

Advances in Distributed Operations and Mission Activity Planning for Mars Surface Exploration

Jason M. Fox^{*} (56652), Jeffrey S. Norris[†], Mark W. Powell[‡], Kenneth J. Rabe[§], Khawaja Shams^{**}
NASA Jet Propulsion Laboratory, Pasadena, CA 91109

A centralized mission activity planning system for any long-term mission, such as the Mars Exploration Rover Mission (MER), is completely infeasible due to budget and geographic constraints. A distributed operations system is key to addressing these constraints; therefore, future system and software engineers must focus on the problem of how to provide a secure, reliable, and distributed mission activity planning system. We will explain how Maestro, the next generation mission activity planning system, with its heavy emphasis on portability and distributed operations has been able to meet these design challenges. MER has been an excellent proving ground for Maestro's new approach to distributed operations. The backend that has been developed for Maestro could benefit many future missions by reducing the cost of centralized operations system architecture.

I. Introduction

A centralized mission planning system for any long-term mission, such as the Mars Exploration Rover Mission (MER), is completely infeasible due to budget and lifestyle constraints. Future system and software engineers must address the problem of how to provide a secure and distributed mission planning system. Distributed operations must be one of the fundamental requirements for the Ground Data System (GDS) architecture.

In the case of MER, the original architecture for the mission planning system was distributed in nature. However, budget cuts did not allow for this capability to be developed. The end result was a set of loosely connected and difficult to monitor scripts functioning to mirror large portions of the GDS at several remote sites. Furthermore, each remote site was required to purchase and maintain very specific hardware in order to accommodate the operation of tools like the Science Activity Planner (SAP) and the Rover Sequencing and Visualization Program (RSVP) that were not developed with distributed operations capability.

We will explain how Maestro, the successor to SAP, with its heavy emphasis on portability and distributed operations has been able to improve upon the costly distributed operations system of MER. Maestro has already been delivered to MER as a replacement for SAP. In addition, Maestro will also be used in a critical mission-planning role for both Phoenix in 2007 and Mars Science Laboratory (MSL) in 2009.

Maestro does not require any costly data replication, nor does it entail the purchasing of special hardware. It relies on a well-designed set of services functioning on highly reliable JPL institutional servers. It can function on almost any modern computer with a network connection, including laptops. When browsing downlink data from the spacecraft, using Maestro is similar to using a web browser. After authenticating the user, it connects to a database server to query an index of data products. It then contacts a web server to download and display the actual data products. Maestro also includes collaboration support based upon a highly reliable messaging system. Modifications made to targets and plans in one instance are quickly and securely transmitted to other instances of Maestro.

^{*} Software Engineer, Planning Software Systems, 301-250D

[†] Planning Software Systems Group Supervisor, Planning Software Systems, 301-250D

[‡] Software Engineer, Robotic Software Systems, 198-219

[§] Software Engineer, Planning Software Systems, 301-250D

^{**} JPL Summer Student, Planning Software Systems

MER has been an excellent proving ground for Maestro's new approach to distributed operations. The backend that has been developed for Maestro could benefit many future missions by reducing the cost of centralized operations system architecture.

II. SAP for MER

The Science Activity Planner¹ (SAP) was the primary science operations tool for the MER mission and co-winner of the NASA Software of the Year Award in 2004. SAP utilized a variety of visualization and planning capabilities to enable the mission operations team to direct the activities of the Spirit and Opportunity rovers. Originally, SAP was designed to support distributed mission operations, but budget pressures forced the descope of all distributed operations features in the tool. Following the successful landing of both rovers and their unexpected long life, it became clear that a distributed operations capability was needed to support the extended mission.

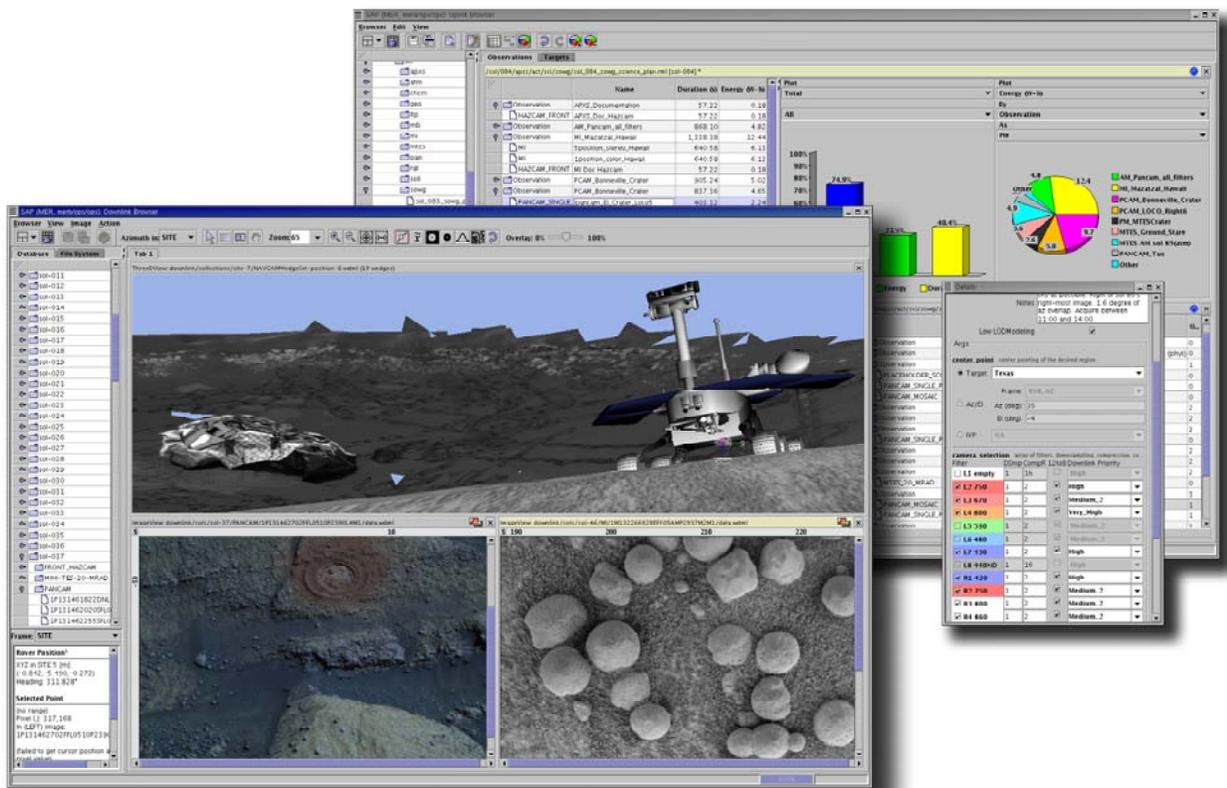


Figure 1. Screenshots of the Science Activity Planner as used by the Mars Exploration Rover Mission.

RemoteSAP² was developed to enable the members of the MER science team to use the SAP tool at their home institutions. More specifically, it was designed to address the following requirements:

- Distribute downlink data (i.e. rover images) to the remote scientists in a timely fashion and make this data available in SAP
- Share labels assigned to surface features among the scientists as they were created
- Facilitate the sharing of files containing the planned scientific observations for the following day

Funding and schedule pressures ruled out the option of making significant modifications to SAP in order to address the above requirements. Instead, RemoteSAP attempted to replicate the mission operations environment that SAP required at each of the remote institutions. JPL Multi-mission Image Processing Lab's (MIPL) File Exchange Interface (FEI) was used to distribute data to each remote site where a set of scripts organized the data in the same way it was organized at JPL. The database that stored targeting information at JPL was automatically synchronized with an external database that served the remote sites. Finally, scripts were provided to enable the retrieval and submission of plan files to/from remote sites.

The results of this approach were mixed. On the positive side, RemoteSAP enabled the mission to transition to distributed operations and supported hundreds of sols of successful operations. The downlink distribution system delivered data to users successfully on most sols, enabling them to analyze the latest data before discussing the plan for the following sol. On the negative side, the system was brittle and difficult to use. Target sharing was slow and unreliable. The plan sharing capabilities often failed, forcing the team to rely primarily on manually transferring files around.

The lessons learned in the development and use of the RemoteSAP system were folded into the design of SAP's successor, Maestro. These issues are summarized below:

- Design operations tools to support distributed operations from the beginning
- Ensure that all users, whether "local" or "remote", use the same tool in the same way
- Eliminate dependencies on shared file systems that won't function across typical firewalls (e.g. NFS)
- Avoid polling-based synchronization in favor of event notification
- Use SQL databases to persist all data that needs to support concurrent editing

III. Maestro Overview

Maestro is a platform independent thin client designed to assist mission operators with the task of downlink analysis and activity planning. One of the main differences between Maestro and its predecessor, SAP, is that Maestro was designed from the beginning to operate in a distributed environment. In other words, Maestro does not have to physically be installed behind the flight operation's firewall in order to function properly. A user can accomplish his/her task with Maestro installed on any computer that has a network connection. To realize this radical new approach to distributed mission operations, Maestro's supporting architecture depends heavily upon robust server side support. These services must be reliable, scalable, secure, and flexible. Fortunately, a readily available and capable open source project already existed for each service that Maestro requires.

A. Relational Database

1. Downlink

Maestro requires access to a relational database. This database maintains a catalog of downlinked data products. Maestro interfaces with the database to execute queries for different types of data products. Maestro makes use of a JPL institutional MySQL instance installed in a clustered environment that provides a secure, flexible, and scalable database. MySQL is also configured to restrict access via IP address. In addition to indexing data products, the database instance is also used to store auxiliary data, such as authorization information, that other services require.

Maestro's use of a relational database for indexing data products is a vast improvement over SAP. When using SAP on MER, the user must navigate the file system structure to search for data products. There is no concept of using query criteria (e.g. instrument, sol number, sequence ID, etc) to whittle down a result set into one containing the desired data products.

2. Planning

Another improvement over SAP is the use of a central planning database. Each Maestro client connects to the same planning database to download and save plans. On MER, SAP users behind the flight firewall all connected to the same file system, also known as the Operations Storage Server or the OSS. However, remote users could not run SAP and connect to the same file system. In order for remote SAP users to view the same plans as local SAP users, an entire data replication system had to be setup where each remote site would mirror the file system at JPL. This approach requires a lot of bandwidth, disk space, and process monitoring. The new system for Maestro eliminates these problems by treating every instance of Maestro in an identical fashion whether the client is local or remote with respect to JPL.

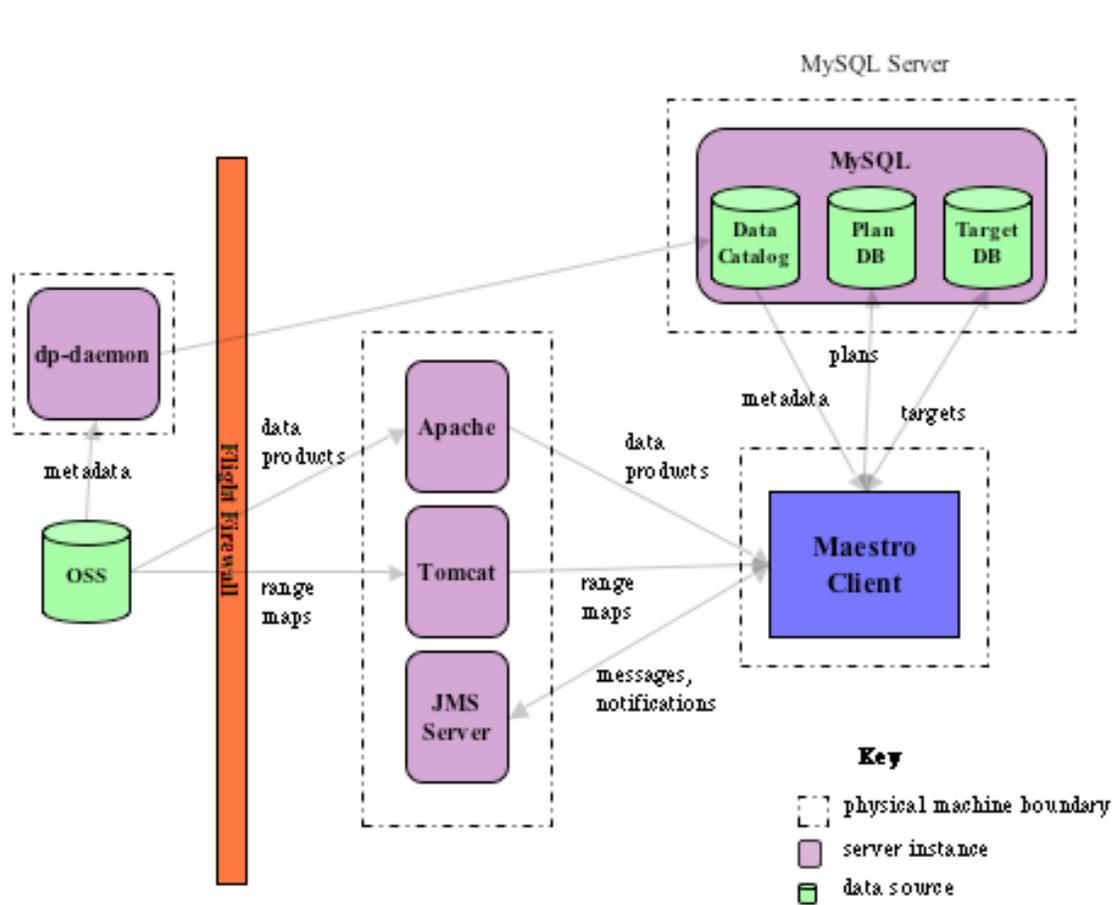


Figure 2. Maestro's server architecture.

B. Web Server

Maestro also requires a connection to a web server that provides access to the cataloged downlinked data. In Maestro's current architecture, the data is first compressed and then cataloged on the web server. Maestro clients download the data via HTTP on an as needed basis. This pull model for data acquisition allows Maestro to be installed on a variety of systems (e.g. laptops) in a variety of locations (e.g. at a scientist's home institution) that may have limited storage space available for data products.

The web server that provides this service to Maestro is the Apache HTTP Server. It's an open source implementation of a web server that is freely available. The Apache server has been configured to require authentication before granting access to the cataloged downlinked data.

In contrast, RemoteSAP relies on MIPL's FEI to deliver data to each remote site. This setup requires that each remote site must be listening (i.e. have its FEI subscription service running) at the time of data delivery. If a remote site is down for any part of the data transfer, its file system will not be completely updated and therefore will be missing some files.

C. Messaging Service

In a distributed, collaborative system such as Maestro; it is crucial that state be synchronized across all the clients. To achieve this synchronization, instances of Maestro must be able to communicate with one other. One mechanism of communication is the central database. In SAP, every instance of SAP polls the database in search of newly created targets. This polling mechanism, while simple and reliable, causes unnecessary network traffic as well as increased load on the database.

The Maestro architecture addresses the synchronization problem through the use of a messaging service to communicate between instances. After successfully authenticating the user, each Maestro instance synchronizes itself with the latest changes from database. In addition, it subscribes to a Java Message

Service (JMS) topic. From this point onward, user modifications to targets and plans leverage JMS to announce the change to other clients.

D. Application Server

One of Maestro's features is the ability to provide the user with range data for image products. Range data (stored in a range map) is generated from stereo image pairs and provides the 3D position for a given image pixel. Range maps are typically large products (approximately 4 MB); therefore, retrieving the entire range map in advance just to provide range information for a single pixel would require an extraneous amount of network traffic and needlessly consume disk space. To address this issue, we built a web application that takes as parameters the image ID and pixel information and returns the range data. Furthermore, the web application caches a region around the pixel requested in order to exploit the locality in the request parameters. The regions are saved in an LRU based cache. This caching strategy reduces the amount of disk I/O and increases the performance and scalability of our application.

E. Security

For MER, the Maestro servers are currently running on a stripped down version of Solaris, with nothing more installed than is absolutely necessary. These servers are maintained by the Information Technology group at JPL and undergo constant monitoring. All the services require authentication and allow differing levels of authorization. Moreover, the passwords are never transmitted in plain text.

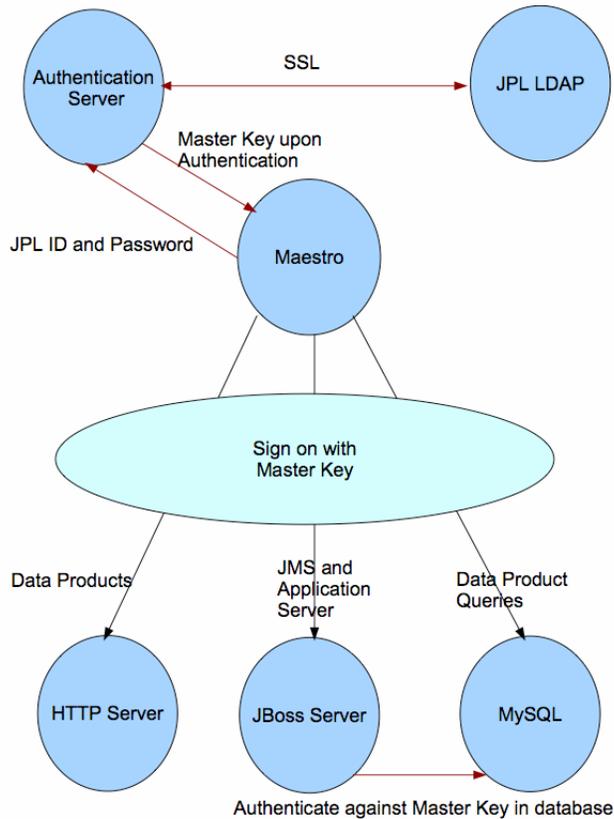


Figure 3. Maestro's security scheme.

1. Authentication

It would be overly troublesome to require our users to remember unique passwords for each of the distributed services. It would be a complete mess if Maestro required its users to sign in for each service separately. The current authentication scheme requires a single sign in to gain access to all of the services that Maestro requires. Users sign in using their JPL username and password. The authentication is sent over an SSL encrypted channel to the Apache server, which then authenticates the user against the JPL LDAP

interface. Upon successful authentication, the Apache server returns to Maestro a username and password that it can use to access all of the other services. The password can be changed at anytime, and since Maestro checks for this password at startup, all clients will transparently update themselves with the latest authentication information.

3. *Authorization*

For JMS and the web application, we are using the JBoss application server. JBoss has an excellent implementation of role based security. Furthermore, JBoss authenticates users against a database, so passwords and roles can be updated without restarting JBoss. For JMS, we can maintain flexible role based authorization for each topic, choosing to give a few of the users publish access, while providing others with just the subscribe access. This feature allows us to have multiple topics on the same JBoss instance.

4. *Password Exchange*

For performance reasons, we wanted a mechanism that would allow us to authenticate in an encrypted format but send the data through a clear text channel. MySQL's authentication scheme uses a challenge response algorithm, which allows us to communicate with the database over clear text for everything besides the authentication. For our web application server and the HTTP web server, we use HTTP Digest authentication, which allows us to accomplish the same goal.

IV. Downlink

A. Downlink Assessment

Contextual awareness is a requirement for planning the activities of any spacecraft. For a mobile surface-based spacecraft like MER, the context for planning can change every day. Indeed, it is the rover's traversal through and in situ interactions with its environment that require that a new activity plan is created and uplinked at frequent intervals (once every 1-3 sols, where plans covering more than one sol are usually restricted from driving and in situ placement activities). After a rover traverse or an in situ instrument placement, the next task is three-fold:

1. Assess the quality of the new science data
2. Understand the state of the rover
3. Identify and prioritize targets of opportunity in the locale

The Maestro tool affords a scientist the ability to quickly and easily query the data of relevance to the task of planning for the next sol of operations. Data products such as images are automatically cataloged according to their metadata (data associated with the image). The catalog of data is designed for remote browsing of products to better support a team of scientists who are geographically separated.

B. Cataloging Data

As MER data products are downlinked from the spacecraft, they are processed via the Operations Product Generation Subsystem (OPGS) pipeline into useful products for operations. The pipeline brings together information from different sources to construct more complete products. For example, in addition to the actual images and science data captured by the instrument payload, metadata such as time of acquisition, rover position and attitude, state of the instrument in question such as temperature, etc. are used to assess the quality of the data. These data and metadata are combined into an Experiment Data Record (EDR). An EDR records the data in its original form, just as an instrument acquired it. Often it is necessary to calibrate this original data (e.g. to account for lighting conditions), or re-process it into other useful data products, such as creating XYZ position maps from a stereo image pair. Such derived products are called Reduced Data Records or RDRs.

Once new EDRs and RDRs are generated by the pipeline they are published onto a file system that is organized first by sol, then by instrument, and lastly by product type. Although the pipeline is designed to publish these products to a central repository, distributed planning requires browsing the data by remote access. We have added a relational database-driven catalog (MySQL 5) on top of the file repository to enable remote browsing and a web server (Apache) to enable remote access to the data files. New EDRs are added to the database along with metadata from the product to use as search criteria. In practice the complete metadata for a product is very large (hundreds of items), some of which is very repetitive from product to product. We therefore down selected a subset of the metadata to be used as optional search criteria in the data catalog. The metadata fields that are included in the catalog are:

Field name (from PDS Header)	Description
PLANET_DAY_NUMBER	The sol when the data product was acquired
PRODUCT_ID	The unique identifier of a data product
INSTRUMENT_ID	The unique identifier of the instrument that acquired the data
FRAME_ID	For stereo camera, which camera (left or right) was used to capture the image
PRODUCT_TYPE	The product type, for example for an image one of EFF (full frame), ESF (subframed) or EDN (downsampled)
ROVER_MOTION_COUNTER	A 5-tuple that uniquely identify the rover's location
PMA ARTICULATION_STATE	The azimuth and elevation angles of the mast head as measured by the encoders
SEQUENCE_ID	The sequence ID associated with the command that acquired the data
SPACECRAFT_CLOCK_START_COUNT	The spacecraft clock start time of data acquisition at inception
SPACECRAFT_CLOCK_STOP_COUNT	The spacecraft clock start time of data acquisition at completion

In addition to cataloging each individual data product, we also group together related data products in an intuitive fashion. For example, for a stereo imaging activity we group together the left and right image for viewing as a product set, which lets the Maestro Image View easily switch between the images at the request of the user. For Pancam observations, we group together all of the individually acquired color filters into a set and provide an approximate-color composite view. We also perform a higher level of grouping to combine a set of imaging observations that comprise a mosaic into an image collection. Maestro is capable of downloading the entire collection of Navcam or Pancam images from a particular rover location and mosaicking them together on the fly to give the user a complete 360-degree view that is useful for general contextual awareness.

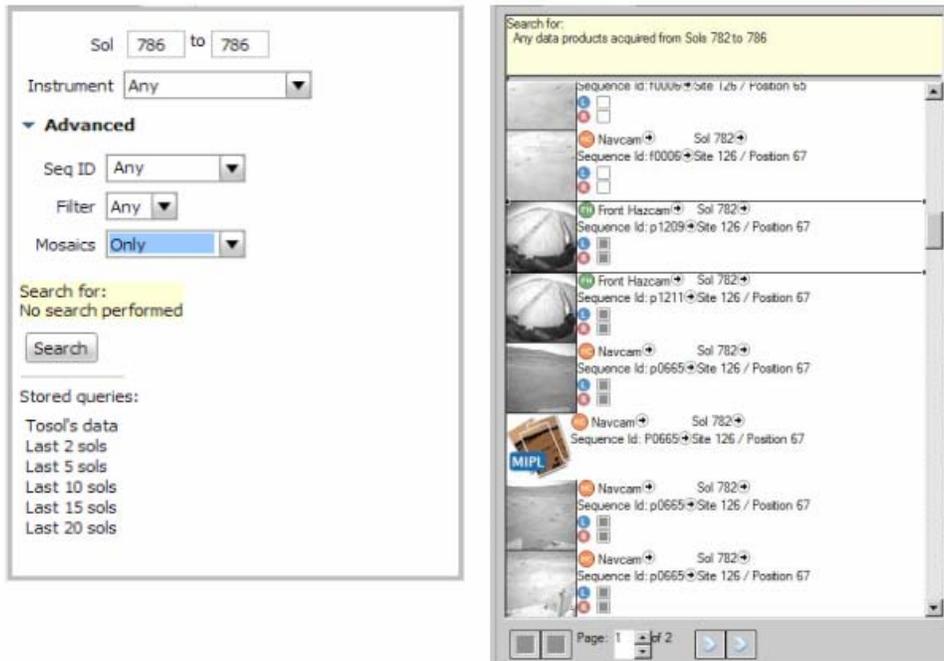


Figure 4. Query view and query results view.

Using the catalog, a user can quickly browse the recently acquired products using any of the available search criteria like sol, instrument, sequence ID, etc. There are also stored queries for frequently used searches like "Tosol's data", "Last 2 sols", and "Last 5 sols". When a query is run and the results are displayed in Maestro, controls are provided to further narrow the search and to page through the results if many products match the user's given criteria. For image products, thumbnail images are included in the results to give a preview of the content that is displayed in full if the user opens the product in an Image View.

Thumbnail images and full image products are accessed through Maestro's remote connection to an Apache web server. The downloaded products are stored on the user's laptop or workstation on the local disk in a cache directory so that subsequent access requests to the product file do not incur a network transfer delay. The cache has a maximum capacity that is user-configurable, and when new files are needed that exceed the cache capacity, older files are deleted on a least-recently-used basis.

The image products are compressed on the server side to reduce the amount of network traffic during product downloads. A compression utility creates a JPEG-compressed version of the product with the same PDS-format metadata header. The resulting file is deposited on the file system where the Apache server can provide remote access to the data. Using JPEG compression, the same products are reduced in size by a factor of 14 resulting in a dramatic reduction of required bandwidth.

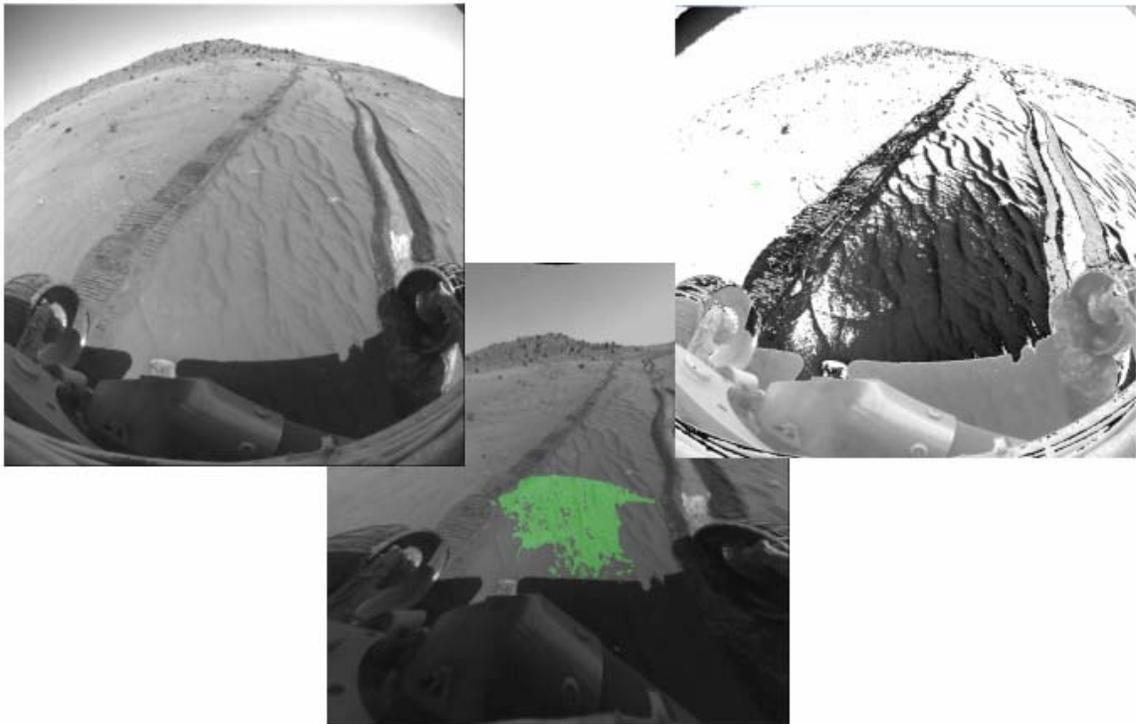


Figure 5. Original EDR, RDR with IDD overlay, RDR with contrast enhancement.

Different operations products may use different compression strategies based on their usage patterns. Reachability maps are one example of a highly compressible product type given its usage pattern. The IDD reachability map has information about which pixels in a left Front Hazcam image correspond to a viable in situ arm instrument target position. To generate this product, every position in the image is evaluated using an arm simulation that attempts to place a particular instrument at that position without creating any arm collisions with itself or the surface. The reachability map contains a wealth of information that is useful to the engineer who evaluates the safety of an IDD placement or the rover planner who designs the command sequence to execute placement, yet the usage pattern of the scientist is far simpler: they require a binary answer, can I place this instrument or not? This simplification creates an excellent opportunity for compression. Since the arm workspace is a contiguous region in the low-center of the Front Hazcam field of view, the data is highly self-similar and compresses well with a simple Huffman encoding strategy. The

compressed version in PNG format using Huffman (lossless) encoding is on average three orders of magnitude smaller than the original product. In Maestro, this product can be overlaid onto the Front Hazcam image of the arm workspace, and the user can select which instruments for which they need to assess reachability. The reachable area is overlaid in green onto the Hazcam image, indicating that placement is possible within the region.

Another example of product types that can be distributed efficiently by exploiting their usage pattern is range maps and surface normal maps. For stereo image pairs, the 3D location and surface normal (orientation vector) are computed for every pixel in the left image. Distributing anywhere from 10 to 100 of these large product sets per sol would be a time-consuming and bandwidth-intensive process if they were to be transferred to every client in full. The operational access pattern comes to our aid again here, however. Generally speaking, the driver for science target selection is its visual appearance in the image, such as its shape, texture, brightness, or morphology. The fact that a 3D position is required for the safe and effective placement of an in situ instrument or the pointing of a mast instrument (in some cases) is largely irrelevant for science target designation. Therefore, we do not distribute the entire range map and surface normal map products to each user. Instead, we allow access to the data in these products on a pixel-by-pixel basis by running ad-hoc requests to a server side process (servlet) running on a Tomcat server. The servlet is designed to access range map and surface normal map product files using random access I/O requests that seek directly to a position relative to the beginning of the file to access the numeric information in a 128 by 128 pixel region around the requested pixel location. All of the position and normal data in the region are read and stored in an LRU cache in the local memory of the servlet process. Frequently, many users will be accessing the same products (recently generated Hazcam, Navcam, and sometimes Pancam range and normal products) and after the regions of data are read from these products once, subsequent access requests for this information never incur a disk access but are returned directly from the in-memory cache. In practice the implementation of this technique results in a round-trip time of 1-2 seconds from initial request to server response and ~500 ms for repeat queries.

V. Uplink

The distribution of downlink products is only one aspect of a complete distributed mission activity planning system. Another equally important facet is an uplink planning system capable of enabling collaboration among a geographically distributed network of scientists. Every instance of Maestro must be able to communicate securely and efficiently with every other instance of Maestro. For example, scientists want to be able to locally specify a target (a point of interest) in an image and have that target be automatically available globally to every other scientist using Maestro. Designing collaboration support that functions in a distributed environment poses some unique challenges.

A central database is one of the necessary components for ensuring synchronization and accessibility for all Maestro clients. All Maestro clients connect to the same database. This connection allows clients to easily share plans and targets regardless of geographic location. The central database is only part of the equation, however. In order to maintain synchronization without employing an inefficient polling technique, a messaging service is used. In the case of Maestro, we use JMS.

C. JMS and Message Ordering

The JMS specification does not guarantee a message order for messages sent from multiple sessions³. However, Maestro must ensure that it receives all JMS messages in the intended order. For example, what should Maestro do in the case where it receives a message to delete a target before it even receives the message to create that target? If not handled properly, this situation could cause some Maestro clients to become out of sync.

To avoid this problem, Maestro implements a simple message ordering layer on top of the JMS layer. This message-ordering layer is a completely generic, configurable component that is only concerned with a single integer value in the message's header—the message payload is inconsequential at this layer. The ordering layer uses the integer value in the header to determine global message order. However, if an unexpected message number arrives the ordering layer must be smart enough to timeout on the particular message it had been expecting and move on to the next available message.

D. Protocol Layer

Modifications to the central database must be handled in a very delicate manner. As a result, the protocol for updating the database incorporates JMS to ensure that all clients remain in perfect synchronization. Clients must remain in sync even if one crashes while attempting to publish a modification to the database.

In the case of synchronizing target information, the database has three responsibilities. First, it stores the actual target data. Next, it keeps a record of recent target modifications. Finally, it maintains a global counter for the purpose of assigning monotonically increasing ordering numbers to JMS messages. To edit or create a target, the Maestro client updates the database within a single transaction, and then broadcasts a JMS message to notify other Maestro clients about the target in question.

Due to the monotonically increasing JMS message numbers, clients will be expecting to receive a sequential list of change broadcasts. If this constraint is violated because a client crashes after updating the database but before publishing to JMS, then the database will contain a history of recently modified targets. A client can then ask the database to retrieve the targets for any messages it had expected to receive. There is also the case where a client attempts a modification and finds the next message number is not the one it had been expecting. This case is handled in a similar fashion.

E. Plan Sharing

A similar JMS / database protocol is employed to achieve plan sharing. In this mode, users can launch Maestro, load a plan, and then watch edits being performed on the plan by another user. This scenario is useful for the remote scientists that wish to follow plan modifications made during the Science Operations Working Group (SOWG) meeting.

Another scenario occurs during the plan refinement process. The distributed plan editing capabilities in Maestro provide several instrument specialists with the ability to edit the same plan simultaneously. This capability allows for parallelism in the activity planning process, therefore reducing the amount of time personnel are required to be on shift.

VI. Future Work

Maestro is by no means in a complete state. The distributed operation capabilities are sufficient to begin the process of replacing SAP on the MER mission. But future mission will no doubt require further enhancements to its already rich feature set.

A. Additional Server Dependencies

There are additional server dependencies that will be added for future missions such as Phoenix (2007) and MSL (2009). The Phoenix mission has decided to use a JPL plan generation and modeling tool called APGEN⁴. However, Phoenix will not be using the traditional GUI that APGEN provides. Instead, it will use a headless version of the software running as a server. Maestro's connection to this server will provide it with additional modeling capabilities, but will also present new challenges since this will be the first mission to use the new headless version of APGEN. In addition, MSL and Phoenix have both decided to use an Ames constraint engine and modeling tool called Europa. Europa will also be available as a server to Maestro, thus increasing the number and complexity of server interactions.

With the ever-growing number of server dependencies, it will be increasingly imperative to provide feedback to the user regarding the current status of all server connections. Furthermore, the tool should be designed to handle server disconnects. For instance, instead of crashing or behaving in an unpredictable manner whenever a server connection fails, the tool's functionality should gracefully degrade while at the same time explaining to the user the exact reason for the loss of functionality.

B. Offline Support

Several users have requested the ability to run Maestro when a network connection is not available. Currently, the tool requires a network connection from start to finish in order to function properly. However, one could envision a mode of Maestro that works in a very limited sense when not connected. In the same way that a web browser can be used in an offline mode to view previously downloaded web sites, Maestro could also be used to view the contents of its data product cache.

C. Better Data Discovery

Currently the only mechanism for discovering new data products that have been delivered to the flight file system is via a polling script. This script detects new products and then populates Maestro's data product catalog. However, this polling method is not always 100% reliable. Future missions, specifically MSL, have already baselined a common mission message bus that will be used to announce the arrival of new data products. This message bus will allow us to remove our polling requirements and provide a much cleaner and more efficient offline data product processor.

VII. Conclusion

A centralized mission activity planning system for any long-term mission is simply not feasible. A distributed system is much more flexible while at the same time more cost effective. Maestro, a tool to assist mission operators with the task of downlink analysis and activity planning, was designed from the beginning to operate in a distributed environment. The lessons learned in the development and use of the RemoteSAP system for MER were folded into its design:

- Design operations tools to support distributed operations from the beginning.
- Ensure that all users, whether "local" or "remote", use the same tool in the same way.
- Eliminate dependencies on shared file systems that won't function across typical firewalls (i.e. NFS)
- Avoid polling-based synchronization in favor of event notification
- Use SQL databases to persist all data that needs to support concurrent editing

Maestro has already been delivered to the MER project as a replacement for SAP. Its features as well as the backend server support are undergoing constant improvement. The current work will benefit many future missions by reducing the cost of centralized operations system architecture.

References

- ¹Norris, J., Powell, M., Vona, M., Backes, P., Wick, J., *Mars Exploration Rover Operations with the Science Activity Planner*, IEEE International Conference on Robotics and Automation, April 2005.
- ²Wick, J., Callas, J., Norris, J., Powell, M., Vona, M., *Distributed Operations for Mars Exploration Rover Mission with the Science Activity Planner*, IEEE Aerospace Conference, 2005
- ³Java Message Service Specification, Version 1.1, Sun Microsystems, April 12, 2002
- ⁴Wissler, S., Maldague, P., Rocca, J., Seybold, C., *Deep Impact Sequence Planning Using Multi-Mission Adaptable Planning Tools With Integrated Spacecraft Models*, SpaceOps Conference, 2006