

Virtual Mission Operation Framework^{1,2}

Affiliated

Meemong Lee, Richard J. Weidner
Jet Propulsion Laboratory
4800 Oak Grove, Pasadena, CA 91109
818-354-2228
meemong.lee@jpl.nasa.gov,
richard.weidner@jpl.nasa.gov,

Abstract— The Virtual Mission Operation Framework (VMOF) is one of the project lifecycle engineering process improvement efforts initiated by the institutional technology infrastructure program at JPL. The VMOF is composed of three frameworks: a model integration framework, a simulation framework, and a visualization framework. The model integration framework interfaces with spacecraft system design, mission design, and structure design. The simulation framework interfaces with the operation scenario design, environmental phenomena science, and science payload system design. The visualization framework interfaces with the flight system testbed, the ground system, and the science analysis. The three frameworks of the VMOF collaborate to create a comprehensive virtual mission operation that enables a “validation-in-the-loop” system design process and lifecycle-continuous science-return validation. This paper discusses the technical approaches for each framework implementation, challenges and approaches involved in streamlining mission information access, and on-going activities toward enabling Model-Based Engineering Design in a collaborative distributed environment.

TABLE OF CONTENTS

1. INTRODUCTION.....	1
2. MODEL INTEGRATION FRAMEWORK	3
3. SIMULATION FRAMEWORK	5
4. VISUALIZATION FRAMEWORK	7
5. FUTURE DIRECTION.....	9
REFERENCES.....	11

1. INTRODUCTION

Mission operation, as the last phase of the project lifecycle, must cope with all of the mission system defects that have been introduced during the previous lifecycle phases. Any late detection of design defects or design deviations increases operation cost, compromises science return, and can jeopardize ultimate mission success. In

order to prevent such disastrous consequences, operability must become an integral part of the system design, and operability validation must be an integral part of the lifecycle engineering process.

The Virtual Mission Operation Framework (VMOF) is one of the project lifecycle engineering process improvement efforts at Jet Propulsion Laboratory. Virtual Mission Operation refers to operation-phase activity modeling and simulation that enables operation planning, command sequence generation and validation, telemetry data processing, and engineering and science information analysis. The potential benefits of virtual mission operation are well recognized for all phases of mission lifecycle: early detection of design defects, accurate analysis of the impacts of design deviations, ground data system verification, operation training, and science-return optimization. The goal of VMOF is to facilitate lifecycle-continuous virtual mission operation capability so that the benefits can be fully realized by a wide range of flight projects. The VMOF implementation has been successfully demonstrated for science-return validation during the early design phase. The future development will address rapid and flexible lifecycle tracking of the spacecraft system properties and lifecycle-wide traceability of the operation deficiencies.

As shown in Figure 1, the three frameworks of the VMOF are loosely connected involving three programs, Model Object Builder (MOB), Science Activity Reasoning and Optimizer (SCENAREO), and Time-based Spacecraft Operation State Simulator (TSOS). The MOB is responsible for generating the subsystem property scripts from the Team-X (the spacecraft system design team at JPL) design parameter database. The simulation framework integrates the SCENAREO and the TSOS where the SCENAREO generates the subsystem command sequences and the TSOS executes them on a set of virtual subsystem prototypes and generate operation state files. The visualization framework integrates a telemetry data server and a set of subsystem state visualization clients on a PC-cluster and simultaneously displays multiple subsystem states.

¹ 0-7803-8155-6/04/\$17.00© 2004 IEEE

² IEEEAC paper #1136, Version 1, Updated Nov 7, 2003

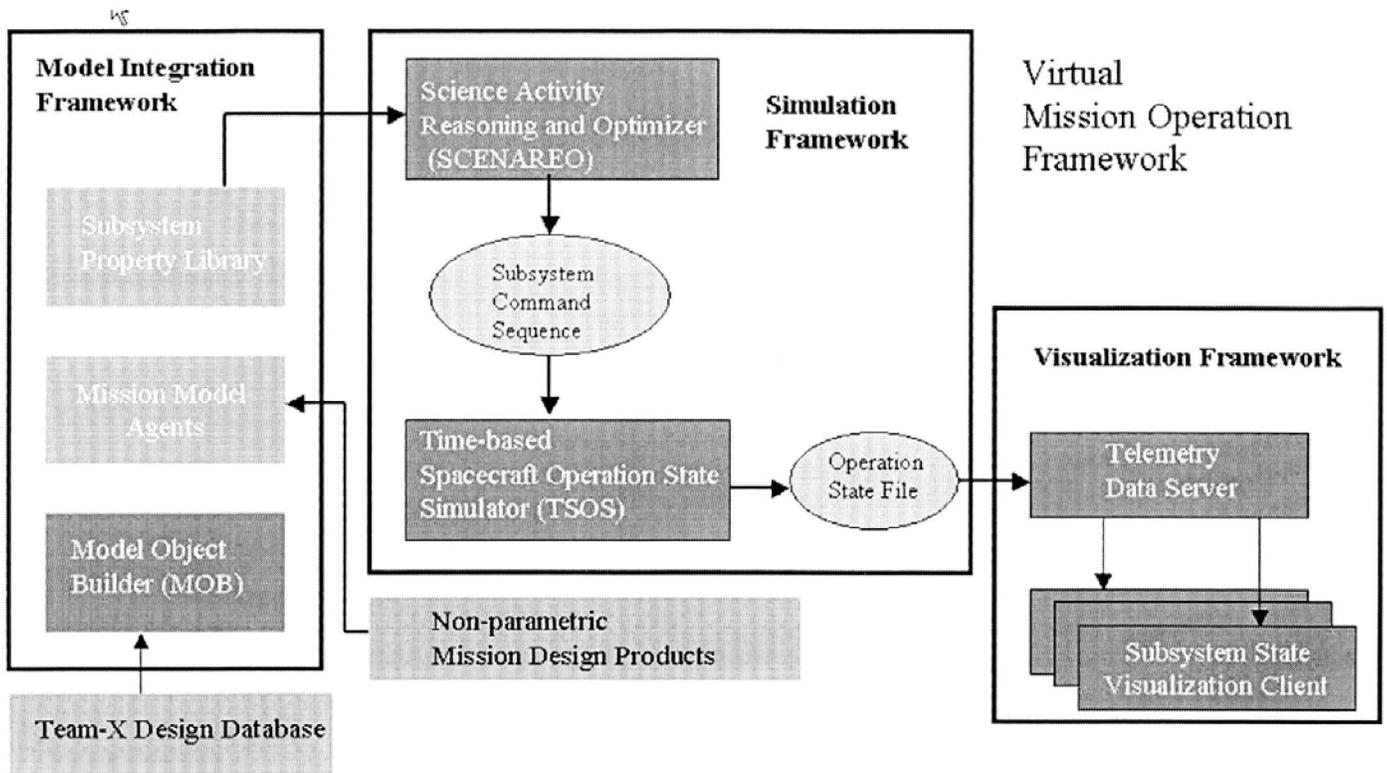


Figure 1 Virtual Mission Operation Framework Architecture

2. MODEL INTEGRATION FRAMEWORK (MIF)

The major goals of the model integration framework are: 1) automated design information capture, 2) distributed discipline model integration, and 3) lifecycle-continuous mission information tracking. Automatic design information capture is implemented for parametric design information employing the Excel-based parameter registration and subscription mechanism. For the non-parametric design information, a set of mission model agents are employed for processing the design information and formulating the VMOF models [3].

Distributed discipline model integration is pursued to develop high-fidelity performance models. Traditionally, the performance of a subsystem is independently analyzed during the operation phase based on the engineering data received from the spacecraft. In the VMOF, these tools can be utilized during the design phase with the simulated engineering data that describe the detailed operation scenario. Currently, the Model Integration Framework interfaces to three multi-mission analysis tools: telecom, power, and instrument.

Lifecycle-continuous mission information tracking is being addressed via a progressive subsystem property data dictionary. The progressiveness of the subsystem property information is tracked for model resolution and

uncertainty range. This section discusses the three components of the Model Integration Framework: MOB implementation for building parametric subsystem property scripts, the subsystem model library that captures static and dynamic performance properties, and mission model agents for non-parametric model integration.

MOB

The MOB provides multiple types of Excel-based user interface (UI) designs for capturing design information. Each UI type is composed of an Excel worksheet and a visual basic program that automatically generates a script file from the work sheet. The VMOF organizes the design parameters into six groups: mission, science target, attitude and articulation control system (ACS), instrument, power, and telecom. Each parameter group is described below with respect to its content and usage. Current implementation of MOB subscribes the parameters specified in the Excel-UI from the IceMaker that manages the parametric design databases of Team-X [1, 3].

The mission UI provides entries for orbit-mission type and encounter-mission type. Both mission types can be specified with a set of type-specific mission parameters or the name of a pre-composed spacecraft trajectory kernel file. The mission information is used to create a virtual navigation system that can compute the position and velocity of the spacecraft relative to a specified reference body. The science target UI is designed to capture the target information including geometric shape, orbit

dynamics, and radiometric properties. This UI is used when the science target is an asteroid or a comet since the current PCK (Physics Constant Kernel) does not provide the above information for asteroids and comets. The science target information is used to create a virtual target that can compute its position and velocity relative to the Sun. The virtual target is also used to determine instrument operation including exposure duration (i.e., surface brightness), footprint analysis, etc.

The Attitude and Articulation Control System (AACS) UI is designed to acquire turn-related performance characteristics for the spacecraft and attached devices, such as solar panels, antennas, and scan platforms. For spacecraft, attitude control related parameters are captured including turn acceleration (delta-V) range, stabilization profile, point control error, and attitude knowledge error. For each articulated device, gimbal parameters are captured. The AACS information is used to create a virtual AACS that can compute the turn duration and pointing accuracy as shown in Figure 2.

10 instruments can be specified. The instrument UI also provides entries for structural information that are related to observation planning, such as, position and orientation of the instrument relative to the spacecraft center, gimbal structure, etc. The instrument information is used to create virtual instruments that can compute science-data-acquisition-related operation parameter settings.

The power UI captures information about the solar panel, batteries, and power load. The MOB creates a configuration file for the MMPAT (Multi-Mission Power Analysis Tool) based on the captured information and attaches the configuration file name in the power subsystem model. The configuration file is utilized to initialize MMPAT during the spacecraft operation simulation [4].

The telecom UI captures information about the antennas and telemetry information system. For each antenna, the MOB composes a link-budget table interacting with a remote telecom performance analysis system. The MOB

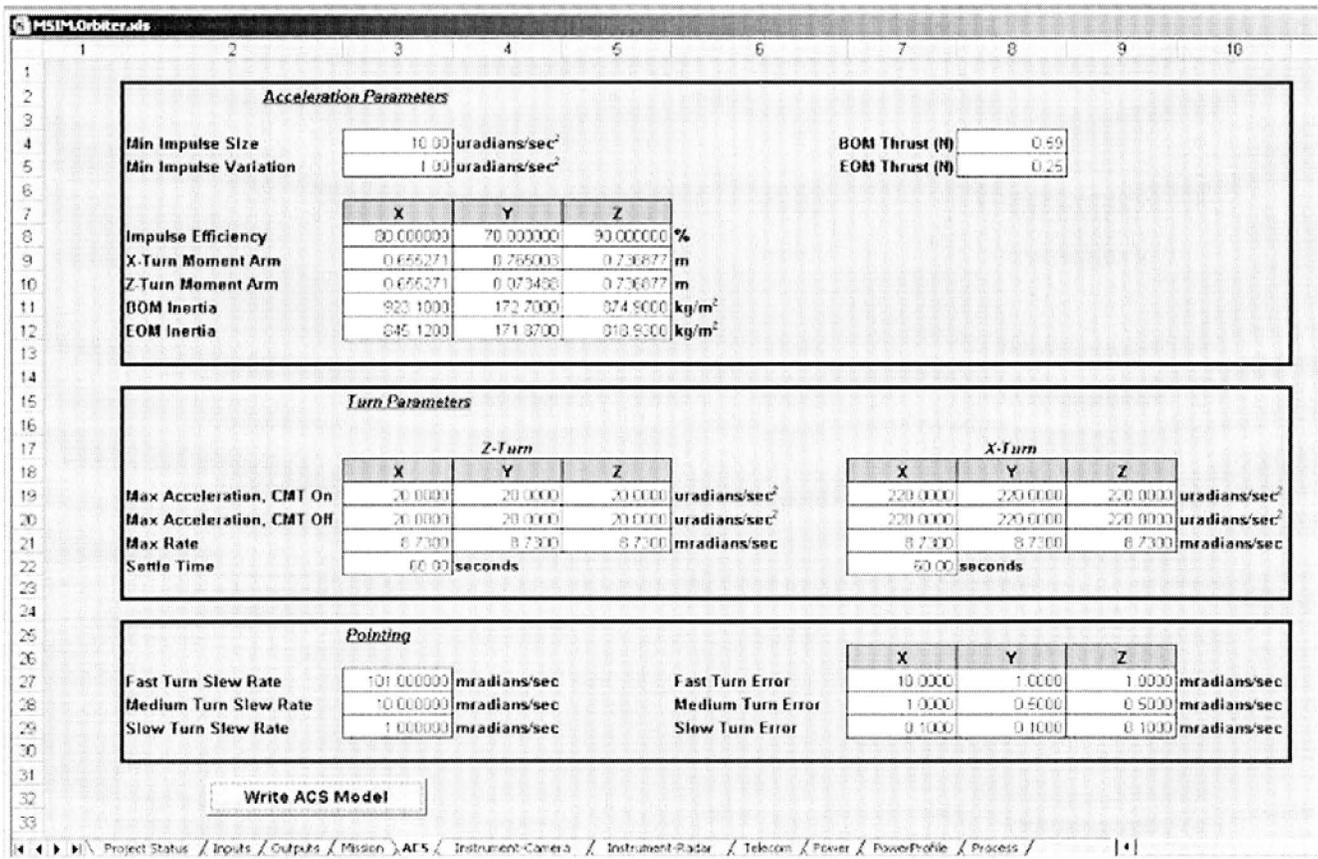


Figure 2 Excel-based ACS UI worksheet

The instrument UI is organized for the optical and non-optical instrument types. For each instrument type, up to

interacts with the TFP (Telecom Forecast and Prediction) system [7] developed at JPL for the missions utilizing the

Deep Space Network (DSN). For missions utilizing non-DSN ground stations, the MOB plans to utilize the Communication module of the Space Tool Kit (STK).

Subsystem Model Library

Based on the captured design information, the MIF creates five subsystem models, a navigation system, a payload system, an attitude and articulation control system (AACS), a power system, and a telecom system. Each subsystem model is implemented as two C++ classes, S-subsystem class and M-subsystem class. The S-subsystem class provides a constructor that parses a subsystem property script generated by the MOB and organizes the parsed parameters into a data structure and a set of member functions for accessing the parameters. Unit checking and conversion are performed automatically.

The M-subsystem class is implemented as a derived class of the corresponding S-Subsystem, and it is used to build subsystem-specific analysis models. The subsystem analysis models are used to predict operation planning related information, such as target visibility, observation time range, and turn duration. The analysis models provide a set of functions that are necessary for operation scenario planning, such as predicted performance at a specific time or under a specific operational condition. The performance prediction is performed interacting with other subsystem models and environment models.

As Figure 2 shows, the S-Mission class is used to derive M-SC and M-Navigation where the M-SC represents the real spacecraft state and the M-navigation represents the estimated spacecraft state. The M-AACS, M-Payload, M-Power, and M-Telecom are created as derived classes of the S-AACS, S-Payload, S-Power, and S-Telecom. The S-Process and M-Process are under development. Each M-subsystem may utilize external models for high fidelity analysis. For example the M-Power links with MMPAT and the M-Telecom utilizes link-budget table composed off-line.

Mission Model Agents

The non-parametric models, such as trajectory models, structure models, and environment models must be able to handle data files created by mission design tools, CAD tools, and science analysis tools. The combination of datasets from multiple sources is often required to form particular information. Polymorphic functionality is required that can recognize different forms and transform them into a desired product. The technical details of the intelligent mission model agents are presented in the paper "Component-based Implementation of Agents and Brokers for Design Coordination" [4].

The intelligent agent is capable of understanding and interacting with its environment on the behalf of its user,

of moving about the web, and of forming and executing rudimentary decisions. The information agent delivers useful information to the user by actively performing search, access, and retrieval of relevant information. As Figure 3 shows, the MOB integrates the non-parametric mission system design information through a set of intelligent mission model agents. Currently, four types of mission model agents have been implemented, target agents, trajectory agents, structure agents, and subsystem agents.

The target agents serve target information phenomena that are relevant to mission operation. For a specified mission type, encounter, orbit, or surface, and a target system, planet, satellite, asteroid, or comet, a specific target agent is required that can acquire appropriate databases and extract desired phenomena information. Currently available target agents include the PCK agent for physical constant information of the solar system, the Viking Mosaic agent for handling multi-resolution surface maps of Mars, and the Star agent for handling star catalogs.

The trajectory agents provide the state of the spacecraft (i.e., position and velocity vectors) with respect to a specified time reference system (e.g., J2000) and a target reference system. For example, state vectors are often propagated with respect to the Solar Barycenter. The position relative to a target then requires translating the vector using the ephemerides of the target. Multiple sources are used for the cruise data and the target ephemerides. Currently, there are two types of trajectory agents, the SPK agent for handling the trajectory files in SPICE format developed by the Navigation Ancillary Information Facility (NAIF) at JPL, and the STK agent for handling orbit parameters specified in the Space Tool Kit (STK).

The structure agents are employed for interfacing with the computer-aided design (CAD) files. A structure agent needs to be implemented to handle a specific CAD tool since each CAD tool employs a proprietary data structure and provides unique export mechanisms. Each structure agent is required to translate the native CAD model to a virtual reality modeling language (VRML) model that a spacecraft visualization client can interact with. The primary goal of the structure agents is to provide object-oriented structure models that can be easily accessed by multiple discipline analysis tools. A major challenge in implementing a structure agent has been the labor-intensive process involved in the simplification of the CAD models. Currently, three structure agents are being developed to interface with Maya by Wavefront, Adaptive Modeling Language (AML) by TechnoSoft and the Unigraphics-NX by Electronics Data Systems respectively.

The subsystem agents provide interfaces to remote subsystem-discipline analysis tools. Currently, three subsystem performance modeling tools are interfaced via

the subsystem agents: TFP for telecom, MMPAT for power, and SceneGen for Instrument. An effort to unify the interface protocol among the subsystems is under way by employing object-oriented middle-ware. The middle-ware offers three functions: request generation, parameter space exploration, and performance table generation. In order to collaborate with various analysis tools, the middle-ware defines a subsystem-specific request protocol in XML (Extensible Meta Language) and utilizes a remote procedure call (RPC) mechanism to send requests.

captures the operation state of the spacecraft. The Simulation Framework combined with the Model Integration Framework enables “validation-in-the-loop design” process and provides the required performance range to achieve the desired science-return to guide the design trade-space analysis.

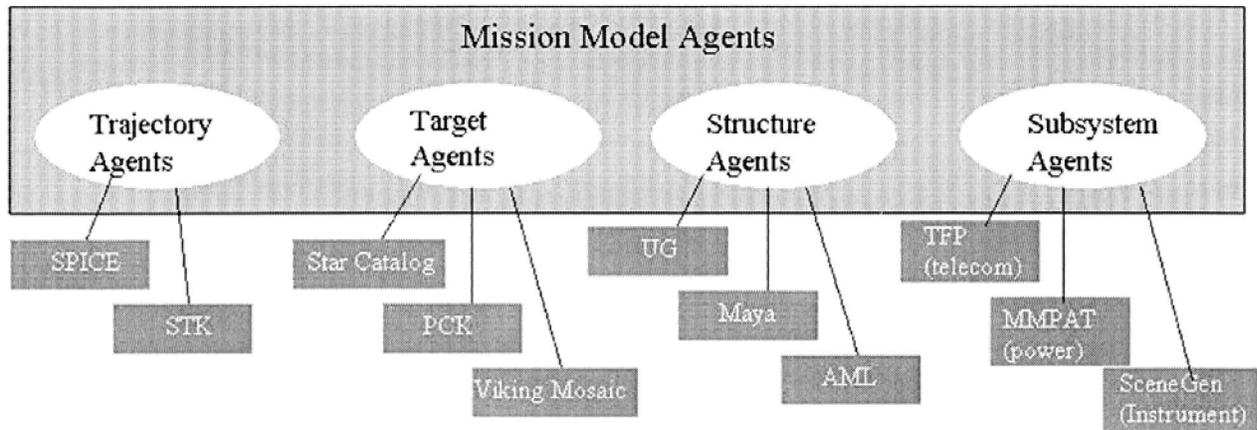


Figure 3 Mission Model Agents and Application Interfaces

3. SIMULATION FRAMEWORK (SF)

The operation phase is composed of on-board processing and ground processing. The ground processing is divided into uplink process and downlink process. The uplink process includes activity planning, command sequence generation, and command sequence validation. The activity planning defines activities and develops activity-level resource usage models. The command sequence generation defines content of the command dictionary and develops activity to command translation mechanism. Finally, the command sequence validation defines flight rules and develops command-level system state models for flight rule validation.

The on-board processing of a spacecraft can be divided into system functions and subsystem functions. The system functions include command sequence management, fault recovery, and telemetry generation. The subsystem functions include command and data handling of individual subsystem.

In the Simulation Framework, the SCENAREO performs the uplink process and the TSOS simulates the on-board processing and generates virtual telemetry file that

SCENAREO

The SCENAREO performs the uplink process in three stages: 1) user interface design for observation objective capturing, 2) preliminary performance/resource analysis and 3) command sequence generation. The Science Objective UI is to interface with the science community to capture the desired science data products in terms of quantity and quality. In addition to the data products, downlink of the acquired data is an important feature when the resource must be shared between observation and downlink. Figure 4 illustrates a science objective specification UI for an orbit mission. The UI divides the observation activities into three groups: surface mapping, target observation, and downlink. For each activity group, orbit allocation and device allocation areas are specified for gathering resource sharing, operational constraints, and observation conditions.

The UI is automatically populated based on the design information captured by the MOB. All of the available instruments are listed both in the surface mapping and target observation area with a default operation mode. All of the antennas are listed in the downlink area. Per observation activity, a user may select orbits, instruments, and antennas for each activity group.

For the surface mapping activity, the surface condition, mapping duration, and off-boresight pointing angle can be

specified for controlling the data quality and volume. For the target observation, search area, surface condition, look angle, multi-frame mosaic option, and exposure duration can be specified. For the downlink operation, antenna pointing condition, data volume, and ground station can be specified for resource sharing and downlink opportunity assessment.

The desired observation specifications described above have direct relationship to the mission parameters and the subsystem properties. The preliminary analysis is to verify the observation specifications against the subsystem properties and provide instantaneous report on any performance or resource conflicts. The conflicts may be resolved by changing the subsystem design or the science-return requirements. The SCENAREO can provide the desired performance and resource values for each science-return requirement in order to help the conflict resolution process.

Comprehensive preliminary analysis requires a mapping between the observation condition and the subsystem properties and inter-dependencies of the subsystem properties. In the surface mapping case, the surface condition is used to obtain the valid time range for the observation. The valid time range (combined with the instrument resolution) provides total data volume obtained during the surface mapping. The data volume and the orbit allocation for downlink determines required downlink rate, which in turn requires antenna properties. In the target observation case, a long exposure duration required due to either low instrument sensitivity or low target albedo may drive the ACS design if the instrument is body fixed in order to minimize the motion blurring. The target size, the search area, and the field-of-view of the instrument all affect the gimbal design in order to capture the target in the instrument's frame as the spacecraft orbits.

Activity Composition [X]

Science Objective |

ScienceFile: D:\mars\WM-Orbit\apps\FVM\FVMUI_Orbit\acttest.science [Browse] [Save]

Activity Name: EuropaTest

Mapping Orbits

All

any N orbit []

Selected Orbit []

Conditional (1, 2, ..., n)

Target Observation Orbits

All

any N orbit []

Selected Orbit []

Conditional (1, 2, ..., n)

Downlink Orbits

All

any N orbits []

Selected Orbit []

Conditional (1, 2, ..., n)

Surface Mapping

G*	Instrument Name*	Active (T/F)	Incidence Angle (deg, deg)	Observation Duration (%)	Pointing Angle (degree)
Y	NAC	F	0.000000, 90.000000	50.000000	0.000000
N	RADAR	T	90.000000, 180.000000	40.000000	0.0

Target List File Name: Q:\Missions\NSI\scienceTargets.doc [Browse]

Target Observation

G*	Instrument Name*	Active (T/F)	Search Range (lon, lan)(deg)	Incidence Ang. (deg, deg)	(Max Targ., Fr./Targ.)	Pt. Cond.	Exp. Dur.(sec)
Y	NAC	T	5.000000, 1.000000	0.000000, 90.000000	5, 1	10.000...	0.050000
N	RADAR	F	5.000000, 15.000000	0.000000, 90.000000	1, 1	0.000000	0.000000

Downlink Operation

G	Antenna Name	Active (T/F)	Pointing Time Ahead (sec)	Data Condition (% of memory used)	Ground Station (station 1, station 2, ...)
Y	HGA	T	10.000000	100.000000	DSN
Y	MGA	F	10.000000	100.000000	

Available Ground Stations

- DSN
- Madrid
- Goldstone
- Canberra
- NOSSORS

*: unchangeable

Figure 4 Science Objective UI

The observation activity is composed as a set of subsystem events. Each subsystem event is described with an operation statement and a condition statement. The event operation is either a subsystem command or a macro command that can be decomposed as a set of subsystem commands. The list of supported subsystem commands is made available via a command dictionary. The event condition is defined as a logical combination of three types of conditions—target condition, time condition, and command condition. The target condition is used to express the necessary target state during the event operation. The supported target states include distance, apparent size, phase angle, and so forth. The time condition is used to express the required time between events within a subsystem. The command condition is used to express interdependency and concurrency of the events among the multiple subsystems. In order to resolve the event conditions of each activity (target condition, time condition, and command condition) a set of condition analysis software modules is employed. By interfacing with the S-subsystem and M-subsystem objects, the condition analysis software modules estimate the range of time when the event conditions can be satisfied

The SF provides a generic command dictionary that defines a set of commands per subsystem representing typical operation control properties of the subsystem. Each command is composed of command name and a list of arguments. Each argument is composed of argument name, value type, value range, and a default value. Currently, a limited set of commands that are essential for science observation planning have been defined. Table I illustrates a few subsystem command examples.

Subsystem	Command	Arg #1	Arg #2
AACS	Point	Device	Target
Payload	Exposure	Device	Duration
Power	On	Device	
Telecom	Downlink	Device	File name

Table I Subsystem Command Examples

The SCENAREO utilizes the M-subsystem library for translating the operation objectives and the operation conditions into subsystem-level command sequences. The translation process involves operation opportunity analysis, resource usage prediction and availability analysis, and command sequence generation. The command sequence specifies the timeline of the subsystem commands, which is determined to satisfy the operation conditions and minimize the resource usage. During the mission operation phase, the SCENAREO can be used as

an activity-planning tool by replacing the command-level models with a mission-specific activity dictionary and command dictionary.

TSOS

The on-board processing of a spacecraft can be divided into system functions and subsystem functions. The system functions include command sequence management, fault recovery, and telemetry generation. The subsystem functions include command and data handling of individual subsystem. The TSOS simulates the on-board processing employing three system-level modules, a sequence manager, CDH, and a virtual telemetry and five V-subsystems. Each V-subsystem is linked with a corresponding M-subsystem, and it simulates the operation behavior of the subsystem including command execution, state propagation, and resource management. The technical details of the TSOS are presented in the paper “Time-based Spacecraft Operation Simulator”.

The five virtual subsystems, V-AACS, V-Payload, V-Telecom, V-Power, and V-Process, simulate command and data handling of the corresponding subsystems. The V-AACS is responsible for articulation of the gimballed devices including antennas, solar panels, and scan platforms. The V-payload is responsible for all instruments, both optical and non-optical. The optical instruments include camera systems and spectrometer systems while the non-optical instruments include radar and laser systems. The V-Telecom is responsible for downlink operation of the acquired data by the V-Payload. The V-Power is responsible for tracking the power load of the devices that are turned on. The V-Process is responsible for on-board data processing and file system operations.

Each V-subsystem is implemented as a virtual device with three basic modes, READY, BUSY, and COMMAND, indicating that the V-subsystem is ready to receive a command, in the middle of executing the previously received command, and just received a new command, respectively. If a new command is received while a V-subsystem is busy, the V-subsystem rejects the command, and the error is logged to indicate that the command timing error has been detected.

The TSOS simulates the operation state of a spacecraft by propagating each subsystem state as the spacecraft clock progresses the time in a regular interval. The subsystem state propagation is performed based on the operation rate and operation duration. The operation rate may be speed of a spacecraft, turn rate of a gimbal, IO rate of a buffer, and so on. Various system noises can be also simulated by specifying error range and distribution. At each sample interval, the TSOS composes a spacecraft state record by integrating the simulated subsystem states and writes out to a file. The spacecraft state record is referred to as a



Figure 5 A Snapshot of the Vis-AACS Execution

telemetry record, and the file is referred to as a telemetry data file.

The telemetry record is composed of five subsystem state fields, Navigation, AACS, Payload, Telecom, and Power. Each subsystem state, T-subsystem, records information necessary to monitor the relationship of the subsystem and its environment (i.e., Sun, Earth, and target). The T-Navigation records time, SC position, and SC velocity relative to the Sun. the T-AACS records attitude of the main body and a list of articulated devices For each device, device name, target name, intersect location and gimbal information are recorded. The gimbal information includes the number of gimbal axes, gimbal angles, and pointing direction. The T-Payload records a list of instruments. For each instrument in operation, the T-Payload records instrument name, target name, intersect location, data file name, and the gimbal information. The T-Telecom records the antenna name, the file name that is being downlinked, and the downlink rate. The T-Power records a list of activated devices with their power usage status.

4. VISUALIZATION FRAMEWORK

The Visualization (Vis) Framework is referred to as “MicroHelm”. The “Micro” indicates the PC-cluster architecture and the “Helm” indicates the comprehensive monitoring capability of the spacecraft operation. The Micro-Helm hosts a telemetry data server and a set of visualization clients that are organized along the subsystem architecture in areas of Navigation, AACS, Payload, Telecom, and Power. Each subsystem

visualization client, Vis-subsystem, employs one or more graphical windows to project the subsystem state information described above. The graphical windows are managed by two servers, Overlay server and Animation server. The Overlay server manipulates 2D images and plots while the Animation server provides 3D solid model rendering.

Each Vis-subsystem implements three functions, initialize, visualize, and exit. During the initialization, it establishes the socket connection with the telemetry data server and receives the subsystem design information as described in the S-subsystem for the static properties of the spacecraft system. During the visualization, it receives the subsystem states as described in the T-subsystem for the dynamic properties of the spacecraft system. Each Vis-subsystem extracts relevant information by calling the member functions of the S-subsystem and T-subsystem objects and displays the extracted information by integrating it with the appropriate environment models.

The Vis-Navigation provides three types of trajectory displays for projecting the spacecraft position during three types of mission phases: cruise, orbit insertion, and in-orbit. The cruise phase trajectory displays the geometric relationship of the spacecraft with respect to the Earth and the target. The orbit insertion phase trajectory displays the geometric relationship between the spacecraft and the target. The in-orbit phase trajectory displays the orbit track on the target surface.

The Vis-AACS provides the spacecraft orientation and the pointing direction of the articulated devices. The X,Y,Z

coordinate axes, the Sun direction, the Earth direction, and the SC nadir direction are indicated to help validating the orientation of the spacecraft and the articulated devices. Vis-AACS also renders the target view and overlays field-of-view of the instruments so that the target pointing can be validated. Figure 5 shows a snapshot of Vis-AACS execution.

The Vis-Payload overlays the footprints of the instruments on the target surface that is shaded for the Sun visibility. The footprint overlay includes the swath coverage of the push-broom operation during surface mapping and the projected field-of-view during the target observation. In addition to the footprint overlay, a surface map is dynamically composed based on the intersect location information to help monitoring the surface coverage status.

The Vis-Telecom provides two kinds of maps: one for monitoring the activated DSN station during the Downlink operation, and the other for monitoring the received data content from the surface mapping operation. When the downlink rate is not sufficient, the surface map doesn't match with the surface map created by the Vis-Payload.

The Vis-Power provides the available power, the power consumption profile of each device, and the state of the device. If the available power is lower than the required power usage, the devices cannot be turned on.

The Vis-subsystems in the Micro-Helm can be utilized during the mission operation phase, enabling visual validation of the test runs prior to the Uplink process as well as the real mission telemetry data. Side-by-side comparison between the simulated telemetry data and the real telemetry data reveals discrepancies between the models and the actual system.

5. FUTURE DIRECTION

The VMOF team plans to extend the architecture to become a multi-discipline collaborative engineering framework in support of the Model-Based Engineering Design (MBED) initiative at JPL. The initiative is to enable advanced systems engineering practice through a series of integrated, increasingly detailed models that provide continuity from architectural concept through detailed design. It will extend current capabilities for rapid conceptual design, allowing thorough exploration of design trade-spaces and selection of an optimal design point with associated cost and rationale; and it provides seamless connection to subsystem models and detailed design tool suites [3].

The VMOF will integrate three innovative technologies: an object-oriented modeling language, a distributed-object paradigm, and multi-media streaming. The object-oriented structure modeling language will be applied for generative

domain knowledge modeling, environment effects analysis tool integration, and project lifecycle process tracking. The distributed-object paradigm will be applied for multidiscipline model/tool coordination to resolve interdependencies among the discipline processes and provide data flow control among the tools on distributed heterogeneous platforms. The multi-media streaming will be applied in synchronized viewing of 4D visualizations for multiple-viewpoint rendering.

Multi-discipline Structure Modeling

Object-oriented structure models enable the configuration and layout of the frame structural members and definitions of their dimensions. The detailed properties of the structural members can be automatically derived as the design process evolves for sizing and shape refinement, thus providing a hierarchical model architecture. The level of fidelity required at each design stage can be captured and represented in the model objects to support collaborative design and multidisciplinary optimization. The model objects can support parametric, rule-based, manifold, and non-manifold mesh generation. AML's attribute tagging process traces the enhancement of model topology during the geometry-creating process; and 2D, 3D, structured, and unstructured meshes can be automatically generated. Attributes for load, boundary, and initial conditions can be directly set on the part geometry and automatically associated with the generated mesh groups. Nodal coordinates, connectivity, elements and material properties, loading conditions, bounding constraints, and solution control parameters will be specified through the analysis model objects.

At JPL, the spacecraft structure designers have chosen Unigraphics as a standard mechanical engineering design tool. Unigraphics utilizes Parasolid's geometric model engine, which is an exact boundary-representation geometric modeler supporting solid modeling, generalized cellular modeling, and integrated freeform surface/sheet modeling. Based on the geometric model engine, AML provides a modeling language that can link the geometric models with external analysis packages. Employing Unigraphics and AML, we will create multidisciplinary structure objects that integrate environment effect analysis tools (e.g., the radiation shielding transport codes, Novice and MCNP) and trajectory analysis tools (STK, SPICE). For visualization of spacecraft interactions with the environment phenomena, we will develop multi-fidelity structure models and develop an export mechanism to OpenGL compatible data formats.

4D Broadcasting System

To fuse, formulate, and deliver visual information to the remotely located design teams in a synchronous manner is a very challenging task. There are many data encoding formats for digital broadcasting of audio and video. The

most popular standards are the Moving Pictures Experts Group (MPEG) standards MPEG-2 (video), MPEG-3 (audio), and most recently MPEG-4 (combined audio, video, and graphics). Actual video imagery is confined to the event at hand. The graphics presentation of state is not limited so. It can be edited, refined, stored, and composed in a time-compressed form to present the most relevant information at the rate it can be assimilated. The viewpoint can be changed. The lighting can be enhanced. The shadows can be removed. Multiple cameras can be introduced where a single actual camera may have missed the event of interest.

In order to share complicated interactions between a spacecraft and the environment, three types of visualization products will be generated and shared: environment phenomena, spacecraft operation, and environment interactions. The visualization products involve 2D plots, 3D solid body rendering, and 3D volumetric rendering. The rendering will be performed employing commercial off-the-shelf tools available on PC platforms (e.g., Nvidia, MentalRay). Sharing of the visualization products will be performed via a 4D broadcasting system that is composed of a video and audio stream interface, a broadcast station, and a receiver. The stream interface will utilize the MPEG-4 encoding standard and extend it to include the kinematics state. The resulting new format will allow for low bandwidth broadcasts capable of supporting an order of magnitude more clients per broadcast machine than those supported by existing commercial digital broadcast systems. Each visualization product will be broadcasted in multiple channels where each channel provides a specific viewpoint (distance and orientation). The receiver will be able to handle up to ten viewpoints (i.e., 10 channels) for comprehensive viewing of the complex phenomena. For synchronized viewing among the multiple users, the broadcast station will dynamically create additional servers based on the transmitted information quantity and update-frequency.

The broadcast system is scalable in multiple levels: per visualization application, a broadcast station can be added to the system; per viewpoint, a channel can be added to the receiver; per user, a receiver client can be added. The common models that are shared between the broadcast station and the receiver will be provided via a separate model service system implemented either as a distributed model object or a web service depending on the model properties. When a receiver is turned on, it will receive the model updates automatically via the model service system. The models can be dynamically updated during the broadcast, if necessary, by being prompted by the broadcast station. This research will utilize the Micro-Helm infrastructure described in Section 4.

ACKNOWLEDGEMENTS

The work described in this paper was performed at Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

REFERENCES

- [1] M. Lee and R. J. Weidner, "Science Observation and Mission Life cycle," presented at SpaceOps-2002, Houston, Texas, 2002.
- [2] R. J. Weidner, "A Component-based Implementation of Agents and Brokers for Design Coordination," presented at 2001 IEEE Aerospace Conference, Big Sky, MN, 2001.
- [3] S. Wall, "The Mission System Design Center: A Pilot of Formulation-Phase Concurrent Engineering in Aerospace Design", *Tenth Annual International Symposium of the International Council on Systems Engineering*, 2000.
- [4] M. Kordon and E. Wood, "Multi-Mission Space Vehicle Subsystem Analysis Tools", *Proceedings of the IEEE Aerospace Conference*, 2003.
- [5] S. Wang, "SCENAREO- Science and Engineering Activity Reasoning and Optimization", IEEE Aerospace Conference, Big Sky, Montana, March 2004.
- [6] W. Lu, "TSOS – Time-based Spacecraft Operation Simulator", IEEE Aerospace Conference, Big Sky, Montana, March 2004.
- [7] S. Y. Wang, M. Lee, K. Peter, C. W. Lau, "Integration of Virtual Mission (VM) and Telecom Forecaster Predictor(TFP)," Presentation at JPL 2002 IT Symposium, JPL, Pasadena, California, November 2002.

Meemong Lee is a principal technologist at the Jet Propulsion Laboratory in the areas of modeling and simulation. She leads mission system modeling and simulation research activities in support of the Project Design Center and advanced engineering environment.



Her current research and development activities include Engineering Framework for Model-based Engineering Design initiative, Virtual Mission Operation Framework for design validation, and Virtual In-situ Site for autonomous In-situ science observation .She also manages the project engineering technology program at JPL. She has a bachelor's degree in Electronics Engineering from Sogang University in South Korea, a master's degree in Computer Science, and a doctoral degree in Electrical Engineering from Oklahoma State University.

Richard J. Weidner is a principal technologist and technical Group Supervisor of the Mission Simulation and Instrument Modeling Group at JPL. He supported automated mosaic sequence generation and real-time panorama camera image interpretation during the Mars



Pathfinder mission. He invented Micro-Helm for real-time telemetry visualization in support of Mars Odyssey. He developed automated parser generator (Sax/Luthor), Overlay server, Mosaic server, Trajectory server, and Spacecraft structure model for the Virtual Mission Operation Framework. His current research and development activities include distributed mission model objects, model exchange network, and four dimensional model broadcasting system. He has a bachelor's degree, a master's degree, and doctoral degree in Electrical Engineering from Oklahoma State University.