planning purposes.

While other software for this task exists, each at the time of this reporting has been contained within a much more complicated package. This tool allows science and mission operations to view the estimates with a few clicks of the mouse.

This work was done by Michael N. Wallick, Daniel A. Allard, Roy E. Gladden, and Corey L. Peterson 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-47722.

Extended Testability Analysis Tool

The Extended Testability Analysis (ETA) Tool is a software application that supports fault management (FM) by performing testability analyses on the fault propagation model of a given system. Fault management includes the prevention of faults through robust design margins and quality assurance methods, or the mitigation of system failures. Fault management requires an understanding of the system design and operation, potential failure mechanisms within the system, and the propagation of those potential failures through the system.

The purpose of the ETA Tool software is to process the testability analysis results from a commercial software program called TEAMS Designer in order to provide a detailed set of diagnostic assessment reports. The ETA Tool is a command-line process with several user-selectable report output options. The ETA Tool also extends the COTS testability analysis and enables variation studies with sensor sensitivity impacts on system diagnostics and component isolation using a single testability output. The ETA Tool can also provide extended analyses from a single set of testability output files.

The following analysis reports are available to the user: (1) the Detectability Report provides a breakdown of how each tested failure mode was detected, (2) the Test Utilization Report identifies all the failure modes that each test detects, (3) the Failure Mode Isolation Report demonstrates the system's ability to discriminate between failure modes, (4) the Component Isolation Report demonstrates the system's ability to discriminate between failure modes relative to the components containing the failure modes, (5) the Sensor Sensitivity Analysis Report shows the diagnostic impact due to loss of sensor information, and (6) the Effect Mapping Report identifies failure modes that result in specified system-level effects.

The ETA Tool provides iterative assessment analyses for conducting sensor sensitivity studies, as well as a command-line option that allows the user to specify the component isolation level. The tool accesses system design information from the diagnostic model to generate detailed diagnostic assessment reports, and command-line processing enables potential batch mode processing of TEAMS Designer models. The tool also features user-specified report options that include internal source calls and access to system environmental variables – features that enable automation of the previously labor-intensive manipulation of input files. The software generates detailed, readable diagnostic assessment reports that can be viewed in an Internet browser or imported into either Microsoft Word or Excel programs. Procedural C code provides fast, consistent, and efficient processing of the diagnostic model information.

This work was done by Kevin Melcher of Glenn Research Center, and William A. Maul and Christopher Fulton of QinetiQ North America. Further information is contained in a TSP (see page 1).

Inquiries concerning rights for the commercial use of this invention should be addressed to NASA Glenn Research Center, Innovative Partnerships Office, Attn: Steven Fedor, Mail Stop 4–8, 21000 Brookpark Road, Cleveland, Ohio 44135. Refer to LEW-18795-1.

Interactive 3D Mars Visualization

The Interactive 3D Mars Visualization system provides high-performance, immersive visualization of satellite and surface vehicle imagery of Mars. The software can be used in mission operations to provide the most accurate position information for the Mars rovers to date. When integrated into the mission data pipeline, this system allows mission planners to view the location of the rover on Mars to 0.01-meter accuracy with respect to satellite imagery, with dynamic updates to incorporate the latest position information. Given this information so early in the planning process, rover drivers are able to plan more accurate drive activities for the rover than ever before, increasing the execution of science activities significantly. Scientifically, this 3D mapping information puts all of the science analyses to date into geologic context on a daily basis instead of weeks or months, as was the norm prior to this contribution. This allows the science planners to judge the efficacy of their previously executed science observations much more efficiently, and achieve greater science return as a result.

The Interactive 3D Mars surface view is a Mars terrain browsing software interface that encompasses the entire region of exploration for a Mars surface exploration mission. The view is interactive, allowing the user to pan in any direction by clicking and dragging, or to zoom in or out by scrolling the mouse or touchpad. This set currently includes tools for selecting a point of interest, and a ruler tool for displaying the distance between and positions of two points of interest.

The mapping information can be harvested and shared through ubiquitous online mapping tools like Google Mars, NASA WorldWind, and Worldwide Telescope.

This work was done by Mark W. Powell 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-47311.

Rapid Diagnostics of Onboard Sequences

Keeping track of sequences onboard a spacecraft is challenging. When reviewing Event Verification Records (EVRs) of sequence executions on the Mars Exploration Rover (MER), operators often found themselves wondering which version of a named sequence the EVR corresponded to. The lack of this information drastically impacts the operators' diagnostic capabilities as well as their situational awareness with respect to the commands the spacecraft has executed, since the EVRs do not provide argument values or explanatory comments. Having this information immediately available can be instrumental in diagnosing critical events and can significantly enhance the overall safety of the spacecraft.

This software provides auditing capability that can eliminate that uncertainty while diagnosing critical conditions. Furthermore, the Restful interface provides a simple way for sequencing tools to automatically retrieve binary compiled sequence SCMFs (Space Command Message Files) on demand. It also enables developers to change the underlying database, while maintaining the same interface to the existing applications. The logging capabilities are also beneficial to operators when they are trying to recall how they solved a similar problem many days ago: this software enables automatic recovery of SCMF and RML (Robot Markup Language) sequence files directly from the command EVRs, eliminating the need for people to find and validate the corresponding sequences.

To address the lack of auditing capability for sequences onboard a spacecraft during earlier missions, extensive logging support was added on the Mars Science Laboratory (MSL) sequencing server. This server is responsible for generating all MSL binary SCMFs from RML input sequences. The sequencing server logs every SCMF it generates into a MySQL database, as well as the high-level RML file and dictionary name inputs used to create the SCMF. The SCMF is then indexed by a hash value that is automatically included in all command EVRs by the onboard flight software. Second, both the binary SCMF result and the RML input file can be retrieved simply by specifying the hash to a Restful web interface. This interface enables command line tools as well as large sophisticated programs to download the SCMF and RMLs on-demand from the database, enabling a vast array of tools to be built on top of it. One such command line tool can retrieve and display RML files, or annotate a list of EVRs by interleaving them with the original sequence commands.

This software has been integrated with the MSL sequencing pipeline where it will serve sequences useful in diagnostics, debugging, and situational awareness throughout the mission.

This work was done by Thomas W. Starbird, John R. Morris, Khawaja S. Shams, and Mark W. Maimone 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-48080.

MER Telemetry Processor

MERTELEMPROC processes telemetered data in data product format and generates Experiment Data Records (EDRs) for many instruments (HAZCAM, NAVCAM, PANCAM, microscopic imager, Mössbauer spectrometer, APXS, RAT, and EDLCAM) on the Mars Exploration Rover (MER). If the data is compressed, then MERTELEMPROC decompresses the data with an appropriate decompression algorithm. There are two compression algorithms (ICER and LOCO) used in MER. This program fulfills a MER specific need to generate Level 1 products within a 60-second time requirement.

EDRs generated by this program are used by merinverter, marscahv, marsrad, and marsjplstereo to generate higher-level products for the mission operations. MERTELEPROC was the first GDS program to process the data product. Metadata of the data product is in XML format. The software allows user-configurable input parameters, per-product processing (not stream-based processing), and fail-over is allowed if the leading image header is corrupted. It is used within the MER automated pipeline.

MERTELEMPROC is part of the OPGS (Operational Product Generation Subsystem) automated pipeline, which analyzes images returned by in situ spacecraft and creates level 1 products to assist in operations, science, and outreach.

This work was done by Hyun H. Lee 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-47797.

pyam: Python implementation of YaM

pyam is a software development framework with tools for facilitating the rapid development of software in a concurrent software development environment. pyam provides solutions for development challenges associated with software reuse, managing multiple software configurations, developing software product lines, and multiple platform development and build management. pyam uses release-early, release-often development cycles to allow developers to integrate their changes incrementally into the system on a continual basis. It facilitates the creation and merging of branches to support the isolated development of immature software to avoid impacting the stability of the development effort. It uses modules and packages to organize and share software across multiple software products, and uses the concepts of link and work modules to reduce sandbox setup times even when the code-base is large. One side-benefit is the enforcement of a strong module-level encapsulation of a module's functionality and interface. This increases design transparency, system stability, and software reuse.

pyam is written in Python and is organized as a set of utilities on top of the open source SVN software version control package. All development software is organized into a collection of "modules." pyam "packages" are defined as sub-collections of the available modules. Developers can set up private sandboxes for module/package development. All module/package development takes place on private SVN branches. High-level pyam commands support the setup, update, and release of modules and packages. Released and pre-built versions of modules are available to developers. Developers can tailor the source/link module mix for their sandboxes so that new sandboxes (even large ones) can be built up easily and quickly by pointing to pre-existing module releases. All inter-module interfaces are publicly exported via links. A minimal, but uniform, convention is used for building modules.

This work was done by Steven Myint and Abhinandan Jain 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-48447.