MPS Editor – An Integrated Sequencing Environment

Barbara A. Streiffert[1] and Taifun O'Reilly[2]
Jet Propulsion Laboratory/California Institute of Technology, Pasadena, CA. 91109

Mitchell Schrock[3]
Jet Propulsion Laboratory/California Institute of Technology, Pasadena, CA. 91109

and

Jaime Catchen[4]
Jet Propulsion Laboratory/California Institute of Technology, Pasadena, CA. 91109

In today's operations environment, the teams are smaller and need to be more efficient while still ensuring the safety and success of the mission. In addition, teams often begin working on a mission in its early development phases and continue on the team through actual operations. For these reasons the operations teams want to be presented with a software environment that integrates multiple needed software applications as well as providing them with context sensitive editing support for entering commands and sequences of commands. At Jet Propulsion Laboratory, the Multi-Mission Planning and Sequencing (MPS) Editor provided by the Multi-Mission Ground Systems and Services (MGSS) supports those operational needs.

I. Introduction

COMMERCIAL software applications have moved in the direction of presenting users with a unified environment to make it easier for them to perform their work. Even though spacecraft have become more complex and operations teams have become smaller, the software applications that the teams use are often not unified. In the past, users have built connecting "glue-ware" scripts to unify the applications. At Jet Propulsion Laboratory the Mission Planning and Sequencing (MPS) element of the Multi-Mission Ground Systems and Services (MGSS) Program has built a software editor that provides the operations teams with a unified environment. In this environment the operations teams can define spacecraft activities, adapt/create the software models for the spacecraft, create/modify sequences of spacecraft activities and commands and check the sequences for constraint violations using an event simulator named Seqgen. MPS Editor allows the operations teams to be more productive by providing them with a tool that supports both development and operations.

II. Overview

MPS Editor is written in Java using the Eclipse Rich Client Platform. Eclipse is an open source software framework for building stand-alone applications or creating plug-ins that can be used by other applications. MPS Editor has four perspectives. A perspective is a grouping of task functions along with their graphical user interface elements. The four perspectives are the Activity Dictionary Perspective, the Spacecraft Model Adaptation Perspective, the Sequence Development Perspective and the Spacecraft Modeling Perspective. The Activity Dictionary Perspective defines the reusable activities that are available to be used for a given mission. The Spacecraft Model Perspective supports the development of models of spacecraft hardware that are to be used by the

[1] Senior Software Systems Engineer, Section 317, 4800 Oak Grove Dr. 301/250D, Pasadena, CA. 91109.
[2] Senior Software Engineer, Section 317, 4800 Oak Grove Dr. 301/250D, Pasadena, CA. 91109.
[3] Software Engineer, Section 317, 4800 Oak Grove Dr. 301/250D, Pasadena, CA. 91109.
[4] Systems Engineer, Section 317, 4800 Oak Grove Dr. 301/250D, Pasadena, CA. 91109.

JPL discrete event simulator named Seqgen. These models contain state, consumption and constraint information used to adapt Seqgen to specific missions. The next perspective is the Sequence Development Perspective that provides functionality for creating sequences of activities and commands to be run on the spacecraft. The final perspective, the Sequence Modeling Perspective, allows users to run Seqgen using the sequences, models and activities created in the other perspectives. MPS Editor has a configuration file that identifies the spacecraft, the perspective to be shown at startup and the perspectives to be included by the project. If a project decides that it does not need one of the perspectives, the perspective can be eliminated using this file. Projects can also use different configuration files for different users. For example, a project may want all team members to see the Sequence Building Perspective but only a few team members see additional perspectives based on their role on the project. MPS Editor Display areas are shown in Figure 1. MPS Editor with three of the four perspectives populating the perspective tabs is shown in Figure 2. Also in Figure 2 the Activity Dictionary Perspective is active.

.

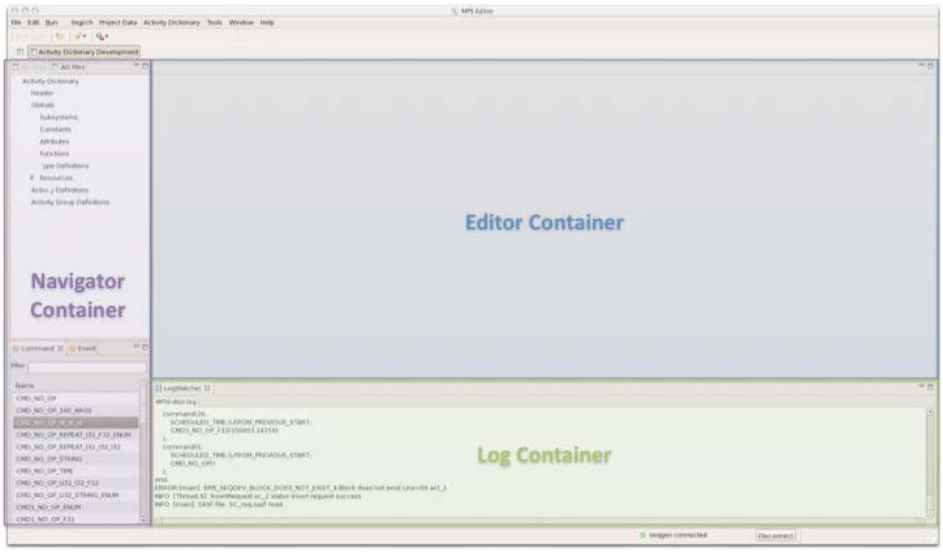

Figure 1. The MPS Editor display with various screen areas identified.

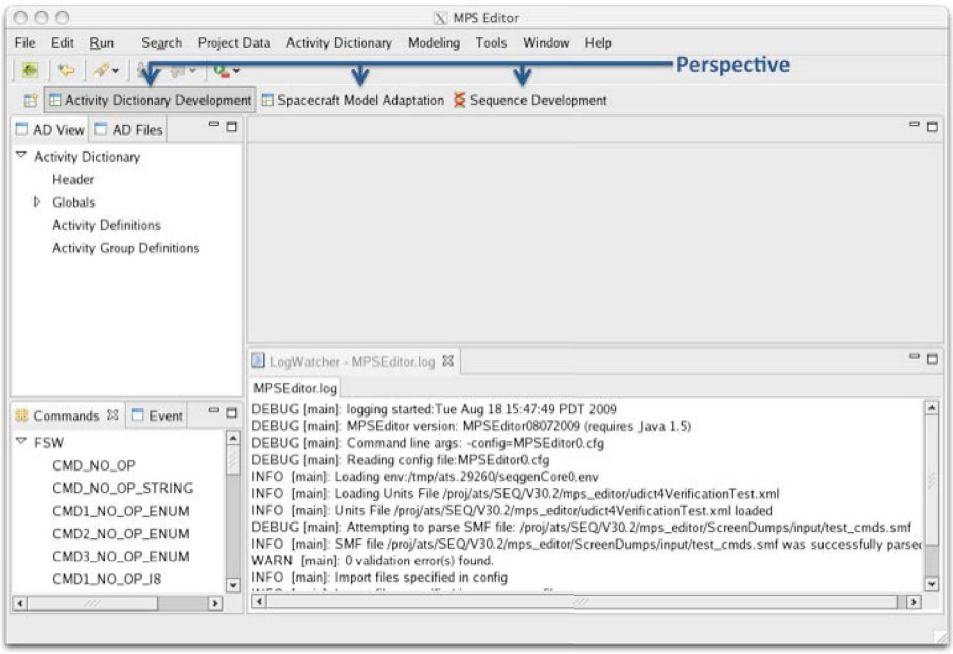

Figure 2. MPS Editor with multiple perspectives open

One major advantage of MPS Editor is that information is presented in a uniform way. The view area is in the same place in all perspectives. The navigator is also in the same place and so is the log information. Users can customize the various areas making them by smaller or larger or repositioning them. The mission or users can perform additional customization based on the configuration or the preference files. The configuration file provides information to Seqgen, the command simulator, such as identifying whether a user is accessing a local or remote version of the software. The preference file provides user preferences, such as whether to log information in a verbose mode or not. MPS Editor has the standard menu bar with entries such as File, Edit and Run as well as other entries that are specific for the certain perspectives. Users also have access to an application tool bar that allows them to perform a user search and view the last edit location along with other useful tasks. In addition, to being able to run MPS Editor as a standalone application, each perspective can be run as an Eclipse Plug-in. In fact, Mars Science Laboratory has incorporated the Sequence Development Perspective plug-in into their planning software. This mission has also used the Activity Dictionary Perspective to develop their activity dictionary.

III. Activity Dictionary Perspective

The Activity Dictionary Perspective is typically used during the development phase of a mission. However, it can be used to update or create new activity definitions as the mission matures and changes in the knowledge of the spacecraft that require the existing activities be updated. The activity dictionary contains observations, calibrations and other groups of spacecraft commands that perform tasks on the spacecraft that are repeated during the mission. Usually the activity definitions have parameters so that the activity definition becomes a generalization similar to software methods or functions. The activity dictionary defines the general case and operations teams use the general case with specific data for a particular sequence.

The activity dictionary is an eXtended Markup Language (XML) file that uses an XML schema when it is created. The XML schema is project specific. The project can design its own schema or use one of the two existing schemas. One of the schemas works with another MPS software element called APGEN (planning software). The other schema is more general and has been used by Mars Science Laboratory (MSL).

There are specific types of elements that are common to all activity dictionary entries. Each entry in the activity dictionary contains Attributes, Functions, Subsystems, Constants, Type Definitions, Resources, and Activity Definitions. Attributes are the parameterized elements for an activity, for example, the number of pictures to be taken for an observation. Functions are repeatable operations within the activity that are usable by several activity definitions, such as a series of turn commands for taking a picture. Subsystems represent the various hardware subsystems on a spacecraft such as the instrument subsystem or the attitude control subsystem. The subsystems can be used to determine operations team access control. Constants are numerical values that do not change. If the camera on the spacecraft frequently takes pictures in a specific way, the camera operations team may want to have an activity that uses constants instead of using a more general activity that requires parameters and uses additional sequencing space on the on-board computer. Type Definitions are a way of specifying specific types that are not part of the few general types that are available. Lists and structures are two list types that are typically defined. Resources are the consumables that are used. These consumables can be numeric values such as the amount of space on the recording device or they can be one-at-time use resources such as being able to point the spacecraft. They can also be restorable resources as the two previous examples or they can be the type of resource that has a measured lifetime such as the number of sample trays. Finally, there is the activity definition itself. The activity definition contains commands and control elements such as loops and conditional statements.

In addition to these fields the activity dictionary can contain global variables. The global variables are variables that can be used by any of the activity definitions within the activity dictionary. The global variables as well as the fields within an activity can be defined as mathematical expressions. Another feature of the Activity Dictionary Perspective is to allow users to define entrance and exit criteria for the activity definition. For example, if an instrument needs to be heated prior to using it for an observation, the activity definition can specify that the heated state is required. The same is true for exit criteria. If the activity changes the attitude of the spacecraft, but is required to return the spacecraft to its original attitude, it can be specified in the activity definition.

The Activity Dictionary Perspective allows the operations teams to build a dictionary from scratch or to read in the current Activity Dictionary and make changes to it. As with most editors if users make a syntax mistake, they are advised of the error. In many cases based on the XML schema, users are presented with pull-down menus to reduce the number of possible errors. Finally, if users want to look at the XML version of the activity dictionary instance, it is available to them within the editor as a menu selection. Figure 3 shows the areas of the Activity Dictionary

Perspective, and Figure 4 shows the Activity Dictionary Perspective with a sample activity dictionary instance loaded.

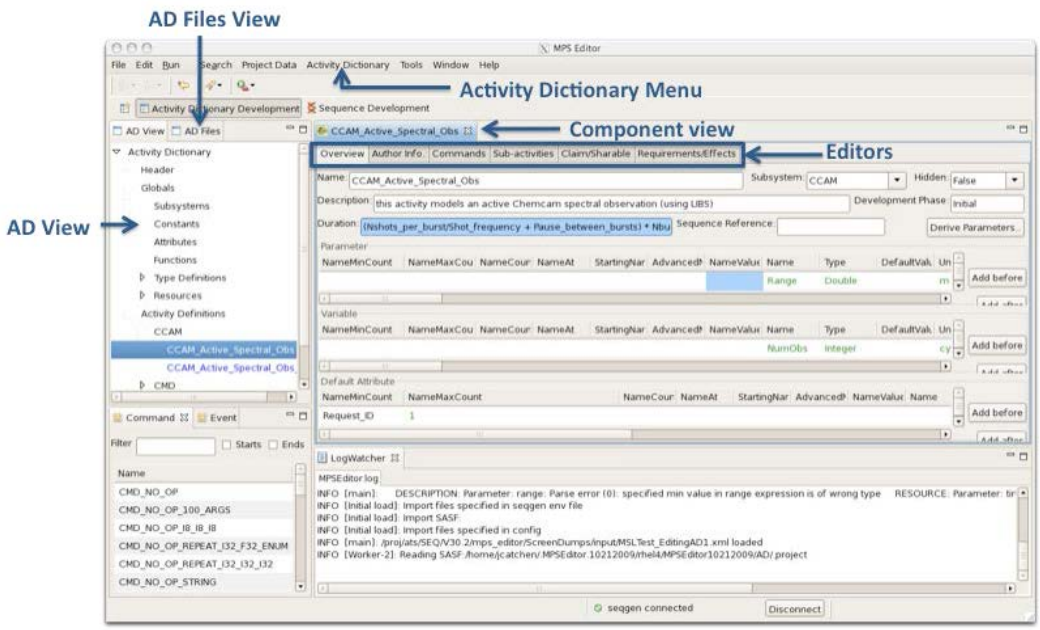

Figure 3. The Activity Dictionary Perspective with the perspective dispay areas identified.

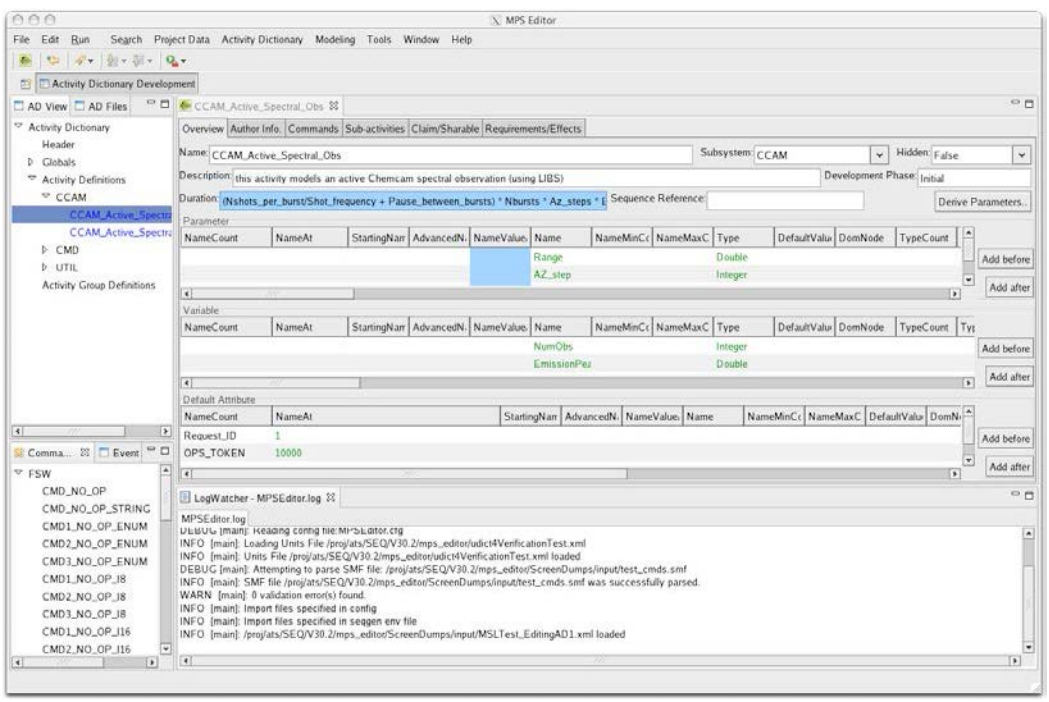

Figure 4. The Activity Dictionary Perspective display with a sample dictionary shown.

IV. Spacecraft Model Adaptation Perspective

The Spacecraft Model Adaptation Perspective is the perspective that allows the mission specific spacecraft models to be built. This perspective allows the development team to define the hardware states, constraints and rules within the model. It contains the commands and the impact that the commands have on the spacecraft state. It also

defines the relative timing of the commands by specifying any delays in triggering the command as well as the command duration. The models are then used in the MPS discrete event simulation software named Seqgen. The models are set up to activate/deactivate states of the hardware, add and subtract consumables and determine constraint violations. The models are triggered by the spacecraft commands. The spacecraft commands are imported from the command database using this perspective. Figure 5 shows the Spacecraft Model Adaptation Perspective with the different areas of the display identified.

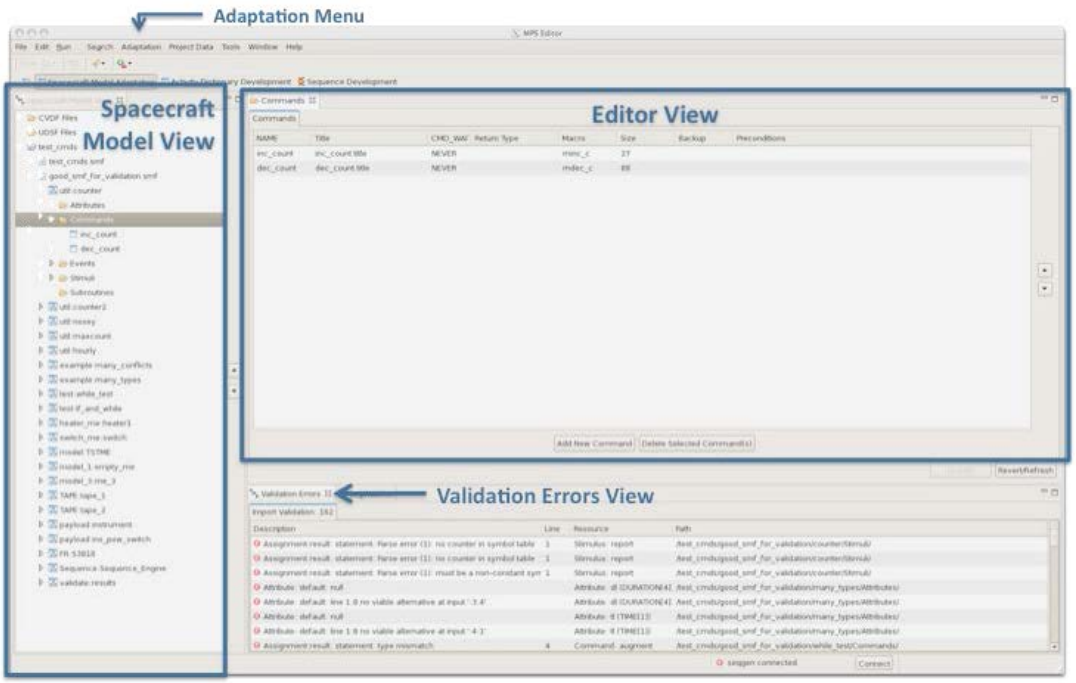

Figure 5. The Spacecraft Model Adaptation Perspective with the various work areas annotated.

The Spacecraft Model Adaptation Perspective allows the development team to create a new adaptation or modify an existing adaptation. On new adaptations the developer reads in the current command database and starts performing the adaptation after the commands have populated the command window. For existing adaptations that have a new command database, the developer can ask this perspective to do a merge of the commands. Figure 6 shows the Merge/Difference display. The software will identify all of the changes and the developer can address them one by one. If the command database has not changed, the developer can modify the models without this extra step. Since command databases change frequently the merge capability is a significant time saver.

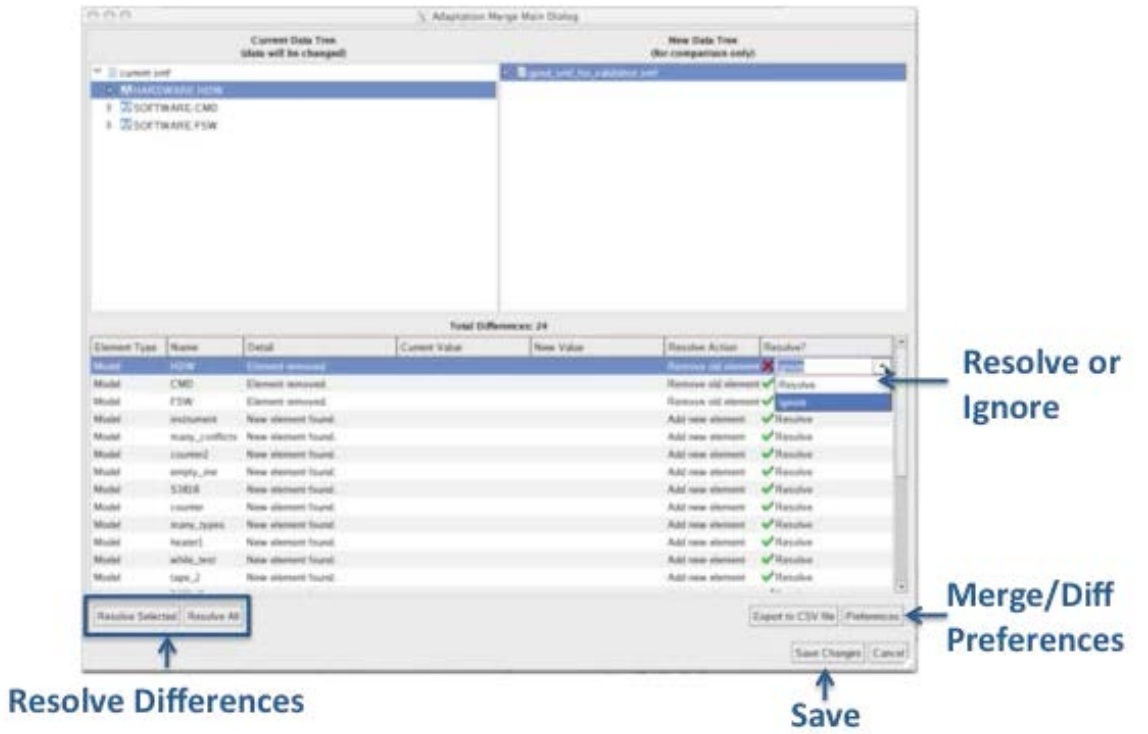

Figure 6. The Merge/Differences display shows the merge capability in the Spacecraft Model Perspective.

Another feature that saves time is the ability to compare models. This perspective allows the developer to make that comparison and to change one of the models based on the differences. It is an excellent feature that saves time and errors since much of the work has been done automatically for the developer.

Often, for convenience, models are grouped in different files or different tables of a database. The models have three tiers – model group set, model group and model element. Typically, operations teams prefer to keep subsystem models together and this hierarchy allows the teams to keep them separate. In these models the developer also sets timing including when the command is to start and when it is to end. Generally, the timing is set to a relative time. If the adaptation language is not rich enough for the model behavior, the developer can create a user defined function to model the behavior. User defined functions can be written in c or C++. These functions can be as complex as is required to model the spacecraft.

The models have attributes, commands, ground events, activation element and subroutines. The attributes are parameters that provide modeling data. The commands are spacecraft commands. The ground events put ground information triggers in the list of events to be modeled. The activation element (called stimuli) places an entry in the list of events to be simulated. Each of these elements has its own editor. Once users create a new model, these elements are automatically placed in the Spacecraft Model View tree shown in Figure 5.

In addition, MPS editor provides context sensitive help and syntax checking during the creation of the model. Because it is more convenient to correct mistakes in the adaptation before it is used for modeling by Seqgen, the Spacecraft Model Adaptation Perspective employs a validation engine to check element fields and implementations. This validation engine checks for syntax, type, and range errors. Any errors which are found are reported in the Validation Errors View. Clicking on an item in this view opens the editor for the element containing the error and also highlights this element in the adaptation.

These capabilities make it easier and more productive for the developer to complete the adaptation task. Traditionally, adaptors have created adaptations by hand and/or via a set of various scripts and legacy programs, but the Spacecraft Model Adaptation Perspective provides all of the editing, merging, and validation functions within one application, making it easier and faster for adaptors to complete their tasks.

V. Sequence Development Perspective

The Sequence Development Perspective is considered by some to be the most important perspective for the operations teams. In this perspective, operations teams build their sequences of commands. In the Sequence Development Perspective, ops team members can use the activities developed in the Activity Dictionary Perspective and the adaptation created in the Spacecraft Model Adaptation Perspective the sequence is checked by the discrete event simulator (Seqgen). This perspective is similar to the other perspectives in providing support to users but in this case it supports building sequences and sequence components. Figure 7 shows the Sequence Development Perspective and its display areas.

Using this perspective the operations teams can create or edit a sequence. Existing sequences can be in any one of four different formats that are used at JPL. However, the internal representation is in a single format called Sequence XML (SML). Since SML is the internal representation, other formats can be adapted to be imported and then converted to SML. At this point the information could be used in Seqgen (the MPS command simulator) to model the commands. In addition, to being able to enter sequences in multiple formats, MPS Editor is able to output the data in any one of the four JPL formats or in the SML format.

Once the sequence has been created or opened for editing, users can create a new request. A request is comprised of activities that were previously defined in the Activity Dictionary Perspective, commands, notes, etc. The ops teams can also create new sequence components and there are several types available. These components can be similar to the ones created in the Activity Dictionary Perspective, but typically they will not have had the extensive testing that has been performed on the activity dictionary activities. The components can also be cyclics (activities that run for a specific number of iterations at specific intervals) or real-time on-board blocks that are used with a JPL on-board sequencing engine Virtual Machine Language (VML). There are several other types that are specific to the Mars Exploration Rovers. For flexibility the Sequence Development Perspective also allows for local variables, epochs, checks to be performed, as wells as defining new parameters or attributes. Each one of these capabilities has a specific editor for the operations team member to use.

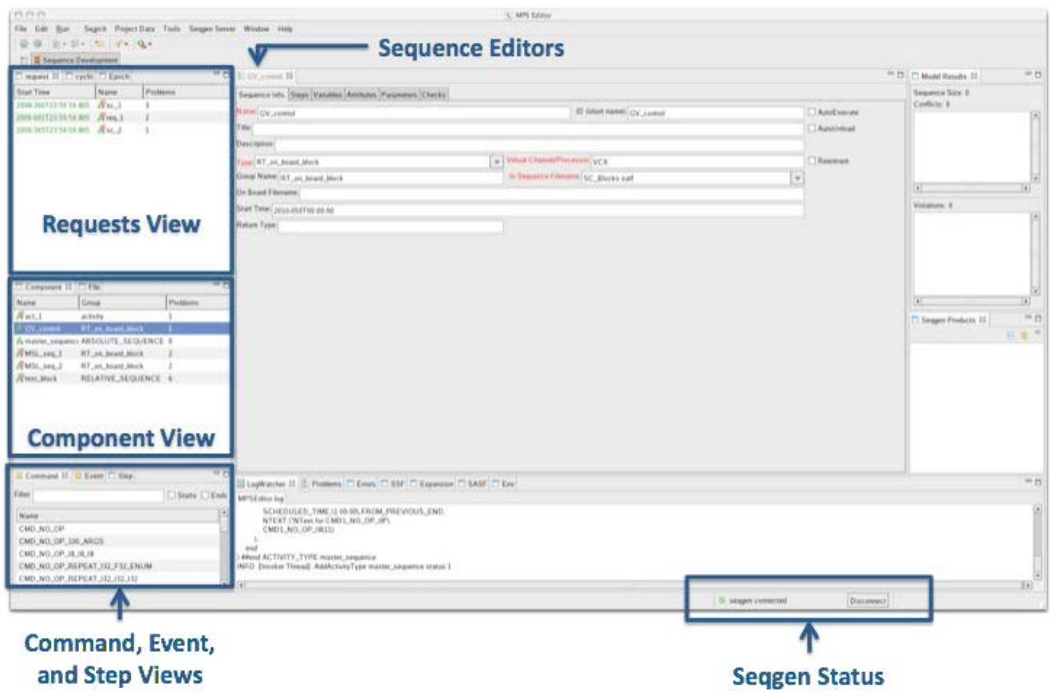

Figure 7. The Sequence Development Perspective and its display areas identified.

Once users have created or edited their sequences, they can select to validate the sequence and make sure that all of their entries are correct. The errors are represented by the red bar as shown in Figure 8. If a user clicks one of these indicators the cursor will jump to the location in the file where the error occurred and a user can easily view and fix the error. Users can also run MPS Editor in batch mode for either validation or for file conversions.

One of the best features is the ability to run Seqgen, the command simulator, after creating a sequence. The Seqgen Status indicator appears green if Seqgen is available and connected to the current session of MPS Editor. Users can disconnect Seqgen while they are creating the sequence and then connect when they are ready to validate the sequence. Seqgen will model the sequence and then provide output products and provide the conflicts and violations detected during modeling. All of this information is displayed. This functionality provides immediate feedback for the operations teams.

The Sequence Development Perspective provides operations teams with an easy to use tool for creating sequences. This perspective has been reviewed by numerous teams at JPL and the response has been overwhelmingly positive.

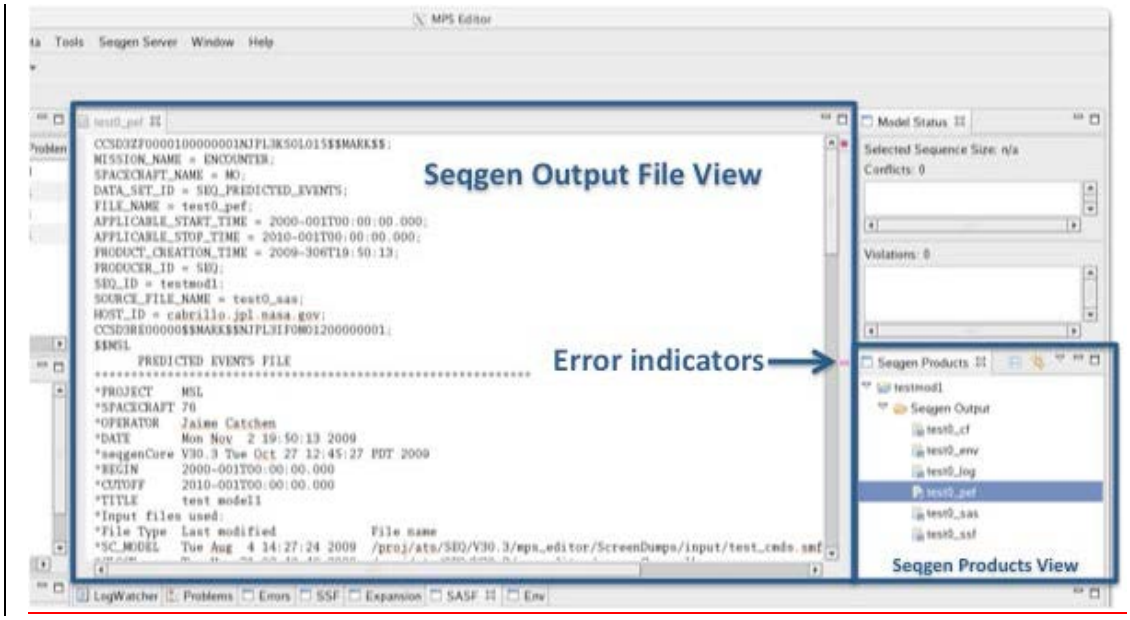

Figure 8. The Sequence Output View of the Sequence Development Perspective.

VI. Sequence Modeling Perspective

The Sequence Modeling Perspective is the perspective that controls the use of the discrete event simulator (Seqgen). Seqgen can be invoked from the Sequence Development Perspective, but if users want to look at the various inputs that are being used or to select them, then they should use this perspective. Users can look at the overview and select the configuration file to be used (env file) as shown in Figure 9 or they can select the various tabs along the top to look at the various inputs that are included in this invocation. Finally, they can look at the entire contents of one of the inputs under the "file" tab. Users can also select the types of outputs that they want including the DSN keyword file (DKF), Sequence of Events (SOE) file and others. On the left, in the navigator window, the output products that are produced are shown in tree form. Users can select one and drag that output into the editor view to look at the information that it contains. The number of model conflicts and violations are on the right side of the display.

In using the Sequence Modeling Perspective the operations teams can select various configurations of Seqgen. They can run a local copy or a remote one. They can also select to run Seqgen in either a batch mode or server mode. Typically if users want feedback in the near term or they are modeling a partial sequence, activity or adaptation, they will select to run in server mode. On the other hand if they are running a long sequence or debugging a large adaptation, they will probably select using the batch mode.

For this perspective the Sequence Modeling Perspective is the client and Seqgen is the server. This MPS Editor perspective communicates with Seqgen using XML Remote Procedure Call (RPC) protocol. The data is passed using Representational State Transfer (ReST). ReST is a simple interface to transmit data using HTTP messaging functionality. The web service version of Seqgen is used for this perspective and the Sequence Development Perspective.

This perspective is still in work at this time. It is part of the forward work on MPS Editor. As this perspective is used by the operations teams, additional features will be added based on their comments and suggestions.

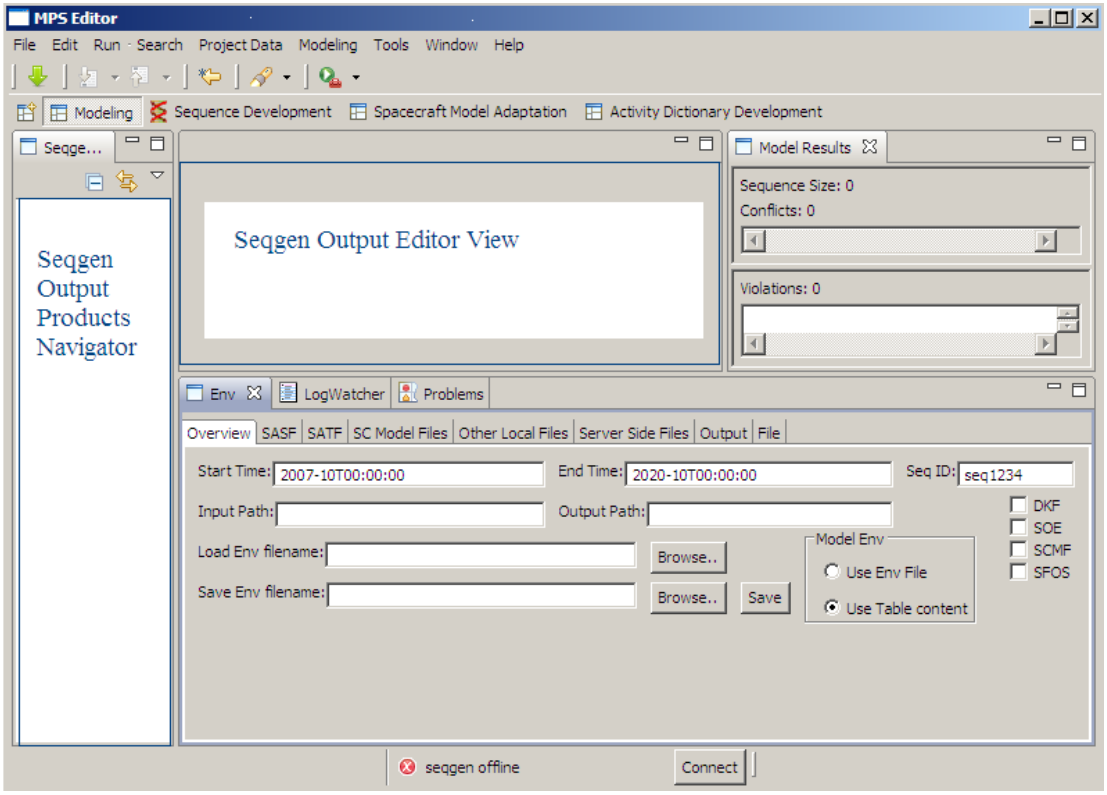

Figure 9. The Sequence Modeling Perspective shown with the display areas marked.

VII. Conclusion

MPS Editor provides an environment for sequencing from the start of a mission through operations. At the start of the mission, the Activity Dictionary Perspective can be used to define repeatable spacecraft activities that are comprised of spacecraft commands for performing calibrations, instrument observations or others. These activities can be parameterized so that they become a template for tasks to be performed on the spacecraft. Next, in the mission lifecycle after spacecraft models are known, MPS Editor's Spacecraft Model Adaptation Perspective can be used to create software models of the hardware to simulate the effects of commands on the state of the spacecraft. Finally, once the operations teams have formed, the Sequence Development Perspective can be used to create sequences containing activities or other spacecraft sequence constructs. At each of these stages the ops team members can go to a different perspective and make modifications and updates. The last perspective, Sequence Modeling Perspective, gives users the ability to immediately run the command simulator (Seqgen) with their activities, model adaptations, or sequences. MPS Editor is a single sequencing environment for the operations teams. Users only need to learn one application and one user interface. Information is presented in a cohesive, integrated set of displays. MPS Editor aids in productivity and reduces frustration for operations teams by presenting a complete sequencing environment. MPS Editor is currently being used by Mars Science Laboratory, and Cassini, Juno and Grail plan to use MPS Editor in the near future. It has turned into one of the more popular applications on the JPL block.

Acknowledgments

The work described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. This work is funded by Multi-Mission Ground Systems and Services (MGSS).

References

[1]Streiffert, Barbara A., and O'Reilly, Taifun, "The Evolution of Seqgen," *AIAA Meeting Papers on Disc* [CD-ROM], Vol. 1, AIAA 2008-3523, SpaceOps 2008, Heidelberg, Germany, 2008.