

# **Advances in Science Planning Tools with the Science Opportunity Analyzer**

**Carol A. Polanskey, Barbara Streiffert, Taifun O'Reilly, Joshua Colwell**

Jet Propulsion Laboratory  
California Institute of Technology  
National Aeronautics and Space Administration  
4800 Oak Grove Drive, Pasadena, CA 91109-8099 USA  
[Carol.Polanskey@jpl.nasa.gov](mailto:Carol.Polanskey@jpl.nasa.gov) (818) 393-7874  
[Barbara.Streiffert@jpl.nasa.gov](mailto:Barbara.Streiffert@jpl.nasa.gov) (818) 354-8140  
[Taifun.Oreilly@jpl.nasa.gov](mailto:Taifun.Oreilly@jpl.nasa.gov) (818) 354-1170

Laboratory for Atmospheric & Space Physics  
University of Colorado  
Boulder, CO. 80309-0392 USA  
[Joshua.Colwell@lasp.colorado.edu](mailto:Joshua.Colwell@lasp.colorado.edu) (303) 492-6805

## **ABSTRACT**

For many years the diverse scientific community that supports JPL's wide variety of interplanetary space missions has needed a tool in order to plan and develop their experiments. The tool needs to be easily adapted to various mission types and portable to the user community. The Science Opportunity Analyzer, SOA, now in its third year of development, is intended to meet this need. SOA is a java based application that is designed to enable scientists to identify and analyze opportunities for science observations from spacecraft. It differs from other planning tools in that it does not require an in-depth knowledge of the spacecraft command system or operation modes to begin high level planning. Users can, however, develop increasingly detailed levels of design.

SOA consists of five major functions: Opportunity Search, Visualization, Observation Design, Constraint Checking, and Communications. Opportunity Search is a GUI-driven interface to existing search engines which can be used to identify times when a spacecraft is in a specific geometrical relationship with other bodies in the solar system. This function can be used for advanced mission planning as well as for making last minute adjustments to mission sequences in response to trajectory modifications. Visualization is a key aspect of SOA. The user can view observation opportunities in either a 3D representation or as a 2D map projection. The user is given extensive flexibility to customize what is displayed in the view. Observation Design allows the user to orient the spacecraft and visualize the projection of the instrument field of view for that orientation using the same views as Opportunity Search. Constraint Checking is provided to validate various geometrical and physical aspects of an observation design. The user has the ability to easily create custom rules or to use official project-generated flight rules. This capability may also allow scientists to easily impact the cost to science if flight rule changes occur. Finally, SOA is unique in that it is designed to be able to communicate with a variety of existing planning and sequencing tools.

From the very beginning SOA was designed with the user in mind. Extensive surveys of the potential user community were conducted in order to develop the software requirements. Throughout the development period, close ties have been maintained with the science community to insure that the tool maintains its user focus. Although development is still in its early stages, SOA is already developing a user community on the Cassini project which is depending on this

tool for their science planning. There are other tools at JPL that do various pieces of what SOA can do; however, there is no other tool which combines all these functions and presents them to the user in such a convenient, cohesive, and easy to use fashion.

## **1.0 Introduction**

Spacecraft tend to be “closed systems” similar in many ways to cars. Both have steering mechanisms; both need fuel and both have instrumentation. However, the instrumentation is different. In cars instruments are generally indicators that give the driver information. On spacecraft they can be scientific instruments that perform observations and collect data. Another difference is that many spacecraft rely on commands sent from the ground to turn hardware on and off, perform diagnostic checks and start instrument observations because they don’t carry human drivers to carry out these functions. They only have computers to relay the information to the spacecraft hardware.

Unlike cars, spacecraft operations are tied to an event-driven timeline that is governed by orbital mechanics. It is not possible to stop the spacecraft to make decisions about the best observation to be made next. These characteristics along with the finite nature of spacecraft resources (there are no gas stations in space) place a premium on planning spacecraft activities. Science Opportunity Analyzer (SOA) is a software tool with broad functionality designed to meet this need for planning.

In order for the scientist to be able to get the highest quality data and the best observations, the scientist needs to have information that shows that the planned observation not only meets with the scientific objectives, but also meets with reality. However, in the past, the science user has been left to develop ad hoc tools that are specific to a particular space mission for a specific instrument built solely for that mission. These tools have not allowed the users to share observation information easily and tend to be a bare minimum of what is actually needed. Generally, these tools don’t communicate with other software tools that are used to setup the commands that are to be sent to the spacecraft. Science Opportunity Analyzer (SOA), a software tool, has been built to fulfill the need of assuring that the science objectives can be met as well as the needs of sharing information and entering the observation into the pipeline that ultimately results in commands to the spacecraft.

For this tool to meet the needs of the user community, it has been important to find out what the end user needed. A methodology called “Quality Functional Deployment” (see, e.g., Belhe and Kusiak, 1996) or “Obtaining the Voice of the Customer” was used. A cross discipline group of scientists, software engineers and system engineers met and created an open-ended questionnaire and interviewed 40 selected stakeholders in the software. This group also developed a closed-ended set of questions (short answer, fill-in-the-blank, multiple choice, rank, etc.) as a check of the results of the interviews using the open-ended questionnaire. The results of the interviews were then transformed into required functional capabilities and ways to measure those functional capabilities. It was important for this group to continue in some way to insure that the software remained true to its charter. The software and system engineers formed the SOA Development Team, and developed the software requirements and the top-level design. The science members became members of the SOA Standing Review Board. The software requirements and the design reviews have been held before this evaluation board. In this way the software has continued to be implemented based on the “Voice of the Customer”.

From the interviews several basic scenarios of how a science user would use a tool like SOA were developed. The following is one typical high-level scenario.

1. One or more time periods of opportunity that satisfy entered geometric criteria are found.
2. The science user selects one of the time periods and chooses to see a display of that time in either a 2-dimensional or a 3-dimensional view.
3. The user determines the time window with the most potential and proceeds to design the observation. During this time the user continues to check the display of the design to make sure that the design, as it unfolds, continues to meet the science objectives.
4. As part of the design process the science user chooses to have the tool check the design against spacecraft constraints. These constraints may be geometric in nature. For example, the instrument may not be able to have the Sun in its field of view without damaging the instrument. The constraints may be hardware state driven. In this case, an example would be that the spacecraft couldn't actually turn as quickly as desired.
5. After the design is constraint-free, the science user refines the design and saves it for future recall.
6. Finally, the user adds the design to the plan of activities that are to be sent to the spacecraft. Constraint checking will be performed again once all of the observations and other spacecraft tasks are entered in the plan of activities.

At any place in the scenario, the science user can go back to a previous step and make changes as needed or desired. SOA has been built to perform all of these tasks. It consists of five major functional areas: Opportunity Search, Visualization, Observation Design, Constraint Checking and Communications.

Before proceeding with a more detailed description of the major functional areas, it is important to understand the process of sending commands to a spacecraft. At The Jet Propulsion Laboratory (JPL), this process is called the Uplink Process. It consists of engineering and science groups deciding on the tasks and observations that they want the spacecraft to perform. These tasks are defined and all of them are placed on a time-line that forms an operational plan for the spacecraft. The time-line is refined so that it contains no constraint violations. In order to eliminate the conflicts the tasks in violation are sent to their respective submitters for modification and then resubmitted. It is important that the task/observation is initially constraint free or it is possible that the task/observation will be removed from the plan. All of the functional areas in SOA support an observation being added to the operational plan.

## **2.0 Science Opportunity Analyzer (SOA)**

Currently, SOA is a java-based application that runs on Suns under Sun Solaris and PCs under NT, XP, 2000 and Linux. It is a multi-mission tool and can be easily configured for different missions. It utilizes Swing, Java 3-D, Java 2-D and XML extensively. SOA uses a hierarchical approach to objects so that project specific objects can be easily added. The project specific objects form the lowest tier of the hierarchy. SOA has tabs (see Figure 2.1) that represent the major work areas of Opportunity Search, Observation Design (Visualization and Spacecraft Task Selection), Constraint Checking and Communications. For the next delivery another tab has been added for Output Data. SOA uses spacecraft trajectory information, planetary constants and spacecraft information provided by JPL Navigation.

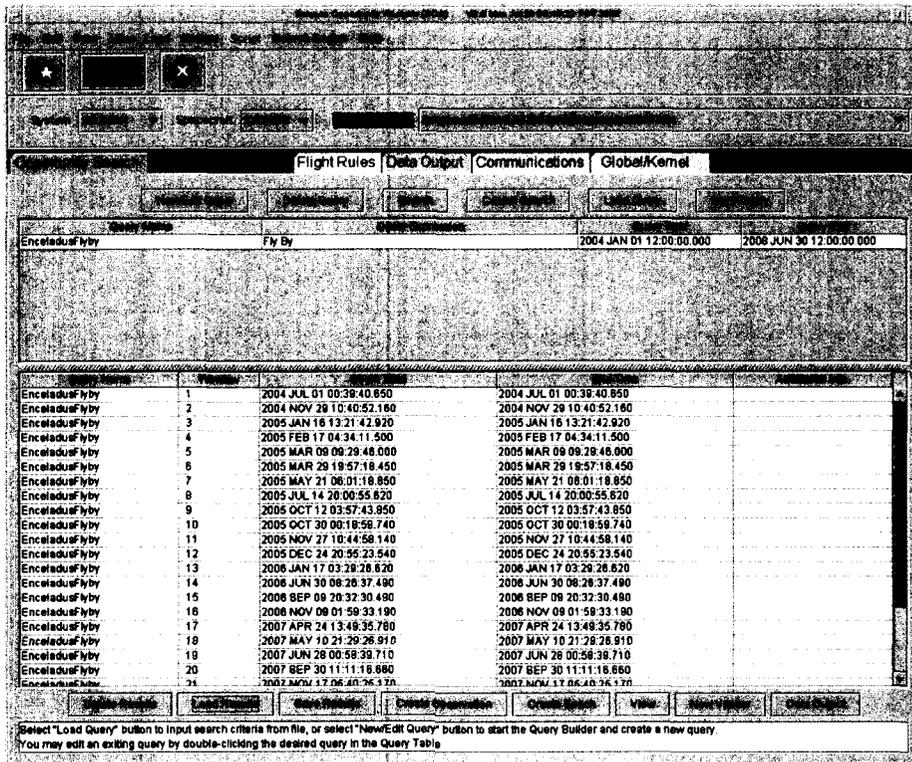


Figure 2.1 shows the Opportunity Search display. A search for a flyby of Enceladus has been performed and the time periods that met the search criterion are shown.

## 2.1 Opportunity Search

In the user scenario above the first area the science user accesses is Opportunity Search. Opportunity Search allows the science user to identify times when a spacecraft is in a specific geometric relationship with other bodies in the solar system. This functional area allows the user to select from a list of more than thirty geometric search criteria including periapse times and apoapse times and various illumination geometries. These search criteria are based on continuous functions that occur either at a specific time (for example, a certain distance from a celestial body) or over a time span (for example, an occultation). A search criterion can be created or entered from a file of previously created criteria. If the search criterion is new, the science user is presented with a drag and drop graphical user interface. The interface also displays a list of the information that is needed by that search criterion – called properties. The science user enters the desired properties associated with the selected search criterion including the celestial bodies involved and other pertinent information such as angles or distance. A search criterion can be a simple single search or a more complex search combining multiple search criteria using Boolean operators of “and”, “or”, and “not”. Once the search criterion is created and written to the list, the science user selects to have the software perform the search. The time periods when the geometric criteria have been satisfied are presented to the user in a list (see Figure 2.1).

SOA uses two search engines that have been created at Jet Propulsion Laboratory. Each of these search engines requires the input data to be entered a specific way. SOA has divided the Opportunity Search objects into two groups – the software models that contain the values for the Opportunity Search criterion properties and the templates that put the properties in the correct format for the chosen search engine. This scheme allows new search engines to be added easily, and for search criterion to be easily changed. Finally, this scheme permits the objects to be

discovered at runtime. “Discovery at runtime” means that SOA loads into the software only the objects that are available to be used. If a search criterion is not needed, then it is simply removed.

## **2.2 Visualization**

Now that the science user has found the time(s) that match the geometric criteria, the next step is to look at a picture of the information. SOA allows the science user to select from several view options: 3 dimensional perspective projection, 3 dimensional arbitrary observer, 2 dimensional sky map and 2 dimensional trajectory plot. The perspective projection renders the view from the point of view of a specified observer looking at a target. Generally, the observer is a spacecraft and the target is a celestial body. The arbitrary observer view is a parallel projection that is rendered from an observer who can be arbitrarily placed in space by the user. The 2 dimensional sky map is an equidistant cylindrical map projection of the celestial sphere as viewed from the spacecraft. The 2 dimensional trajectory plot is a view of the spacecraft’s trajectory around the target body. If the target body has satellites, this display also shows their orbits. This plot can be viewed from the ecliptic or the equatorial planes. The user can select items to be included in the picture such as: Right Ascension/Declination (RA/Dec) grid, latitude/longitude grid, stars, magnetic field, planets, satellites, light/dark terminator, and other geometric information. If an item is not appropriate for a view, that selection is not made available to the user. For example, in the perspective view, the spacecraft trajectory can’t be seen since the observer is the generally the spacecraft itself. The selection to make it visible is not presented to the user. The user can also chose to see more than one view in a single window or multiple windows can be rendered with different views.

In Visualization the same hierarchical approach as Opportunity Search has been taken. The real world coordinates and formulas plus the characteristics of the real world entity form the software model objects. The actual Java 3-D constructs form the primitive objects. For example, an RA/Dec grid is comprised of a model object that has its properties of a line model, a text model, the grid spacing for both Right Ascension and Declination, the label spacing for both, etc. The associated primitive sets the Java 3-D components of appearance and the attributes for both the lines and the labels. This approach again allows a specific project to easily add, modify, customize or delete objects that are specific to that particular project. All of the objects are discovered at runtime.

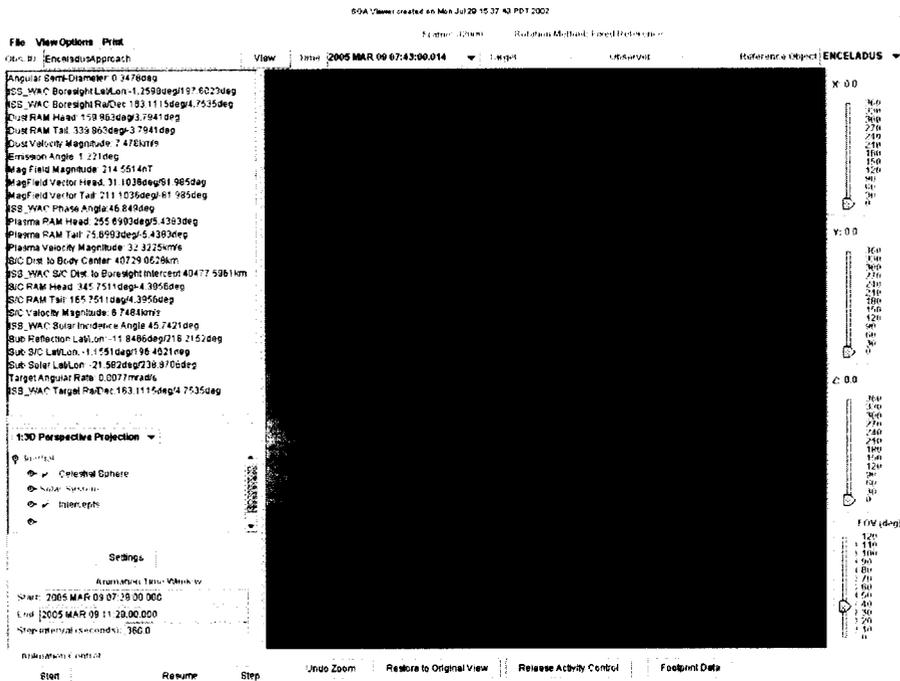


Figure 2.2 is a perspective projection of the closes approach of the Cassini spacecraft to Enceladus. The red square is a field of the view of the camera projected onto the sky. It is red because it violates a constraint.

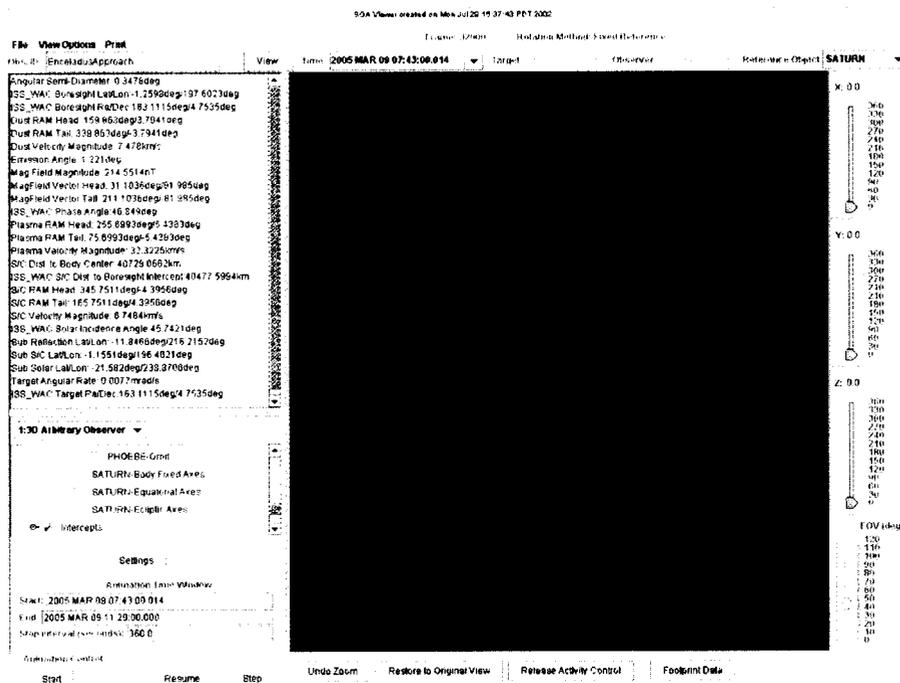


Figure 2.3 is an arbitrary observer view of the pole of Saturn. In this view the spacecraft trajectory is also shown. The lines originating at the spacecraft are various physical phenomena.

## 2.3 Observation Design

Once the user sees a picture that conforms to the desired objectives, an observation design can be started. The science user can first choose to just look at the time and by specifying the spacecraft attitude, the user can look at a display that shows the scene, but also contains an instrument field of view. A field of view is an instrument aperture; generally they are squares, rectangles or circles. These fields of view can be projected onto the target – similarly to the way a person uses a camera with a viewfinder. This projection gives the scientist an idea of the coverage of the observation. Once the science user is satisfied with the coverage and the view, an observation type is selected.

The current choices are start-stop mosaic, continuous scan, roll about an axis, and stare. A start-stop mosaic consists of a series of pictures that are taken. The spacecraft or instrument platform or the instrument itself is moved to a location and waits while a picture is taken. This step is repeated until all the desired pictures are taken for the observation. A continuous scan is a series of measurements made at different pointing geometries while the spacecraft or instrument platform or the instrument itself are continuously moving. Roll about an axis is an observation that is performed while the spacecraft is rotating around a single axis. The stare observation is simply one that is performed while the spacecraft maintains a fixed attitude with respect to the target. Each observation type has properties that must be selected. There are general properties such as the target and observer that apply to all of the observation types.

There are also properties that are specific to a particular type of observation such as the roll axis for a roll about an axis observation. Once the properties are selected, the user can choose to view the observation again with the new information that has been provided. At this point the scientist may want to animate the depiction to see how the scene changes over time. The user may review and change the properties and re-plot the depiction as many times as desired.

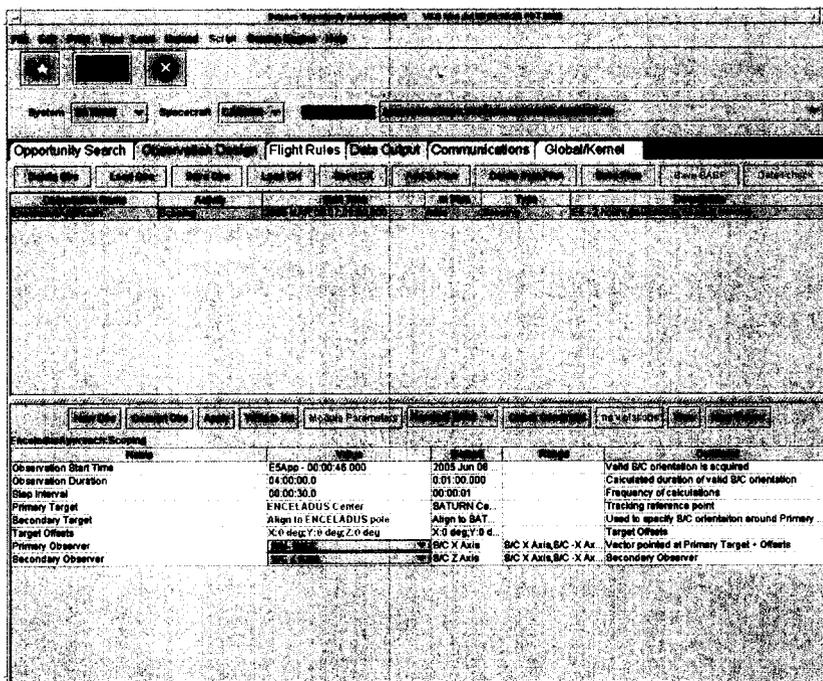


Figure 2.3 is the Observation Design display. The observation that is shown is a scoping level observation.

For Observation Design the software objects come in two flavors. The main object contains the information common to these observations like the start time and the target. The secondary object specifies the information that is specific for that type of observation, like the number of pictures to be taken. In addition to these objects, this area has objects that map the SOA observation properties to other tools such as the software that contains the plan of spacecraft activities. Most spacecraft missions have their own way of specifying observations and other spacecraft tasks (or activities). This area has a strong hierarchical component so that missions will have an easier time adding mission specific observations.

## 2.4 Constraint Checking

At this point if the science user has not already performed constraint checking on the observation, it is time to make sure that there are no constraint violations. The constraint violations are of two varieties. The first variety is the group of constraints that are geometric in nature. This group consists of various exclusion zones or impediments to performing the observation. An exclusion zone might be an angle that specifies a region where the Sun is too bright for an instrument or an area where another bright body is visible and may hurt a sensitive instrument. Sometimes it is not damaging if the distance from the bright body places the instrument in a safe zone or if the exposure to the bright light is below a given threshold. The exclusion zone object has four variations. It can simply be an angle that must be excluded. It can be an angle with a distance attached. It can also be either of these two with an exposure time attached. Impediments may not be dangerous, but may cause the observation not to meet its objectives. An example of this case might be the occulting of the target body by another body or that another body is transiting across the target body in such a way as to spoil the observation. Currently, this type of violation hasn't been implemented. The second group consists of state violations. Examples of this type could be that the spacecraft maximum rates and/or accelerations are exceeded. The converse could also be true – the minimum rates and/or accelerations are not met. For either type of constraint, the user enters the required properties through the drag-and-drop user interface. If the user finds that the observation causes constraints to be violated, the observation can be modified and the process can begin again.

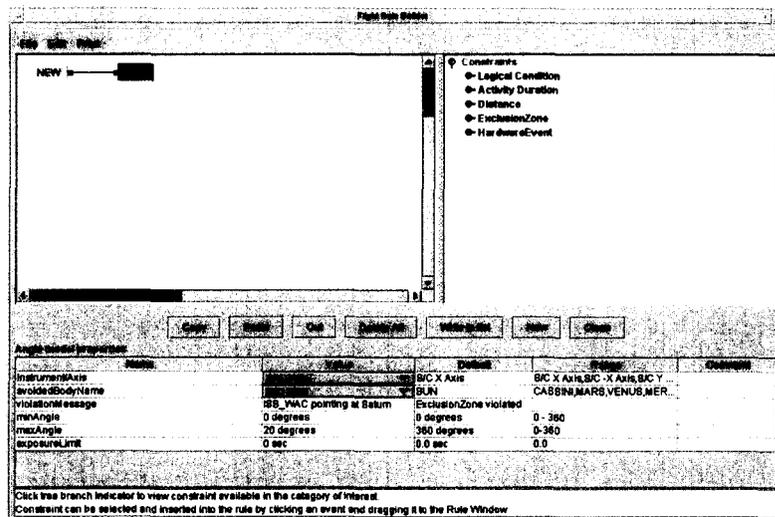


Figure 2.4 shows the Constraints rule builder with the angle exclusion zone selected.

Flight rule objects consist of building blocks objects that can be combined to create the constraint rules. Each exclusion zone type has its own combined object. The drag-and-drop graphical user interface can be used to create the mission specific rules for exclusion zones. In addition, there are rate objects and acceleration objects. Since it is possible to have spacecraft rates, instrument platform rates and articulating instrument rates, the specific space mission can tailor these building blocks to their own needs also using the graphical user interface. The object hierarchy allows missions to add different types of rules that haven't been provided by SOA.

## **2.5 Communications**

The last task that the science user performs using SOA is to place the finished observation in the plan of activities with all of the other observations and the engineering tasks (like calibrations and maneuvers). SOA is the first tool designed to communicate with other tools used for developing the plan of activities for the spacecraft. In the past the scientist had to create the observation and then the information had to be re-entered into the software that would prepare it for the spacecraft. Now SOA communicates with that software either by using inter-process communications (IPC) or through the use of files. The planning software provides a visual timeline and resource consumption graphs. SOA communicates with this software tool using inter-process communication. Observations are sent directly to this software by simply pressing a button. Other legacy software requires input via files. SOA, also, creates these files so that they can be ingested into these legacy tools.

In addition to allowing the science user the ability to send their observations designs to other software, SOA allows the science user to share the observation information with other scientists. SOA can save the observation information in the form of a C-Kernel file (a binary file that contains quaternions for the spacecraft's attitude over time). The C-Kernel file is a relatively standard file used by many scientists and engineers in the space industry. By producing this file, SOA permits software applications written by others to ingest the observation information. C-Kernel files are maintained by the Planetary Data System Navigation and Ancillary Information Facility (NAIF) at JPL

Communication objects exist at two levels. The first level of objects contains the data that forms the observation. These objects have the properties for the observation and how those properties are to be translated to the planning software tools. The second level of objects contains the messages that are to be sent to the planning software through IPC. A corresponding set of objects contains the information on how to write the information to the files (the observation file and the C-kernel). Again, this separation allows projects to easily add or change the data or the format to meet their specific needs.

## **3.0 Technical Challenges**

Creating SOA with these capabilities has not been an easy task. When SOA began, Java 1.2 had not been released and Java 3-D had had only a few releases. There were performance issues, memory leaks (on the SOA side as well as the Java side), and questions of accuracy in the graphics presentation. In addition, translating the data so that it could be recognized by other software has been difficult in terms of making sure that apples were translated as apples and not to oranges. At this time Java and Java 3D have made significant advances and SOA has had the support of Java experts at Sun. Additionally, SOA has been supported by several members of the science community in overcoming the obstacles. The end result is a reliable, stable SOA containing a wide variety of fully functional capabilities.

#### **4.0 Conclusion**

In conclusion, the approach that has been taken in creating SOA has been to keep the scientist in mind at all times. It began by collecting the science user's needs and proceeded by keeping this user involved throughout the project. The tool fills a void that has existed since science instruments were placed on a spacecraft. Many people have envisioned a tool of this nature. SOA is the beginnings of all those visions. Over time it will continue to improve to meet those expectations. But most importantly, SOA enables the scientist to create his/her observation easily.

#### **References:**

Belhe, U. and A. Kusiak, The House of Quality in a Design Process, International Journal of Prod. Res., Vol. 34, No. 8, 2119-2131, 1996

#### **Acknowledgements**

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.

We would also like to thank the Cassini Project for their support.