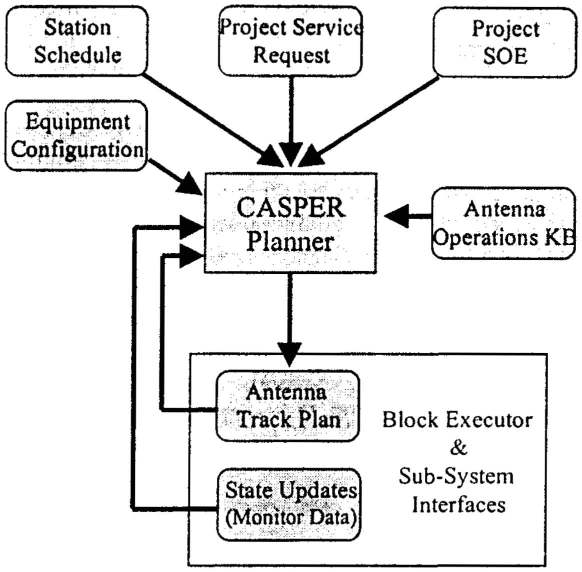

Figure 3 Inputs and Outputs to CLEaR

Given a set of inputs: a station schedule, service request, spacecraft sequence of events (SOE), equipment configuration, an antenna operations knowledge base (KB), an track plan if one exist, and station state information the automation system produces a track plan or control script (figure 3). These control scripts are referred to as Temporal Dependency Networks (TDNs). The TDN scripts are made up of smaller components, called ALMO blocks which are executable scripts implemented in the Automation Language for Managing Operations (ALMO) scripting language [14]. The ALMO blocks are represented in the knowledge base as planning activities. The knowledge base expresses the behavior of the blocks (pre and post conditions) as well as temporal relation, temporal estimates on execution times, resource (equipment) usage, and domain knowledge which are all used to determine the necessary steps involved in providing the high level service request.

The service request represents the high-level communication services that must be performed. These services might include downlink data at a given frequency and bitrate, then transmit (uplink) a new spacecraft command sequence. The service request is used in conjunction with the spacecraft SOE to create the planning goals inputted into the planning engine. It is necessary to use the spacecraft SOE because in order to maintain the communication link with the spacecraft the ground system must be aware of and synchronized with the communications activities of the spacecraft. The types of information that are expressed in the SOE are the modes the spacecraft is in (whether transmitting, receiving, or both, what frequency and bitrate, etc.) and the times which those modes will change.

From this set of inputs (mentioned above) CLEaR considers the goals, which are extracted from the service request and SOE, within the context of the station configuration provided and then produces an initial track plan (control script) based on the available operations defined in the antenna operations knowledge base (KB). In order to produce the track plan (control script) CLEaR utilizes artificial intelligence (AI) planning and scheduling techniques provided by the continuous planning system CASPER (Continuous Activity Scheduling, Planning, Execution and Recover)[3,4], which is further described in the next section.

Once the TDNs are produced, the executive component of the CLEaR system begins stepping through the plan. As time progresses through the plan and the start time of an activity (block) arrives the block is sent to the ALMO script interpreter and the block is executed, which results in command directives being sent to the appropriate equipment (subsystems). Each of these subsystems in turn produces feedback information in the form of monitor data. This monitor data is fed back into the continuous planner and used to update the state. If the predicted state of the planner differs from the true (as we understand it) state of the world (the station), the planner will begin to massage the plan in order to regain consistency with the original plan. It is important that the plan converges quickly because the spacecraft will continue with its intended communication sequence whether anyone on the ground is listening or not. As this might imply, the ground station is not responsible for determining what communication should take place, but how to control the ground station equipment to provide the requested communication service. It is in this fashion that the monitor, control, execution and recovery are performed through the utilization of a continuous planning approach.

A communication track or pass is broken up into three portions: pre-track, in-track and post-track. Pre-track consists of configuring the station's equipment to perform the requested communications services. During the configuration phase, equipment is powered on, warmed up, configuration files loaded, etc. Once the appropriate configuration has been performed the station is ready to perform the desired communication services. The next portion of the track is the in-track phase. It is here where the actually communication service is performed. At a high level, this consists of transmitting and receiving of data. During this phase the station must be commanded to maintain the antenna pointing at the spacecraft, to acquire and maintain a signal as well as transmit. All of these activities require that the equipment be commanded within very tight tolerances. At the conclusion of the track, the post-track phase is performed to return the station to a standby state and wait for the next scheduled communication pass. This last phase includes steps such as archiving data, generating reports, data deliver (of data that was not delivered in real-time) and commanding the sub-systems into a standby state.

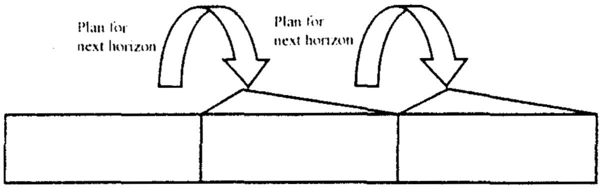

Figure 4 Traditional Batch "Plan then Execute" Cycle

The steps in each of these three phases vary depending on the types of service requested. While there is a large possible set of combinations of services, the services can all be categorized into one of four basic service types: Doppler, Telemetry, Commanding, and Ranging.

Doppler service refers to tracking the spacecraft as it moves across the sky and adjusting the receiver's frequency to adjust for the Doppler shift. The receiver is used to confirm that the spacecraft is being "tracked" by the ground station.

Telemetry service refers to the collection (downlink) of spacecraft health data (engineering telemetry) and science data (science telemetry).

Commanding service refers to the transmitting (uplink) of command sequence to the spacecraft.

Ranging service refers to the process of confirming the position of the spacecraft and is used to confirm the spacecraft's trajectories.

Each of these four services are additive onto the previous service (i.e. ranging involves all four, while commanding would involve Doppler, telemetry and commanding but not ranging).

Many of the services result in complex interactions between command directives. Other variables in the mix that complicate the process are the vast combinations of possible equipment configuration. Although the steps involved in controlling the antenna do not differ greatly whether the antenna station is being controlled manually or autonomously, all of these factors contribute to the difficulty of communication antenna operations.

4. CASPER

Integrating Planning and Execution

Traditionally, much of planning and scheduling research has focused on a batch formulation of the problem. In this approach (see Figure 4), time is divided up into a number of planning horizons, each of which lasts for a significant period of time. When one nears the end of the current horizon, one projects what the state will be at the end of the execution of the current plan. The planner is invoked with a new set of goals and this state as the initial state (for example the DS1 (Deep Space One) Remote Agent Experiment operated in this fashion [13].)

This approach has a number of drawbacks. In this batch oriented mode, typically planning is considered an off-line process which requires considerable computational effort, hence there is a significant delay from the time the planner is invoked to the time that the planner produces a new plan. If a negative event occurs (e.g., a plan failure), the response time until a new plan may be significant. During this period the system being controlled must be operated appropriately without planner guidance.

If a positive event occurs (e.g., a fortuitous opportunity), again the response time may be significant. If the opportunity is short lived (e.g., activities finishing early), the system must be able to take advantage of such opportunities without a new plan (because of the delay in generating a new plan).

Finally, because the planning process may need to be initiated significantly before the end of the current planning horizon, it may be difficult to project what the state will be when the current plan execution is complete. If the projection is wrong the plan may have difficulty.

For example, consider the operations of a spacecraft. In a traditional plan-sense-act cycle, planning occurs on a relatively long-term planning horizon. In this approach, operations for a spacecraft would be planned on the ground on a weekly or daily basis. The spacecraft state at the start of the planning horizon would be determined (typically predicted as the construction of the weekly plan, which would need to begin significantly before the week of execution). The science and engineering operations goals would then be considered, and a plan for achieving the goals would be generated. This plan or sequence would then be uplinked to the spacecraft for execution. The plan would then be executed onboard the spacecraft with little or no flexibility. If an unexpected event occurred due to environmental uncertainty or an unforeseen failure occurred, the spacecraft would be taken into a safe state by fault protection software. The spacecraft would wait in this state until the ground operations team could respond and determine a new plan. Correspondingly, if an unpredictable fortuitous event occurs, the plan cannot be modified to take advantage of the situation.

To achieve a higher level of responsiveness in a dynamic planning situation, we utilize a continuous planning approach and have implemented a system called CASPER (Continuous Activity Scheduling Planning Execution and Replanning) [3,4]. Rather than considering planning a batch process in which a planner is presented with goals and an initial state, the planner has a current goal set, a plan, a current state, and a model of the expected future state. At any time an incremental update to the goals, current state, or planning horizon (at much smaller time increments than batch planning) may update the current state of the plan and thereby invoke the planner process. This update may be an unexpected event or simply time progressing forward. The planner is then responsible for maintaining a consistent, satisficing plan with the most current information. This current plan and projection is the planner's estimation as to what it expects to happen in the world if things go as expected. However, since things rarely go exactly as expected, the planner stands ready to continually modify the plan. In each cycle from the point of view of the planner the following occurs:

- changes to the goals and the initial state first posted to the plan,
- effects of these changes are propagated through the current plan projections (includes conflict identification)
- plan repair algorithms are invoked to remove conflicts and make the plan appropriate for the current state and goals.

This approach is shown in Figure 5. At each step, the plan is created by using iterative repair with:

- the portion of the old plan for the current planning horizon;
- the updated goals and state; and
- the new (extended)planning horizon.

Even though our intent is to make the planning process very responsive (on the order of seconds), there still remains a synchronization process between planning and execution. We handle this by an activity commitment process. Execution has an activity commitment window that represents the near future. When an activity overlaps with this window (i.e. the activity is scheduled to begin very soon) it is committed. This means that the planner is forbidden from altering any aspect of this activity (such as by moving the activity or altering the activity parameters).

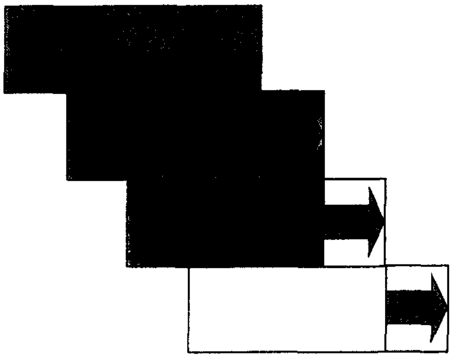

Figure 5 Continuous Planning Incremental Extension

Thus far we have focused on time-based commitment strategies (e.g., strategies that commit any activities scheduled to begin in the next T time units), however, our architecture supports more complex commitment strategies (such as it being dependent on the class of activity and allowing parameter changes later than activity moves, etc.).

In addition to increasing the responsiveness of planning, the continuous planning approach has additional benefits:

- The planner can be more responsive to unexpected (i.e., unmodeled) changes in the environment that would manifest themselves as updates on the execution status of activities as well as monitored state and resource values.
- The planner can reduce reliance on predictive models (e.g., inevitable modeling errors), since it will be updating its plans continually.
- Fault protection and execution layers only need to worry about controlling the spacecraft over a shorter time horizon (as the planner will replan within a shorter time span).
- Because of the hierarchical reasoning taking place in the architecture there is no hard distinction between planning and execution – rather more deliberative (planner) functions reside in the longer-term reasoning horizons and the more reactive (execution) functions reside in the short-term reasoning horizons. Thus, there is no planner to executive translation process.

In conjunction with this incremental, continuous planning approach, we are also advocating a hierarchical approach to planning. In this approach, the long-term planning horizon is planned only at a very abstract level. Shorter and shorter planning horizons are planned in greater detail, until finally at the most specific level the planner plans only a short time in advance (just in time planning). This paradigm is illustrated in Figure 6. Within each of these layers, the planner is operating continuously in the mode described above. However, the length of the planning horizon, and the frequency with which the plan is updated varies. In the longer-term more abstract levels, the planning horizon is longer and the abstract plan is updated less frequently. In the more detailed short-term level, the plans are updated more frequently.

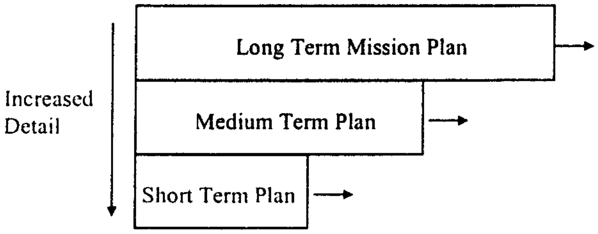

Figure 6 Hierarchical Planning Horizons

The idea behind this hierarchical approach is that only very abstract projections can be made over the long-term and that detailed projections can only be made in the short-term because prediction is difficult due to limited computational resources and timely response requirements. Hence there is little utility in constructing a detailed plan far into the future – chances are it will end up being re-planned anyway. At one extreme the short-term plan may not be "planned" at all and may be a set of reactions to the current state in the context of the near-term plan. This approach is implemented in the control loop described above by making high-level goals active regardless of their temporal placement, but medium and low-level goals are only active if they occur in the near future. Likewise, conflicts are only regarded as important if they are high-level conflicts or if they occur in the near future. As the time of a conflict or goal approaches, it will eventually become active and the elaboration/planning process will then be applied to resolve the problem.

An Architecture for Integrated Planning and Execution

Our approach to integration of planning and execution relies on four separate classes of processes.

- The Planner Process(es) - this process represents the planner, and is invoked to update the model of the plan execution, to refine the plan, or when new goals are requested.
- The Execution Process(es) - this process is responsible for committing activities and issuing actual commands corresponding to planned activities.
- The State Determination Processes - this process is responsible for monitoring and estimating states and resource values and providing accurate and timely state information.
- The Synchronization Process - this process enforces synchronization between the execution, planner, and state determination processes. This includes receiving new goals, determining appropriate timeslices for planning and locking the plan database to ensure non-interference between state updates and the planner.

We describe planning, execution, and state determination as sets of processes because often these logical tasks will be handled by multiple processes. For example, spacecraft attitude control execution might be handled by one process, data management by another, etc. However, for the purposes of this paper, the only relevant issue is that our synchronization strategy can be applied to a multiple process scheme for planning, state determination, etc.

The overall architecture for the continuous planning approach is shown in Figure 7. We now describe how each of the four basic components operates.

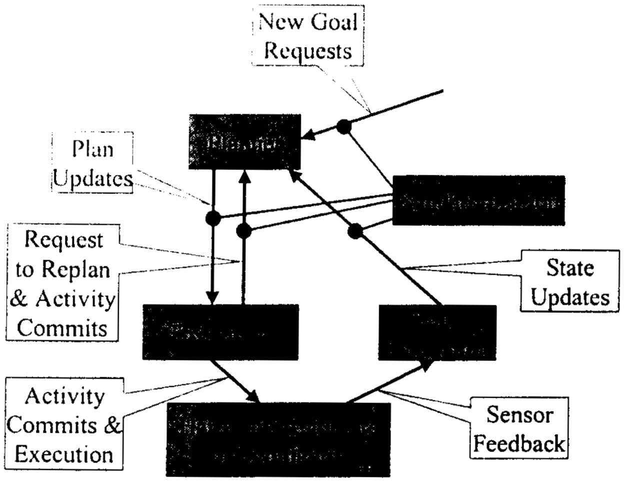

Figure 7 CASPER Architecture

The planner process maintains a current plan that is used for planning (e.g. hypothesizing different courses of action). It responds to requests to replan initiated by the execution processes, activity commitments from the execution module, state (and resource) updates from state estimation, and new goals (from external to the system). All of these requests are moderated by the synchronization process that queues the requests and ensures that one request is complete before another is initiated. The planner's copy of the current plan is also where projection takes place and hence it is here that future conflicts are detected. However, as we will see below, requests to fix conflicts occur by a more circuitous route.

The execution process is the portion of the system concerned with a notion of "now". The execution module maintains a copy of the plan that is incrementally updated whenever the planner completes a request (e.g., a goal change, state change, or activity change). This local copy includes conflict information. The execution module has three general responsibilities:

1. to commit activities in accordance with the commitment policy as they approach their execution time;
2. to actually initiate the execution of commands (e.g., processes) at the associated activity start times
3. to request re-planning when conflicts exist in the current plan

The execution module performs 1 & 2 by tracking the current time and indexing into relevant activities to commit and execute them. The execution module also tracks conflict information as computed by the projection of the planner and submits a request for replanning to the synchronization module when a conflict exists.

The state estimation module is responsible for tracking sensor data and summarizing that information into state and resource updates. These updates are made to the synchronization module that passes them on to the planners plan database when coordination constraints allow.

The synchronization module ensures that the planner module(s) are correctly locked while processing. At any one time the planner can only be performing one of its four responsibilities: (re)planning, updating its goals, incorporating a state update, updating the execution module's plan for execution, or updating commitment status (otherwise we run the risk of race conditions causing undesirable results). The synchronization module serializes these requests by maintaining a FIFO task queue for the planner and forwarding the next task only when the previous task has finished.

The execution module also has a potential synchronization issue. The planner must not be allowed to modify activities (through replanning) if those activities might already have been passed on to execution. We enforce this non-interference by "commit"-ing all activities overlapping a temporal window extending from now to some short period of time in the future (typically on the order of several seconds). We ensure that the planner is called in a way that each replan request will always return within this time bound and we enforce that the planner never modifies a committed activity. This ensures that the planner will not complete a replan with an activity modified that is already in the past. Additionally, we use the synchronization process to ensure that the Execution module does not commit activities while the planner is replanning. This prevents the planner from modifying activities that have been committed subsequent to the planner call (but still in the future).

5. STATUS

While the CLEaR task is ongoing with considerable implementation still to be done, successful preliminary work has been done.

The current status of CLEaR is an initial knowledge base model has been built to support the current NMC TDNs used to configure the station in the pre-track and post-track phases in order to support the four basic classes of service.

Currently the CLEaR knowledge base is being extended to support recovery for each of these four station configuration scenarios of pre-track and post-track. The next step, which we are addressing in parallel, is to produce control scripts for the in-track phases of the communication service as well as the knowledge to perform recovery for events during execution.

In order to validate our approach, we are integrating CLEaR into a prototype Deep Space Station Controller (DSSC). The DSSC system will consist of the CLEaR system to perform monitor, control, execution and recovery through the use of AI planning and scheduling techniques in order to perform the decision making process, and a FDI component mentioned in the next section, "Future Work." These technologies are being integrated with the existing NMC control software in order to enhance the capabilities of the automation infrastructure that has been developed.

6. FUTURE WORK

This CLEaR effort is also being integrated with a Fault Detection, Isolation and Recovery (FDIR) system [11]. FDIR is an expert system providing monitor data analysis. As is often the case with large complex systems, monitor (sensor) data is often related in unintuitive ways that are difficult for humans to detect. The advantage of combining these two systems is that FDIR can first interpret the vast amount of data and summarize it into a set of meaningful values for a planning system to react to. We think of this union as intelligent analysis and intelligent response, much like a careful design and implementation; one without the other is of little use.

Another area of future work is in the area of mixed-initiative control. This deals with how a system capable of autonomous operations interacts with an operator such that neither interferes with the other, and once control is returned to the autonomous system the system must understand both the state of the world and the changes that the user has made.

7. RELATED WORK

While the automation techniques utilized in the development of CLEaR rely heavily on AI planning and scheduling, we ask the reader to look to our papers on ASPEN [9] and CASPER [3,4] for work related to our planning and scheduling techniques.

There are a number of existing systems built to solve real-world planning or scheduling problems [15,16,17]. The problem of track plan generation combines elements from both these areas and thus traditional planners and schedulers cannot be directly applied. First, many classical planning elements must be addressed in this application such as subgoaling to achieve activity preconditions (e.g. the antenna must be "on_point" to lock up the receiver) and decomposing higher-level (abstract) activities into more detailed sub-activities. In addition, many scheduling elements are present such as handling metric time and temporal constraints, and representing and reasoning about resources (e.g. receiver, antenna controller) and states (e.g. antenna position, subcarrier frequency, etc.) over time.

Another approach to DSN antenna automation was taken by the Network Monitor and Control (NMC) task. NMC approach uses canned control scripts to automate antenna operations, compared to the CLEaR approach of dynamically constructing the control script out of smaller static scripts. This canned script approach is cumbersome, due to the large set of possible scenarios that may take place. It is because of this that management is considering the pursuit of integrating CLEaR into the larger NMC infrastructure.

Two other systems were previously designed to generate antenna track plans, the Deep Space Network Antenna Operations Planner (DPLAN) [2]. DPLAN utilizes a combination of AI hierarchical-task network (HTN) and operator-based planning techniques. Unlike DPLAN, ASPEN has a temporal reasoning system for expressing and maintaining temporal constraints and also has the capability for representing and reasoning about different types of resources and states. ASPEN can also utilize different search algorithms such as constructive and repair-based algorithms, while DPLAN uses a standard best-first based search. And, as described in the next section, ASPEN is currently being extended to perform dynamic planning for closed-loop error recovery, while DPLAN has only limited replanning capabilities.

For the reasons stated above the DS-T automation controller was developed using the ASPEN planning and scheduling system [8]. This greatly improved the capabilities for generating track plans, over the DPLAN system. DS-T utilized a classical approach of batch planning to produce the control script and then handed the script over to an execution environment. Unlike the DS-T approach, CLEaR utilizes the CASPER[2] planning and scheduling engine which enables CLEaR to perform dynamic replanning in response to changes detected during the execution of the control scripts. This tighter coupling of planning and execution provides substantial benefits and enables closed loop control.

8. CONCLUSION

This paper has described the Closed Loop Execution and Recovery (CLEaR) system and the manner in which it performs the monitor and control functionality for track automation of DSN communication antennas. Through the use of CASPER, CLEaR utilizes a knowledge base of information on tracking activity requirements and a combination of Artificial Intelligence planning and scheduling techniques to generate antenna track plans that will correctly setup and perform a communications link with spacecraft. The monitor and control capabilities are further enhanced by dynamically feeding monitor data (sensor updates) back into the planning system as state updates, which enables the planning system to verify the validity of the current plan. If a violation is found in the plan, the system will perform local modification to construct a new valid plan. Through this continual planning

---

[2] CASPER is the soft real-time continuous planner extension to ASPEN.