The model library can be used to support SysML user models in various ways. A simple approach is to define and document libraries of reusable systems of units and quantities for reuse across multiple projects, and to link units and quantity kinds from these libraries to Unit and QuantityKind stereotypes defined in SysML user models.

This work was done by Nicolas F. Rouquette of Caltech, Hans-Peter DeKoenig of the European Space Agency, Roger Burkhart of Deere & Company, and Huascar Espinoza of the French Centre of Atomic Energy for NASA's Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov.

The software used in this innovation is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47251.

Sptrace

NASA's Jet Propulsion Laboratory, Pasadena, California

Sptrace is a general-purpose space utilization tracing system that is conceptually similar to the commercial "Purify" product used to detect leaks and other memory usage errors. It is designed to monitor space utilization in any sort of "heap," i.e., a region of data storage on some device (nominally memory; possibly shared and possibly persistent) with a flat address space. This software can trace usage of shared and/or non-volatile storage in addition to private RAM (random access memory).

Sptrace is implemented as a set of C function calls that are invoked from within the software that is being examined. The function calls fall into two broad classes: (1) functions that are embedded within the heap management software [e.g., JPL's SDR (Simple Data Recorder) and PSM (Personal Space Management) systems] to enable heap usage analysis by populating a virtual time-sequenced "log" of usage activity, and (2) reporting functions that are embedded within the application program whose behavior is suspect. For ease of use, these functions may be wrapped privately inside public functions offered by the heap management software. Sptrace can be used for VxWorks or RTEMS real-time systems as easily as for Linux or OS/X systems.

This work was done by Scott C. Burleigh of ACRO 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 Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-41626.

S-Band POSIX Device Drivers for RTEMS

NASA's Jet Propulsion Laboratory, Pasadena, California

This is a set of POSIX device driver level abstractions in the RTEMS RTOS (Real-Time Executive for Multiprocessor Systems real-time operating system) to S-Band radio hardware devices that have been instantiated in an FPGA (field-programmable gate array). These include A/D (analog-to-digital) sample capture, D/A (digital-to-analog) sample playback, PLL (phase-locked-loop) tuning, and PWM (pulse-width-modulation)-controlled gain. This software interfaces to S-band radio hardware in an attached Xilinx Virtex-2 FPGA. It uses plug-and-play device discovery to map memory to device IDs. Instead of interacting with hardware devices directly, using direct-memory mapped access at the application level, this driver provides an application programming interface (API) offering that easily uses standard POSIX function calls. This simplifies application programming, enables portability, and offers an additional level of protection to the hardware.

There are three separate device drivers included in this package: sband_device (ADC capture and DAC playback), pll_device (RF front end PLL tuning), and pwm_device (RF front end AGC control).

This work was done by James P. Lux, Minh Lang, Kenneth J. Peters, and Gregory H. Taylor of Caltech for NASA's Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov.

This software is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47496.

MaROS: Information Management Service

NASA's Jet Propulsion Laboratory, Pasadena, California

This software is provided by the Mars Relay Operations Service (MaROS) task to a variety of Mars projects for the purpose of coordinating communications sessions between landed spacecraft assets and orbiting spacecraft assets at Mars. The Information Management Service centralizes a set of functions previously distributed across multiple spacecraft operations teams, and as such, greatly improves visibility into the end-to-end strategic coordination process. Most of the process revolves around the scheduling of communications sessions between the spacecraft during periods of time when a landed asset on Mars is geometrically visible by an orbiting spacecraft. These "relay" sessions are used to transfer data both to and from the landed asset via the orbiting asset on behalf of Earth-based spacecraft operators.

This software component is an application process running as a Java virtual

machine. The component provides all service interfaces via a Representational State Transfer (REST) protocol over "https" to external clients. There are two general interaction modes with the service: upload and download of data. For data upload, the service must execute logic specific to the upload data type and trigger any applicable calculations including pass delivery latencies and overflight conflicts. For data download, the software must retrieve and correlate requested information and deliver to the requesting client.

The provision of this service enables several key advancements over legacy processes and systems. For one, this service represents the first time that end-to-end relay information is correlated into a single shared repository. The software also provides the first multimission latency calculator; previous latency calculations had been performed on a mission-by-mission basis.

This work was done by Daniel A. Allard, Roy E. Gladden, Jesse J. Wright, Franklin H. Hy, Gregg R. Rabideau, and Michael N. Wallick 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 Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47454.

Interplanetary Overlay Network Bundle Protocol Implementation

NASA's Jet Propulsion Laboratory, Pasadena, California

The Interplanetary Overlay Network (ION) system's BP package, an implementation of the Delay-Tolerant Networking (DTN) Bundle Protocol (BP) and supporting services, has been specifically designed to be suitable for use on deep-space robotic vehicles. Although the ION BP implementation is unique in its use of zero-copy objects for high performance, and in its use of resource-sensitive rate control, it is fully interoperable with other implementations of the BP specification (Internet RFC 5050).

The ION BP implementation is built using the same software infrastructure that underlies the implementation of the CCSDS (Consultative Committee for Space Data Systems) File Delivery Protocol (CFDP) built into the flight software of Deep Impact. It is designed to minimize resource consumption, while maximizing operational robustness. For example, no dynamic allocation of system memory is required. Like all the other ION packages, ION's BP implementation is designed to port readily between Linux and Solaris (for easy development and for ground system operations) and VxWorks (for flight systems operations). The exact same source code is exercised in both environments.

Initially included in the ION BP implementations are the following: libraries of functions used in constructing bundle forwarders and convergence-layer (CL) input and output adapters; a simple prototype bundle forwarder and associated CL adapters designed to run over an IP-based local area network; administrative tools for managing a simple DTN infrastructure built from these components; a background daemon process that silently destroys bundles whose time-to-live intervals have expired; a library of functions exposed to applications, enabling them to issue and receive data encapsulated in DTN bundles; and some simple applications that can be used for system checkout and benchmarking.

This work was done by Scott C. Burleigh of ARCO 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 Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-41628.

STRS SpaceWire FPGA Module

NASA's Jet Propulsion Laboratory, Pasadena, California

An FPGA module leverages the previous work from Goddard Space Flight Center (GSFC) relating to NASA's Space Telecommunications Radio System (STRS) project. The STRS SpaceWire FPGA Module is written in the Verilog Register Transfer Level (RTL) language, and it encapsulates an unmodified GSFC core (which is written in VHDL). The module has the necessary inputs/outputs (I/Os) and parameters to integrate seamlessly with the SPARC I/O FPGA Interface module (also developed for the STRS operating environment, OE).

Software running on the SPARC processor can access the configuration and status registers within the SpaceWire module. This allows software to control and monitor the SpaceWire functions, but it is also used to give software direct access to what is transmitted and received through the link. SpaceWire data characters can be sent/received through the software interface, as well as through the dedicated interface on the GSFC core. Similarly, SpaceWire time codes can be sent/received through the software interface or through a dedicated interface on the core.

This innovation is designed for plug-and-play integration in the STRS OE. The SpaceWire module simplifies the interfaces to the GSFC core, and synchronizes all I/O to a single clock. An interrupt output (with optional masking) identifies time-sensitive events within the module. Test modes were added to allow internal loopback of the SpaceWire link and internal loopback of the client-side data interface.

This work was done by James P. Lux, Gregory H. Taylor, Minh Lang, and Ryan A. Stern of Caltech for NASA's Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov.

This software is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47434.