scheduled or physical events. Users can manipulate objects in the simulation environment under programmatic control. Inputs to the scripting module are Actions, Conditions, and the Script. Actions are arbitrary modifications to constructs such as Platform Objects (i.e., satellites), Sensor Objects (representing instruments or communication links), or Analysis Objects (user-defined logical or numeric variables). Examples of actions include changes to a satellite orbit, changing a sensor-pointing direction, and the manipulation of a numerical expression. Conditions represent the circumstances under which Actions are performed and can be couched in IF-Then-Else logic, like performing \( v \) at specific times or adding to the spacecraft power only when it is being illuminated by the Sun.

The SOAP script represents the entire set of conditions being considered over a specific time interval. The output of the scripting module is a series of events, which are changes to objects at specific times. As the SOAP simulation clock runs forward, the scheduled events are performed. If the user sets the clock back in time, the events within that interval are automatically undone.

This script offers an interface for defining scripts where the user does not have to remember the vocabulary of various keywords. Actions can be captured by employing the same user interface that is used to define the objects themselves. Conditions can be set to invoke Actions by selecting them from pull-down lists. Users define the script by selecting from the pool of defined conditions. Many space systems have to react to arbitrary events that can occur from scheduling or from the environment. For example, an instrument may cease to draw power when the area that it is being illuminated by the Sun.

The SHINE Knowledge Base Interchange Language software has been designed to more efficiently send new knowledge bases to spacecraft that have been embedded with the Spacecraft Health Inference Engine (SHINE) tool. The intention of the behavioral model is to capture most of the information generally associated with a spacecraft functional model, while specifically addressing the needs of execution within SHINE and Livingstone. As such, it has some constructs that are based on one or the other.

As NASA/JPL autonomous science missions go deeper and deeper into space, the collection of unexpected data becomes a problem. Data structures can easily be implemented in advance that can collect any kind of data; however, when it comes to processing the data into information and taking advantage of serendipitous science discovery, designing a fixed and efficient data structure becomes increasingly complex. This software defines and implements a new kind of data structure that can be used for representing information that is derived from serendipitous data discovery. It allows the run-time definition of arbitrarily complex structures that can adapt at run-time as the raw science data is transformed into information.

This solves the problem where decision trees can be prone to, namely how expensive they can be to execute because of the need to evaluate each non-leaf node and, based upon its truth, to either progress deeper into the structure or to examine an alternative. This requires many machine cycles, which can negatively affect time-critical decisions.

This software runs on a variety of different platforms, including SUN, HP, Intel, Apple Macs, Flight Processors, etc. It can be distributed in either source code or binary code and requires a LISP compiler to run with a number, such compilers being either commercially available or found as shareware. The software has no specific memory requirements and depends on the applications that are running in it. It is implemented as a library package and folds into whatever environment is calling it.

Currently, this software is a component of the Common Automation Engine (CAE) that was developed for Deep Space Network (DSN). It has been in active use for over three years and has been installed in a shadow mode running at Goldstone and DSN monitoring operations at JPL.

This work was done by Mark James, Ryan Mackey, and Raffi Tikidjian of Caltech for NASA’s Jet Propulsion Laboratory. Further information is contained in a TSP (see page 1). This software is available for commercial licensing. Please contact Karina Edmonds of the California Institute of Technology at (626) 395-2322. Refer to NPO-44546.

**XML-Based SHINE Knowledge Base Interchange Language**

The SHINE Knowledge Base Interchange Language software has been designed to more efficiently send new knowledge bases to spacecraft that have been embedded with the Spacecraft Health Inference Engine (SHINE) tool. The intention of the behavioral model is to capture most of the information generally associated with a spacecraft functional model, while specifically addressing the needs of execution within SHINE and Livingstone. As such, it has some constructs that are based on one or the other.

As NASA/JPL autonomous science missions go deeper and deeper into space, the collection of unexpected data becomes a problem. Data structures can easily be implemented in advance that can collect any kind of data; however, when it comes to processing the data into information and taking advantage of serendipitous science discovery, designing a fixed and efficient data structure becomes increasingly complex. This software defines and implements a new kind of data structure that can be used for representing information that is derived from serendipitous data discovery. It allows the run-time definition of arbitrarily complex structures that can adapt at run-time as the raw science data is transformed into information.

This solves the problem where decision trees can be prone to, namely how expensive they can be to execute because of the need to evaluate each non-leaf node and, based upon its truth, to either progress deeper into the structure or to examine an alternative. This requires many machine cycles, which can negatively affect time-critical decisions.

This software runs on a variety of different platforms, including SUN, HP, Intel, Apple Macs, Flight Processors, etc. It can be distributed in either source code or binary code and requires a LISP compiler to run with a number, such compilers being either commercially available or found as shareware. The software has no specific memory requirements and depends on the applications that are running in it. It is implemented as a library package and folds into whatever environment is calling it.

Currently, this software is a component of the Common Automation Engine (CAE) that was developed for Deep Space Network (DSN). It has been in active use for over three years and has been installed in a shadow mode running at Goldstone and DSN monitoring operations at JPL.

This work was done by Mark James, Ryan Mackey, and Raffi Tikidjian of Caltech for NASA’s Jet Propulsion Laboratory. Further information is contained in a TSP (see page 1). This software is available for commercial licensing. Please contact Karina Edmonds of the California Institute of Technology at (626) 395-2322. Refer to NPO-44546.
parison and assessment of lab requirements. Budgets and program estimates are based on empirical data.

Features of the system include addition, modification, deletion, and viewing of lab hardware and software equipment data. Equipment data include equipment name, type, description, manufacturer, and other key parameters. CTCLMS also prioritizes obsolete equipment replacement at three levels of responsibility, manages the addition of new users and the assignment of roles and submits current lab conditions and costs.

This work was done by Linda Shaykhian, Curtis Dugger, and Laurie Griffin for Kennedy Space Center. Further information is contained in a TSP (see page 1). KSC-13051

**MRO SOW Daily Script**

The MRO SOW daily script (wherein “MRO” signifies “Mars Reconnaissance Orbiter” and “SOW” signifies “sequence systems engineer of the week”) is a computer program that automates portions of the MRO daily SOW procedure, which includes checking file-system sizes and automated sequence processor (ASP) log files. The MRO SOW daily script effects clear reporting of (1) the status of, and requirements imposed on, the file system and (2) the ASP log files.

This program was written by Forest W. Fisher, Teerapat Khanamponpan, and Roy E. Gladden of Caltech for NASA’s Jet Propulsion Laboratory.

This software is available for commercial licensing. Please contact Karina Edmonds of the California Institute of Technology at (626) 395-2322. Refer to NPO-45439.