2.4 Prototype Features

2.4.1 Phase 4 P & S Prototype

The phase 4 P & S prototyping focused on two main areas, critical design risks and usability. Two of the risk areas included the resource model process and its ability to handle the distribution of plan updates, and the problems that might occur going from design to development. The usability areas were related to user write access to portions of the schedule and the defining of constraints. The previous designs for constraints and accesses were deemed confusing for users and thus presented unusable so additional prototyping was deemed necessary.

2.4.1.2 Distributing Resource Model Data

Performance testing was performed during the phase 4 prototype due to concerns that design set in place for the resource model process was inadequate. In order to investigate possible bottlenecks in the system, three designs were prototyped/analyzed. The three designs included a single, central resource model process running at the EOC, a single master resource model at the EOC with slave resource models running at the ISTs, and multiple resource models running independently at both the EOC and ISTs. The results of the prototyping and analysis are described herein.

The performance of a single resource model running at the EOC was tested simulating 9 ISTs and 1 EOC machine all running connected to one resource model process. Initial scheduling incurred no performance problems. Subsequent scheduling caused the single resource model process to hang since it had to both perform scheduling of the new activities and distribute the existing schedule out to client tools located at the IST sites. Although the single resource model would require no additional coding or design work, it was deemed unsuitable due to performance problems. In addition, a single resource model process would incur a single point of failure for the P&S suite of software.

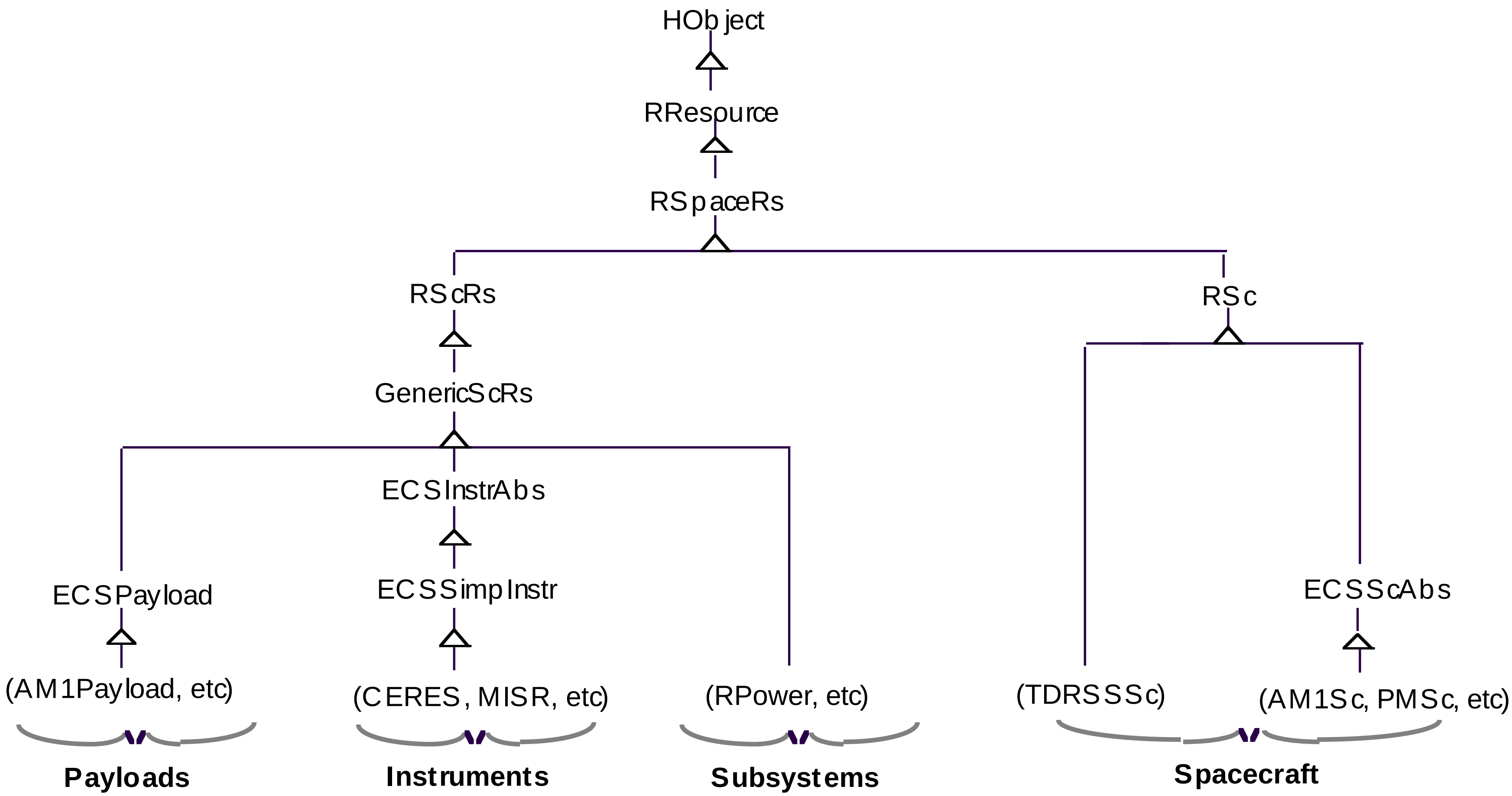

Figure 2-4. P & S Object Oriented Spacecraft Representation

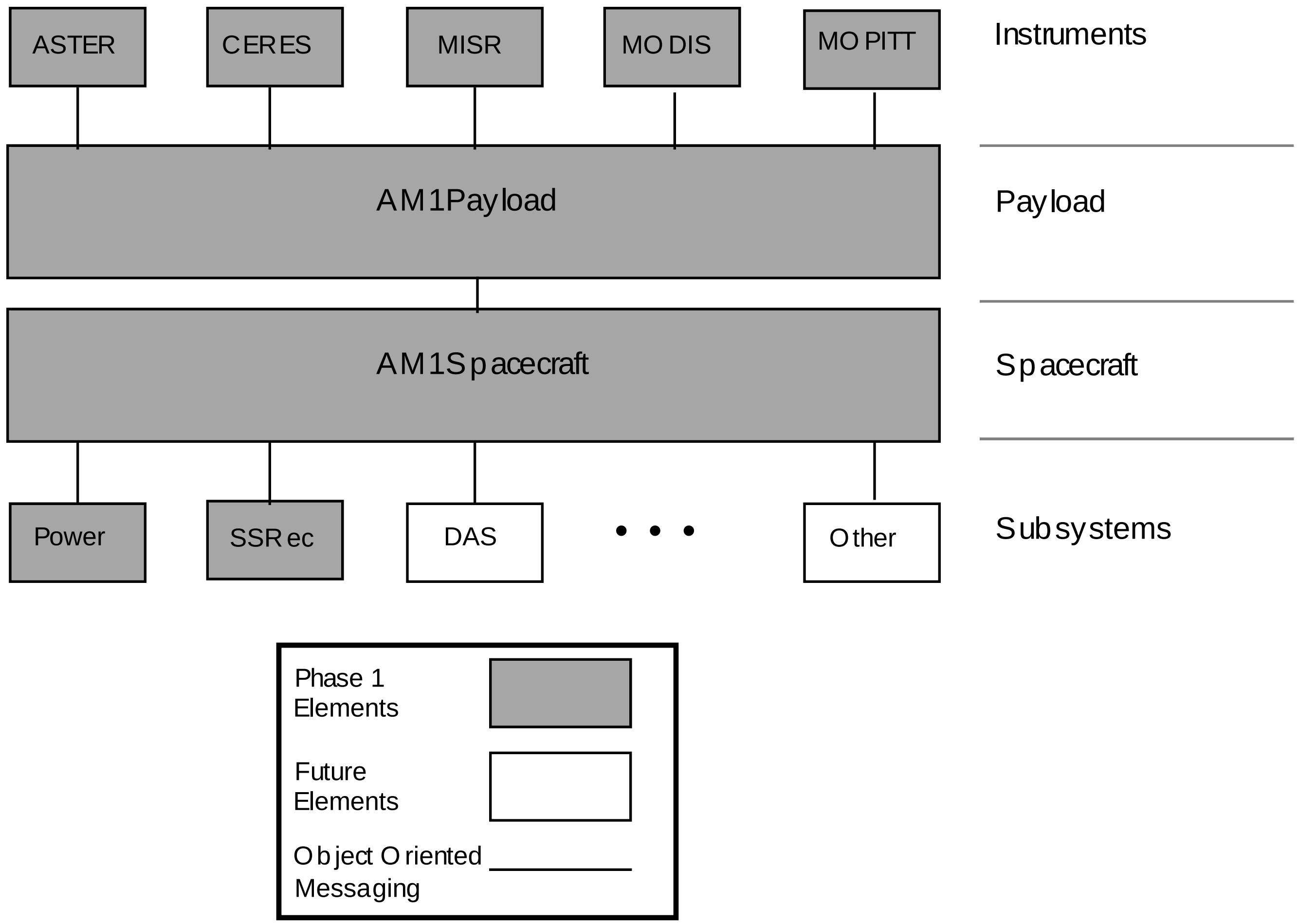

Figure 2-5. AM-1 Spacecraft Representation

The second design analyzed consisted of again a single resource model acting as a master to slave resource model processes running out at the ISTs. This design manifested the same problems as the single resource model process in that the master resource model posed as a single point of failure and the its distribution duties were only lessened by a small factor.

The third and final design introduced a new process called the data distributor. The data distributor process was made to be run as a companion process to the resource model and to perform the function of distributing schedule changes to other resource models that happen to be running at the time.

Figure 2-6 demonstrates the various stages involved in distributing a message through the data distributor network. The large circles on the diagram represent data distributor groups, each group being a set of distributor processes, usually within a single geographic location. Within the groups, smaller circles represent the data distributors themselves, and the number within each circle indicates the data distributor's priority within its group. A line represents a link between a data distributor and its resource model. Arrows indicate the flow of a distribution message from one process to another, and the number on each arrow identifies which stage corresponds to that distribution message being sent. The overall context of this diagram is that of a distribution message originating at a resource model and propagating out to all other resource models within the distribution network.

Stage 1: The SRM sends a scheduling message to its data distributor; the scheduling message gets packaged in a distribution message,

Stage 2: The data distributor sends a copy of the distribution message to each of the other data distributors within its group (group 3).

Stage 3: The original data distributor (priority 5, group 3) sends a copy of the distribution message to the highest priority running data distributor within each group external to group 3 (priority 3, group 1 and priority 6, group 2). The other data distributors within group 3 (priority 1 and priority 4) forward the distribution model to their resource models.

Stage 4: The data distributors which have just received a message (priority 3, group 1 and priority 6, group 2) immediately forward a copy of the distribution message to their resource models.

Stage 5: The same data distributors send a copy of the distribution message to each of the other data distributors within their groups.

Stage 6: Each distributor that has just received a message forwards that message to its resource model.

2.4.1.2 Constraints

The Planning and Scheduling subsystem checks three primary types of constraints: instrument and spacecraft subsystem mode transitions, algorithmic constraints(e.g. shared resource consumption, slew limitations), and user-defined temporal constraint(e.g. activity A not during activity B). Delphi (a set of class libraries upon which the current prototype is based) provides resource modeling and allocation, as well as hooks for checking algorithmic constraints. Based upon what Delphi provides, the phase 4 prototype focused on evaluating options for providing user-defined, database-driven constraint checking. Three different solutions for defining and evaluating user-defined constraints were considered: the Planning and Resource Reasoning (PARR) libraries developed by Goddard; a COTS set of class libraries (consisting of classes for finite capacity scheduling and resource allocation) developed by ILOG; and a custom solution building on the constraint checking already provided by Delphi.

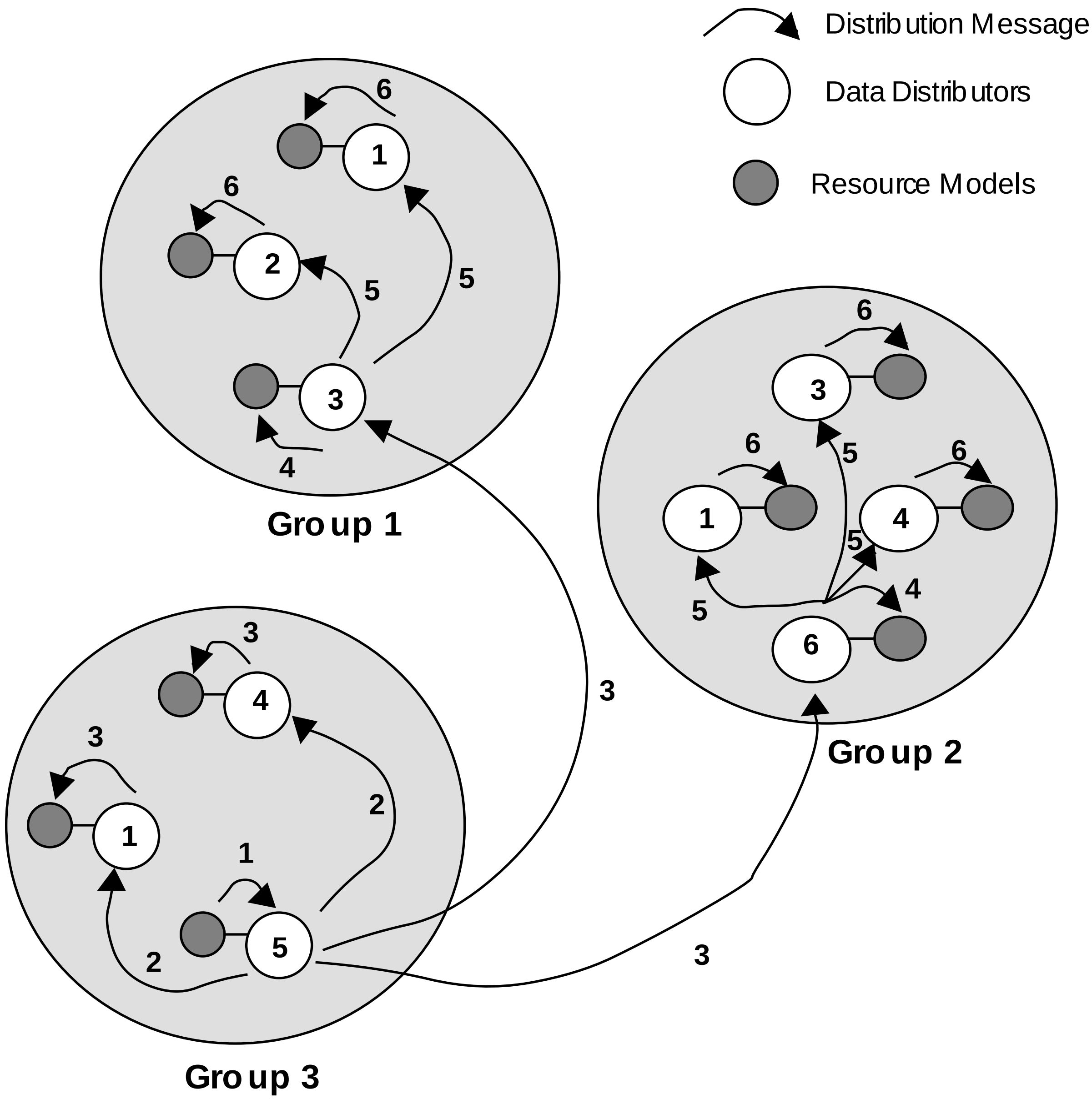

Figure 2-6. Data Distributor Network

The PARR libraries from Goddard provide the ability to define activities, constraints for those activities and scheduling strategies to be used in scheduling and rescheduling those activities. The Delphi class libraries provide activities as well and a method of scheduling those activities. Unfortunately, the incompatibilities between the two types of activities, coupled with differences in collection classes used by the two libraries forces extensive rework of either class library in order to integrate the two libraries together. Therefore, due to the cost of integrating the two libraries, PARR was eliminated as a viable option. Evaluating the PARR libraries, however, provided a great deal of insight into the types of constraints that the system would need to model.

The second class library evaluated for constraint checking was a library built by ILOG. Although ILOG's class library provides strong resource optimization and rescheduling strategies, the classes overlapped functionality already provided by the Delphi class libraries. The overlap in functionality would make integration of the two class libraries difficult and would produce either duplicate information (increasing memory requirements) or extra function calls in order to share information between similar, but different, classes (reducing run-time efficiency). Furthermore, the cost of the ILOG libraries coupled with the cost of training developers using ILOG's libraries makes ILOG's solution prohibitive.

Finally, as an alternative to integrating a COTS product into the current design, a custom solution using the strengths of Delphi was investigated. By adding no more than ten lines of code, the current prototype was extended to handle two types of user-defined constraints (A during B, and A not during B). This quick prototype revealed that, in general, developing the types of rules needed to define is more difficult than writing the code necessary to implement those types of rules. As mentioned earlier, PARR provided great insight into the types of rules that the system would need to support and helped define the final design of the system. Based on this prototype and the study of PARR, an estimate of only three to five hundred additional lines of code will be needed in order to implement the types of rules supported by PARR.

2.4.1.3 Timeline Access Display

In addition to the existing pull down menus available at the top of the timeline display, there will also be a menu for toggling to the timeline display that shows accesses. Accesses provide a method of reserving a portion of a resource on which to perform scheduling. This will prevent two or more users who have write permission on a resource from simultaneously scheduling over the same portion of the resource.

After the user brings up the access display, a timeline showing the same resources is shown with user accesses displayed. To the left of each resource is a check box that will toggle accesses on and off. This will provide a simple method for a user to remove their access when they have finished scheduling over their resource(s).

The timeline tool is used to display accesses because this gives the user an easy interface on which to view accesses. Since accesses have an associated resource and start and stop time, they can be displayed similarly to scheduled activities. Also, because the timeline tool already exists for displaying activities, this tool was easily modifiable to display user accesses.

2.4.1.4 Design To Development

The transition process from design to development was investigated by the P & S team. The purpose of this investigation is to make the development towards the final release package more efficient. This is accomplished by looking for process improvements in each phase of full-scale development.

The approach used was to take an actual design and start developing the application by following design specification. The first step of the procedure is for the developer to have a basic understanding of how the application works. Once the developer understands the functionality of the application, coding can begin. The developer would first generate skeleton templates from the Stp/OMT CASE tool. These templates are code blocks that are set up to meet the FOS coding style guideline. Using skeleton templates, the developer can integrate comments from the data dictionary. To code specific functionality of the application, the developer would translate the program description language (PDL) into C++ code. At the same time, fine-tuning of the prototype design is necessary because it is likely that there are a few minor discrepancies in the design. The next step is to use screen mock-ups as guides to develop application screens. At this point, the developed application is ready for debugging and testing.

The result of this investigation will help all future developments because improvements can be made at different points during the transition from design to development. The most important discovery is that the developer needs to prioritize coding. This means classes need to be coded in a specific order to eliminate the paralyzing effect of code interdependency. By coding classes with common functionality first, these classes will be ready for use when needed thus improving development efficiency.

2.4.2 Prototype Displays and Features

2.4.2.1 Activity Definer

The activity definer provides the user with a graphical interface for creating and modifying activity definitions. Figure 2-7 shows an example of the activity definer display.

In order to create or modify an activity definition, the user must first select the instrument or subsystem for which they wish to create and modify definitions. The list of pre-existing activity definitions appear in a scrolled list. If the user wishes to modify previously defined activities, they simply click on the activity they are interested in and the definition's information appears in the display. The user may modify the parameters and save the changes to the project database by hitting the Save button. The changes may be discarded by hitting the Reset button.

If the user wishes to create a new activity, they may do so by one of two ways. First, they can select a previously defined activity and change the name field to the name of the new activity. Secondly, they may create a new activity from scratch. The user can modify any of the definition's parameters or its command list. The command list is modified by hitting the Commands button, which brings up a command list editing window. In this window, the user can add and delete commands in the activity, set the times the commands will execute (relative to the start or stop time of the activity), and change the default values of the command parameters. User modifications can be saved or discarded by hitting the Save or Reset buttons, respectively.

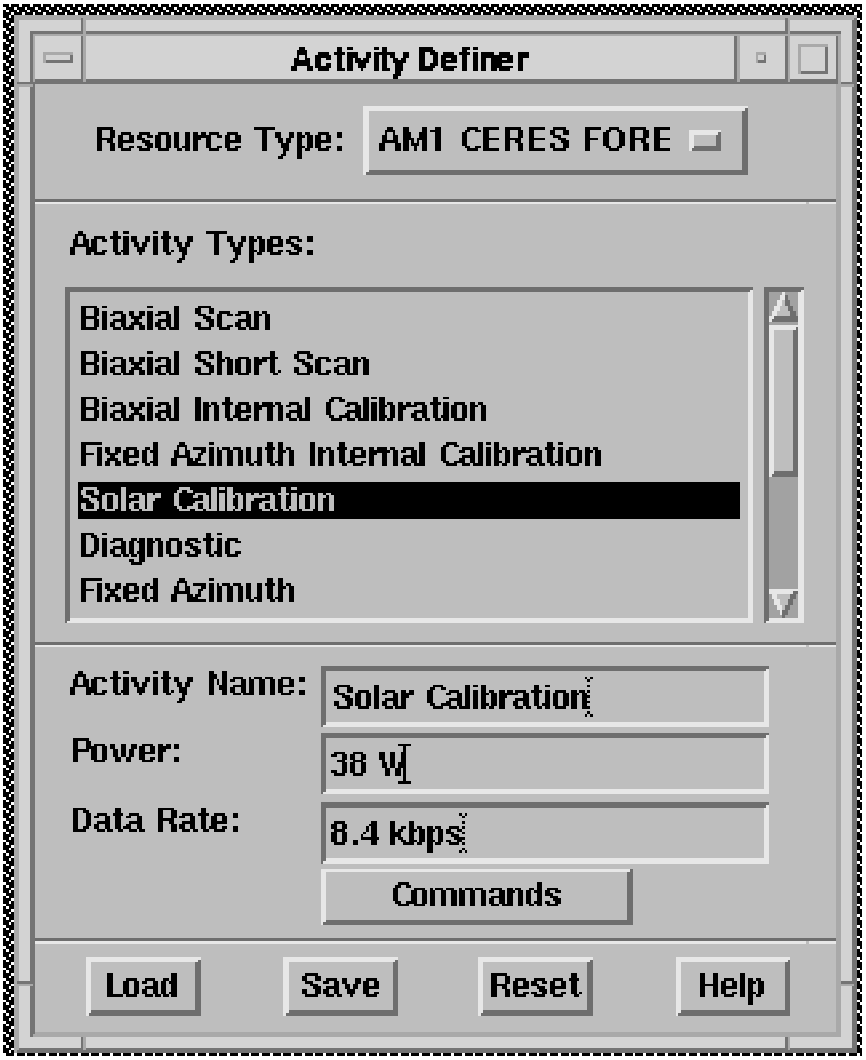

Figure 2-7. Activity Definer

2.4.2.2 BAP Definer

Figure 2-12. Activity Definer

The BAP definer provides the user with a graphical interface for creating and modifying Baseline Activity Profile definitions. Figure 2-8 shows an example of the BAP definer display.

The user selects from a list of instruments and subsystems in order to create or modify a BAP for that resource. The activities for that resource appear in a scrolled selection list called the Activity Types list. The user may add activities to the BAP by selecting an activity in the

Activity Types list and hitting the Add button. When an activity is added to the BAP, it appears in the BAP Activity list. The user may delete activities from the BAP Activity list by selecting the activity in the list and hitting the Delete button. The user may also edit parameters for the activity and provide relative start and stop time for that activity by hitting the Edit button.

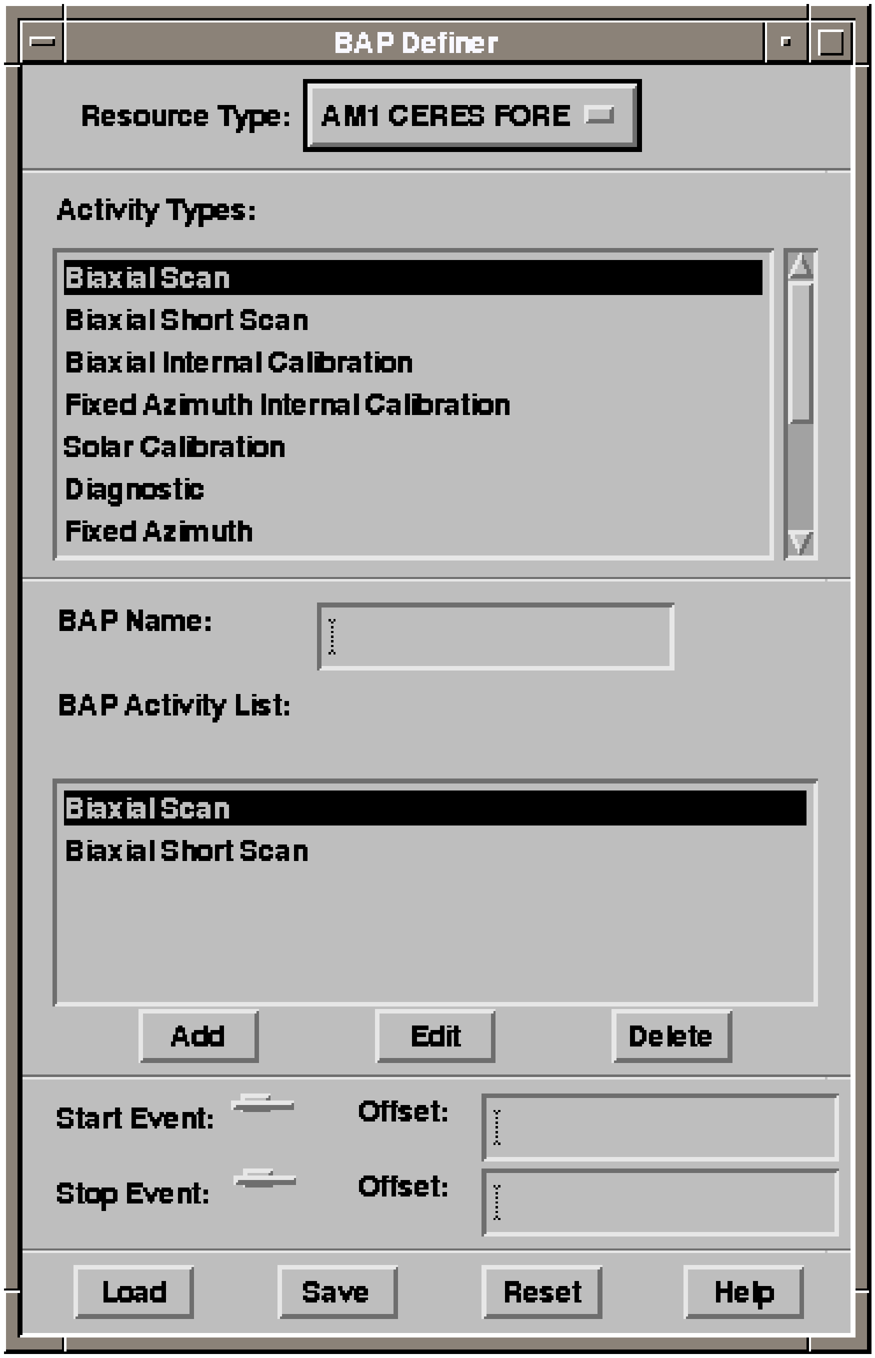

Figure 2-8. Baseline Activity Profile Definer

The user must also specify the name they wish to assign to this BAP definition. The user's changes may be saved by hitting the Save button. The changes may be discarded by hitting the Reset button.

2.4.2.3 Timeline Display

The timeline process presents a graphical color representation of instrument and subsystem activities as a function of time. Figure 2-9 gives an example of the timeline display. A time axis appears on the top of the display showing the current time range. A scroll bar appears at the bottom of the display that allows the user to scroll back and forth through the current time window. To scroll through time, the user holds down the leftmost mouse button and drags the scroll bar either right or left, depending on the range of time that the user wishes to view. As the user drags the scroll bar, a popup text field gives the time range to which the display will adjust after the user releases the mouse button. The user can also click anywhere in the scroll bar trough and the time range will be adjusted by a default amount of time.

To adjust the range of the current time window, the user can use the menu entitled "Time" and choose the option "Change Plan Window". This action will present the user with a display that will allow the user to enter start and stop times to change the timeline window boundaries. The user can then choose one of the option buttons to either confirm, apply or cancel the time range change.

To give the user further control of display views, the timeline also gives a method of limiting the time range that is currently displayed. The "Show" option menu in the lower right-hand corner of the timeline controls the range of time that can currently be displayed. This will allow users to zoom in on a particular region of interest. The user can click on the menu and choose the length of time to be displayed.

The left hand side of the timeline process shows the various resources that can be scheduled. The data in the center represent the resource states. The rectangular boxes are scheduled activities, each with the name of the activity as the label. The timeline process displays an asterisk when an activity label is too small to appear. Activity types can be represented with a specific color. This aides the planner and scheduler by providing easy recognition. A description of the different types of resources are shown below:

> **Instruments—**All of AM-1's instruments are represented on the prototype timeline display (refer to Figure 2-9). A simple list of activities were developed based upon the instruments' modes of operation. For activities that have a data rate and consume power, scheduling will impact the associated SSR buffer and spacecraft power subsystem.
>
> **Solid State Recorder Data Buffers—**These resources represent data storage areas for instrument science and housekeeping data. The dashed line corresponds to the data storage limit that is available in the buffer, while the two-dimensional line plot shows the data volume as a function of time. When the two-dimensional line plot exceeds the dashed line, the plot color turns red, indicating that data will either be overwritten or lost. Whenever a user schedules an instrument activity which consumes buffer space, the amount of consumption is reflected on the corresponding buffer. When a user schedules

a TDRSS contact (also known as a TDRS playback request), the data volume plot reflects the contact use by sloping downward indicating that data is being downlinked and thus space is made available in the buffer for future data collection.

AM1 Power—This resource corresponds to the power subsystem on the AM-1 spacecraft. The AM-1 power subsystem permits multiple resources to utilize its power output. The dashed line corresponds to the power consumption limit that is available for the entire spacecraft while a histogram represents the power that is currently being consumed. When the histogram exceeds the dashed line, it turns red, indicating a power consumption constraint. Whenever a user schedules an instrument activity which consumes power, the amount of power consumption is appended to the power histogram.

TDRS-1—This resource represents an example of a Tracking and Data Relay Satellite that will be used to downlink data from the EOS spacecraft. Shown on the timeline are activities which represent scheduling windows for TDRS that indicate when the EOS spacecraft is in view of the TDRS spacecraft. The scheduling windows show available periods for requesting TDRS contacts in which to uplink loads or downlink data. The scheduling periods are provided by the Flight Dynamics Facility (FDF).

AM1 HGA—This resource corresponds to activities related to the EOS spacecraft's high gain antenna. In the current planning and scheduling prototype, this resource only shows contact activities between AM1 and TDRS. The contact activities consist of transfer of data to TDRS (or uplink of loads from TDRS to AM1) and only occur during a TDRS scheduling window described previously.

AM1—This resource displays orbit event information such as sunrise and sunset times. This will allow a flight operations planner to schedule instrument and spacecraft subsystems based on spacecraft activities. For example, the CERES scheduler will schedule the CERES biaxial short scan activities at each sunrise and sunset time. Event information is usually provided by the FDF.

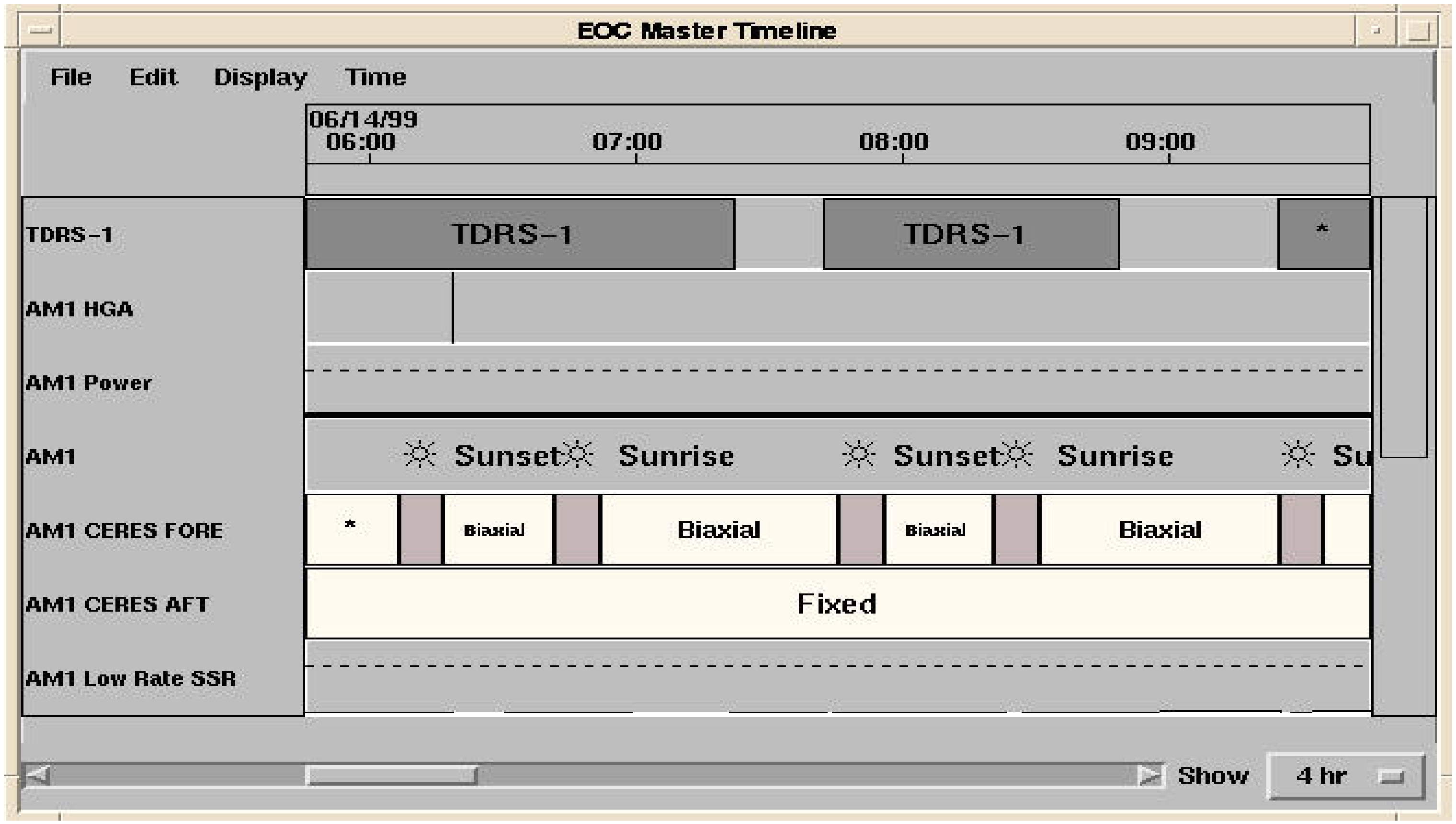

Figure 2-9. Timeline Display

Another feature of the timeline allows the user to change the display configuration. A menu in the upper left-hand corner titled "Display" contains two options, one to change colors and another to change resources. When the user chooses the "Change colors" option, a color palette window is presented to the user along with a scrollable list of activity names. The user may click on an activity name and then choose a color. As soon as the user clicks the confirm or apply button the chosen type of activity will appear in the chosen color on the timeline resource line. Activities of the same type that are scheduled later will also show up in the chosen color. The "Change resource" option allows the user to selectively choose the resources that they would like to view on the timeline. If the user selects this option, a window is displayed that allows the user to add or remove resources to be viewed from the list of available resources. After the user clicks the confirm or apply button, the timeline is re displayed showing the selected resources.

The timeline also has a file menu in the left-hand corner that contains options for the user to either load a timeline configuration, save a configuration or get a hard copy of the timeline. If the user selects the save configuration option, a file dialog panel is displayed that allows the user to save the current configuration of the timeline. The timeline configuration consists of the colors and resources that are displayed to the user and that are dynamically modified by the submenus of the Display options described previously. The load configuration option allows the user to load a previously saved configuration. Note that the definition of a configuration in this context only contains the types of resources and activity colors, not the actual activities and their occurrences on the timeline.

Another pulldown menu at the top of the timeline allows the user to toggle to the access timeline display. This will display the same set of resources but user accesses will be displayed instead of instrument and subsystem activities. Check boxes to the left will allow the user to toggle the accesses on and off.

2.4.2.4 Segmented Timeline Display

The segmented timeline process presents a graphical color representation of instrument and subsystem activities as a function of time in a similar fashion to the regular timeline. Figure 2-10 gives an example of the segmented timeline display. A time axis appears on the top of the display showing the current time range. A rectangular scroll bar appears at the bottom of the display that allows the user to scroll back and forth through the current time window. By sliding the scroll bar back and forth with the mouse, the user can change the time period of the displayed activities. In addition, a user can shrink or enlarge the scroll bar, effectively zooming in and out of time.

The left hand side of the timeline display shows a set of resources for each segment. Each segment is a pre-determined length of time defined in the segment timeline database. This feature allows the user to view multiple segments on one display in order to give him the capability to see repetitive cycles of scheduled activities. Activities are displayed as rectangular boxes similar to the regular timeline.

Other features such as displayed resources, configuration changes and loading and saving configurations are similar to the capabilities of the regular timeline that was described in section 2.4.2.3

2.4.2.5 Instrument Activity Scheduler

The instrument activity scheduler is a process that schedules individual activities or lists of activities, called baseline activity profiles (BAPs), over a given time range. Figure 2-11 shows an example of the scheduling display. The user schedules an activity or a baseline activity profile by first selecting the resource on which to schedule the activity. Once the resource is selected, a list of activity types the user can schedule will appear. The user then selects one of the activities or one of the baseline activity profiles. A plan upon which to schedule must be selected and a start and stop time must be entered, if the default values are not desired. Once all of the above has been performed, the user hits the Schedule button to schedule the activity or the baseline activity profile.

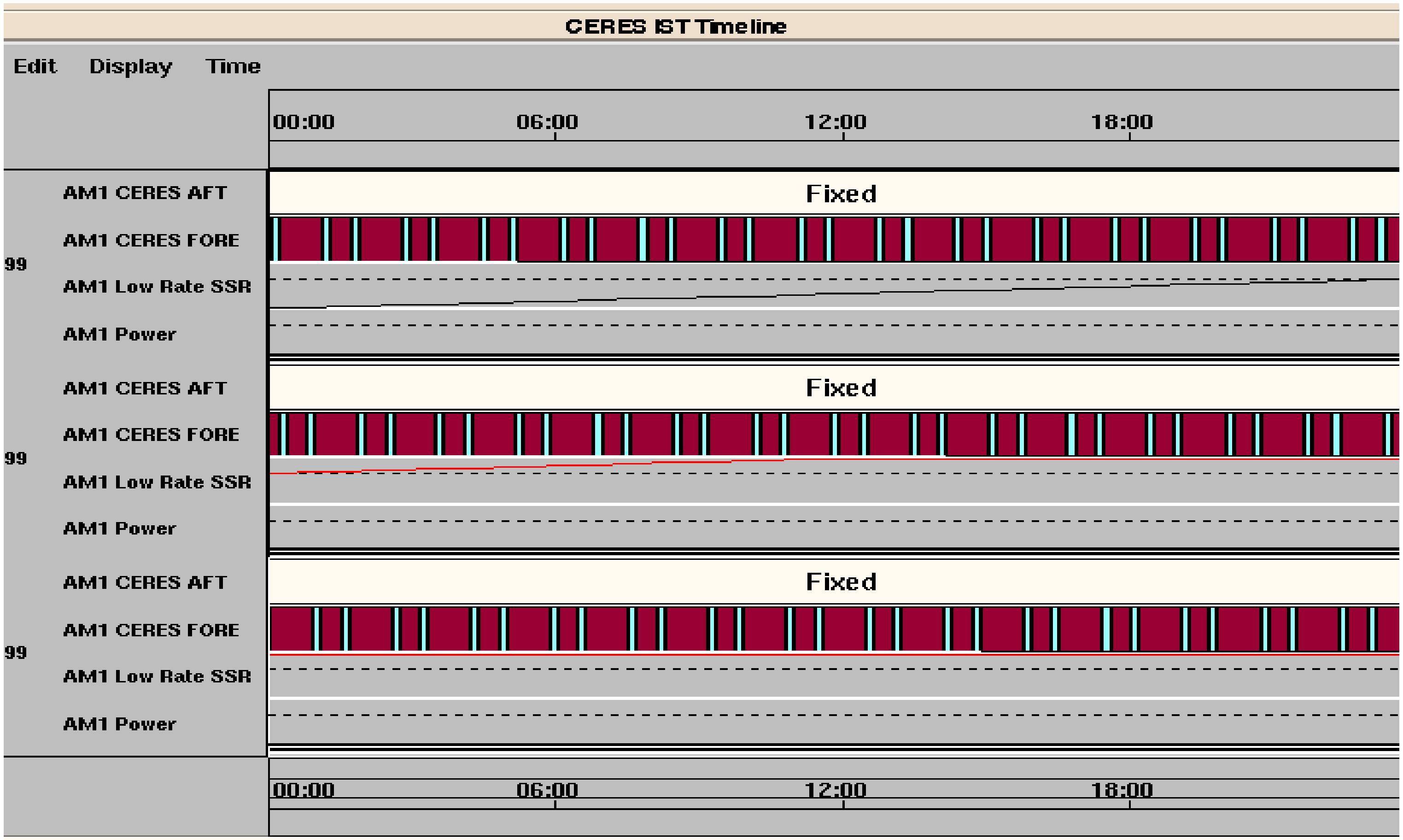

Figure 2-10. Segmented Timeline Display

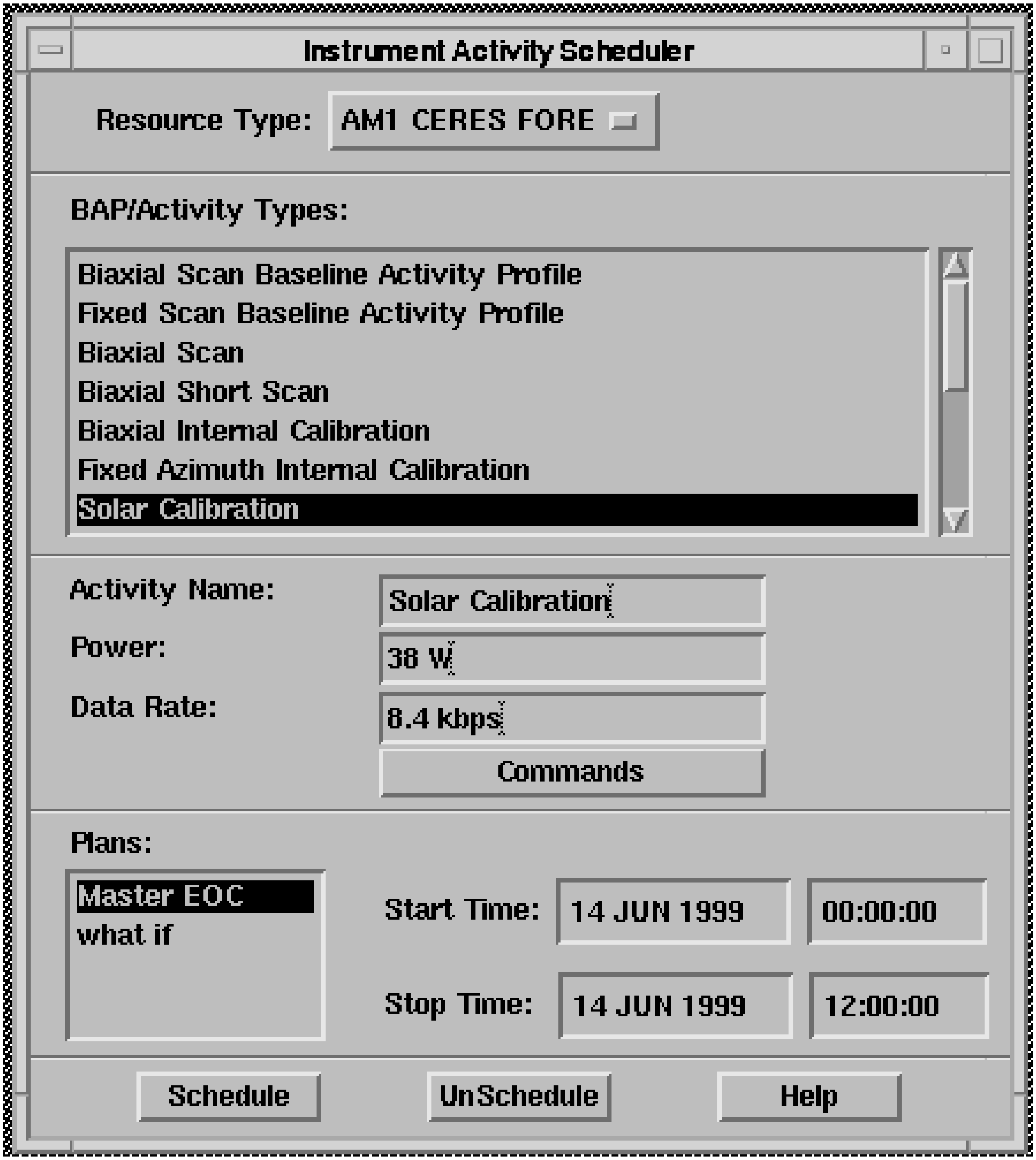

Figure 2-11. Instrument Activity Scheduler

Figur

2.4.2.6 Communication Contact Scheduler

The communication contact scheduler is a tool which allows the user to schedule requests for TDRSS and ground contacts. This phase of the prototype does not provide a mechanism to process the responses from the NCC concerning those requests.

Figure 2-12 shows an example of the Communication Contact Scheduler display. The user provides the following information: the plan they wish the contact requests to be scheduled on and the start and stop times for which contact times are needed. When the user hits the Schedule button, the scheduler determines TDRSS availability based on the times that the TDRSS is in view of the spacecraft and schedules contact requests over the given time range. By hitting the Unschedule button, the user may remove contacts over a given time range on the given plan.

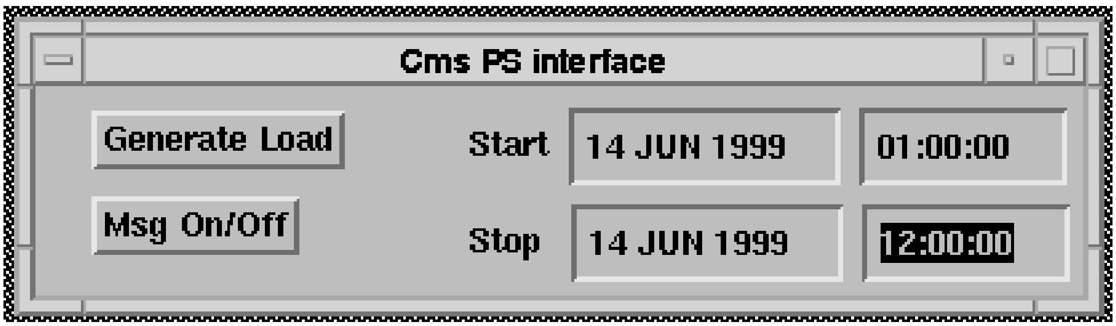

Figure 2-12. Command Management Tool

2.4.2.7 Plan Tool

The plan tool interface is used to create, delete, and copy plans. It also has a function to snap to a plan. Figure 2-13 shows an example of the plan tool display.

To create a new plan, the user hits the Create button and a window is displayed that prompts for a plan name. The user enters the plan name and then hits the Apply button. The new plan name appears in the plan list on the main window.

Another function allows the user to delete a plan. The user selects the plan from the list and then the user hits the Delete button. When a plan is deleted, all activities and associated resources on that plan are also deleted. The plan name that the user selected is removed from the list indicating a successful plan deletion.

The third function offered by the plan tool allows the user to copy one plan to another. The user first selects the plan to be copied from the plan list and hits the Copy button. A window requesting information about the destination plan is presented. The user then enters the destination plan name and (optional) start and end times for the portion to be copied. If no start and end times are specified, the entire source plan is copied to the destination plan. If the destination plan already exists, the source plan overwrites the destination plan. After the user enters the destination information, the Apply button is hit. The destination plan name is added to the plan name list in the main window.

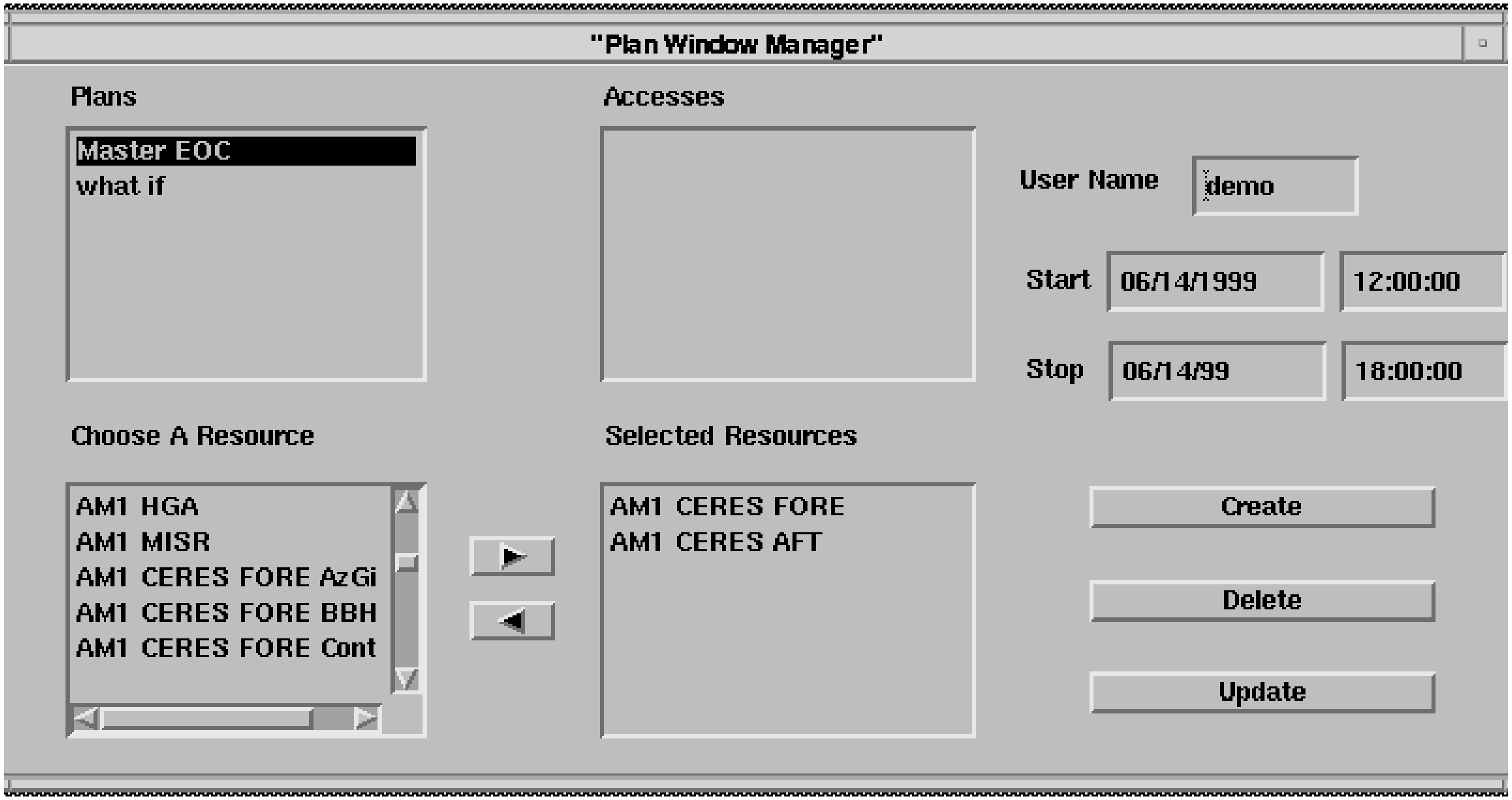

Figure 2-13. Plan Window Manager

Another function provided by the plan tool lets the user "snap" to a plan. "Snapping" to a plan allows the user to broadcast the plan to any tool that requires knowledge of a current plan such as a timeline. When the user "snaps" to a plan, visualization tools, such as a timeline, will immediately display the activities and resources associated with the "snapped" plan. This allows all tools that have visibility into a plan to display views of the same plan. The user clicks the snap button on the main plan tool display and a window appears, prompting the user for a time range in which to snap. If no time range is selected, the snap will cover the entire plan. The user then clicks the apply button to initiate the snap and all tools having visibility into the snapped plan will change to that plan.

2.4.2.8 Plan Window Manager Display

The plan window manager display tool is a graphical user interface that manages user access to plan resources. User accesses are required if a user wishes to update a plan (i.e. schedule activities on a plan). This will prevent users from attempting to update the same portion of a plan at the same time. Figure 2-14 shows an example of the plan window manager display.

The window contains areas that show a plan list for which accesses may be created, a resources list, an accesses list and a selected resources list. To create an access, the user first selects a plan from the plan list. Then the user selects resources from the resource list and clicks the add arrow to add resources to the selected resources list. They then enter the start and stop times of the portion of time on which they wish to make modifications to the plan. Finally, the user hits the Create button and the new access is added to the accesses list. By choosing an existing access from the access list, the user may call up the information about the selected access (i.e. the resources and time span). Then the user may make modifications to the resources and timespan (by either changing the time span or selecting different resources) and hit the Update button to make an update to an access. Also, a user may delete the access by choosing an existing access and hitting the Delete button.

2.4.2.9 CMS Interface

The command management interface is a graphical user interface for releasing the detailed activity schedule to Command Management. The interface allows the user to enter a time period for the detailed activity schedule and press a Generate Load button with the mouse. When this button is pressed the detailed activity schedule is passed on to command management which generates an ATC load and a ground script. Figure 2-15 shows an example of the command management tool.

2.4.3 Multi-Platform Capability

During the phase 3 prototype, all P & S code was ported to four major platforms including Sun, DEC, SGI and HP. Because the P & S software is built upon the existing Hughes Class Library foundation, no ECS specific code needed modification in order to run on any of the other platforms. For the phase 3 prototype scenario, all processes were executing on Sun workstations, but when operational, the P & S system will function in a heterogeneous computer environment. Different platforms and operating systems may comprise the P & S infrastructure. In order to establish flexibility, the ECS P & S software must be maintained for different platforms and operating systems. In order to accomplish this, nightly builds are run on each of the major platforms to catch any platform dependencies immediately.

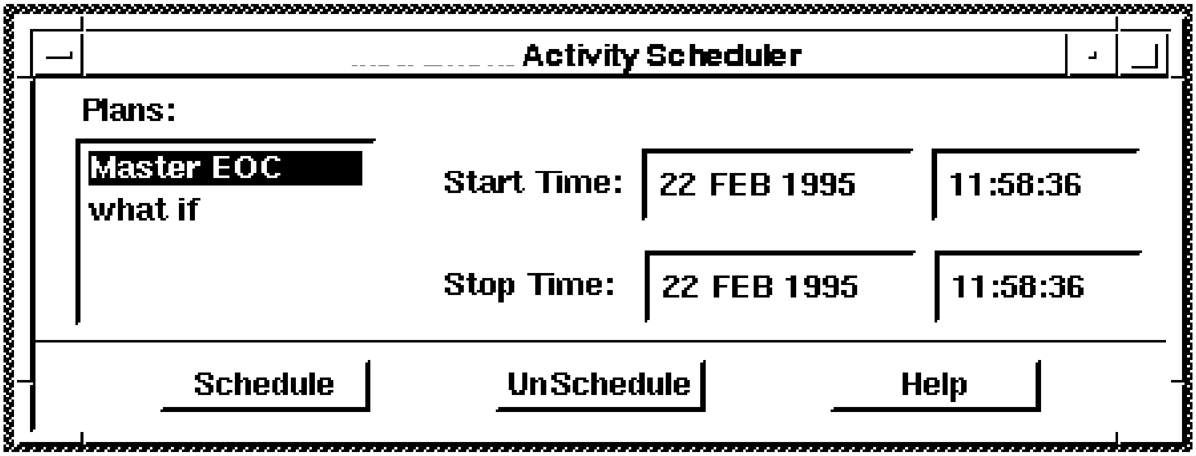

Figure 2-14. Communication Contact Scheduler

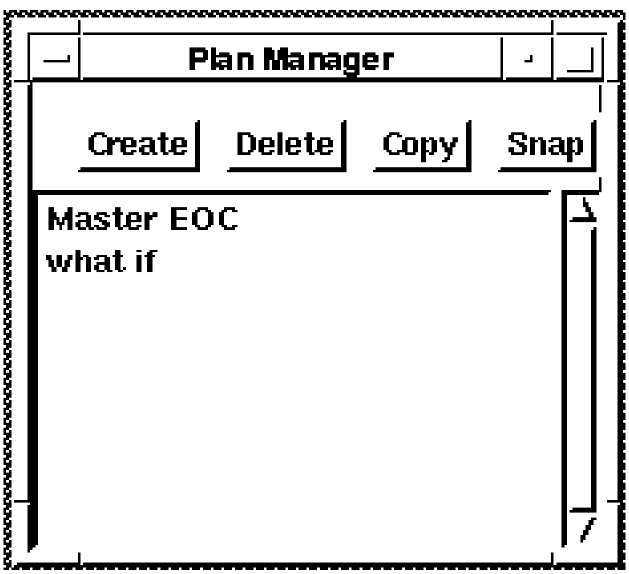

Figure 2-15. Plan Tool