

Multi-Purpose Spacecraft Simulator for LADEE

Nathaniel Benz
MEI
Moffett Field, CA 94035
nbenz@meicompany.com

Danilo Viazzo
Cummings Aerospace
Dublin, CA 94568
danilo.viazzo@cummingsaerospace.com

Karen Gundy-Burlet
NASA-Ames Research Center
Moffett Field, CA 94035
karen.gundy-burlet@nasa.gov

Abstract—The Lunar Atmosphere Dust Environment Explorer (LADEE) was a small explorer class spacecraft that was launched on Sept 7, 2013. After completing all of the mission objectives, LADEE was de-orbited and successfully impacted the moon’s surface on April 17, 2014. In order to reduce costs and leverage previous research and development, LADEE utilized the “common modular bus” design. The physical construction utilized a low-cost, rapidly prototyped design and the same philosophy drove the development of the software base. To achieve this goal, a Model Based Design approach was utilized to develop the On-board Flight Software (OFSW), and coincident with this, a model-based multipurpose simulator was created of the LADEE spacecraft and its mission environment. The spacecraft simulator was designed to have sufficient fidelity that it could be used to test the required modes and responses of the OFSW. A faster-than-real-time Workstation Simulator (WSIM) version of the LADEE simulator was used to develop and test the software control algorithms in the Simulink environment. The automatic code generation feature in Simulink was used to port the simulation to several real-time environments to support Processor-in-the-Loop (PIL) and Hardware-in-the-Loop (HIL) testing, verification and validation. Since the simulation interface was designed to be compatible with the command interface employed by the LADEE mission operation team, the WSIM, PIL and HIL simulators were used for both personnel training (nominal and off-nominal operations) prior to the mission and to perform command sequence verification during the mission. This approach resulted in time and budget savings, and allowed changes to the model to be quickly propagated through all the simulation target environments. The mode- based design approach also allows the simulation framework to be generalized and reused to model future small-satellite missions.

TABLE OF CONTENTS

1	INTRODUCTION	1
2	SIMULATION ARCHITECTURE.....	2
3	ENVIRONMENT MODELS	2
4	VEHICLE MODEL	6
5	ELECTRICAL SYSTEM MODEL	7
6	THERMAL MODEL.....	7
7	COMMUNICATION MODEL	8
8	PROCESSOR AND HARDWARE IN THE LOOP	9
9	SOFTWARE DEVELOPMENT AND TESTING	10
10	MISSION OPERATIONS	11
11	RESULTS AND CONCLUSIONS	12
	ACKNOWLEDGMENTS	14
	REFERENCES	14
	BIOGRAPHY	14

1. INTRODUCTION

The Lunar Atmosphere Dust Environment Explorer (LADEE) was a lunar orbiter developed by NASA to study the lunar atmosphere and dust environment prior to pending human activities that would perturb the pristine state. The mission goal was to determine the composition of the lunar atmosphere and investigate the processes that control its distribution and variability, including sources, sinks and surface interactions. LADEE also determined that dust is present in the lunar exosphere, and took measurements to determine the processes that contribute to the source and variability of the lunar dust in the exosphere. The results of the mission will help in the understanding of surface boundary exospheres and dust processes of similar bodies, address questions regarding the origin and evolution of lunar volatiles, and have potential implications for future exploration activities [1].

The top-level programmatic and science requirements for the LADEE project were designed to accomplish the following mission objectives [1]:

- 1) Determine the composition of the lunar atmosphere and investigate the processes that control its distribution and variability, including sources, sinks and surface interactions.
- 2) Characterize the lunar exospheric dust environment and measure any spatial and temporal variability and impacts on the lunar atmosphere.
- 3) Demonstrate that Lunar Laser Com Demonstration (LLCD) can operate at high data rates from lunar distances.
- 4) Create a low-cost reusable spacecraft architecture that can meet the needs of certain planetary science missions.
- 5) Demonstrate the capability of the Minotaur V as a launch vehicle for planetary missions.

To accomplish the fourth goal above, the LADEE spacecraft bus was the first use of the Modular Common Spacecraft Bus (MCSB) architecture designed and developed at NASA Ames. The MCSB is a small, low-cost spacecraft designed to deliver scientifically and technically useful payloads to a variety of locations, including Low Earth Orbit (LEO), lunar orbit and lunar surface, Earth-Moon Lagrange points, and Near Earth Objects (NEOs) [1].

The spacecraft bus is a lightweight carbon-composite structure designed to accommodate launch loads and provide attenuation of impact loads. It is also designed for ease of manufacturing and assembly. The modularity of the design is intended not only for multiple mission configurations but also parallelism in development and assembly. The system-level components were drawn from low-cost, flight-proven product lines [1].

With the LADEE physical construction proceeding down

a low-cost, rapidly prototyped product-line type of effort, a similar philosophy drove the development of the software base. Successful examples of rapidly deployed small-spacecraft missions were surveyed. Of these, the AFRL XSS-10 and XSS-11 spacecraft utilized model-based development methodologies and tools. Additionally, a trade study on the methodologies and tools was performed using a hover test vehicle. After the success of the hover test vehicle, a model-based development approach was selected for the flight software development for LADEE [2].

A spacecraft simulator was developed that also uses model-based design to test and validate the generated flight software. Sections 3 through 7 of this paper will discuss in detail the components of this spacecraft simulator. Section 8 will show how the simulator was auto-coded to create a processor and hardware in the loop simulators. Section 9 discusses how the simulator was used for flight software development and testing. Section 10 will review how the simulator was used beyond the flight software development process during mission operations. Section 11 will provide a specific mission event where the simulator was used which resulted in time and cost savings. Section 11 will also review the lessons learned and future directions of the LADEE spacecraft simulator.

2. SIMULATION ARCHITECTURE

The LADEE flight software utilizes a model-based development approach layered on minimal hand-code, Government-off-the-shelf (GOTS) and Commercial-off-the-shelf (COTS) software packages. [2]. The core of the on-board flight software as well the simulation software is modeled using Mathwork's Simulink.

One advantage with model-based development was that control system engineers were able to rapidly prototype algorithms in their design environment and software engineers could directly auto-code the models. This minimizes the likelihood for communication errors between algorithm designers and software developers. The model based-methodology also enhances early prototyping of requirements, enables validation and verification during early stages of development and provides a common platform for communication between subsystems, software engineers and stakeholders [2].

A Simulink workstation simulator (WSIM) was created as an integrated system model that references "model libraries". A model library is a Simulink feature to create reusable components. As shown in Figure 1 the top level of the Simulink model is divided into two libraries: On-board Flight Software (OFSW) and the Ground Side Simulation Equipment (GSSE). The GSSE model is broken up into component models of the Environment, Communication, Electrical, Thermal and Vehicle using Simulink library blocks as shown in Figure 2. The model is broken up into library blocks for the following reasons:

- 1) Avoid a large, monolithic system-model file.
- 2) Allow easier version control of subsystem files.
- 3) Allow collaborators to work on separate subsystems without a difficult merge process.
- 4) Break up and assign requirements to specific subsystems and test independently.

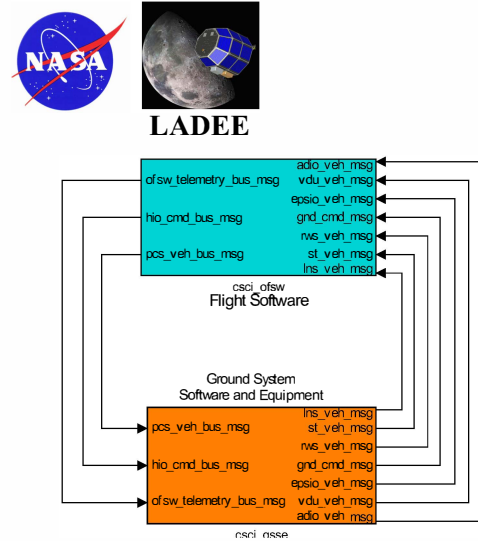


Figure 1. Top Level of WSIM model

- 5) Create a modular model catalog that allows for easier re-use.

For organization and readability, model libraries are further classified as either Utilities (Util), Computer Software Component (CSC) or Computer Software Unit (CSU).

Utilities—A utility library is an implementation of a simple computational unit. For example, unit conversions, quaternion math function, or look-up tables are all members of a utility library. Utility libraries usually are referenced in multiple different models including the onboard flight software algorithms and the simulator models.

Software Unit—A CSU is a model that performs a single task. A software unit can only be composed of basic Simulink blocks or utility blocks, and cannot contain any other CSU or CSC. Each CSU has an associated unit test that is tied to the requirement(s) for that specific unit. For readability, software units are identified with a light blue background.

Software Component—A CSC integrates one or more CSUs or other CSCs. Each CSC is associated with one or more software requirements on system model integration. For readability, software components are colored yellow

3. ENVIRONMENT MODELS

The Environment Subsystem provides models that represent the physical world and its effects on the state of the spacecraft. The model is partitioned into five components: Time, Celestial Bodies, Gravity, External Disturbances and Rigid-Body Dynamics.

Time

The Time model computes the environmental reference time in the form of mission elapsed time (MET), Coordinated Universal Time (UTC) time, and Julian Date. A 'Tic' block

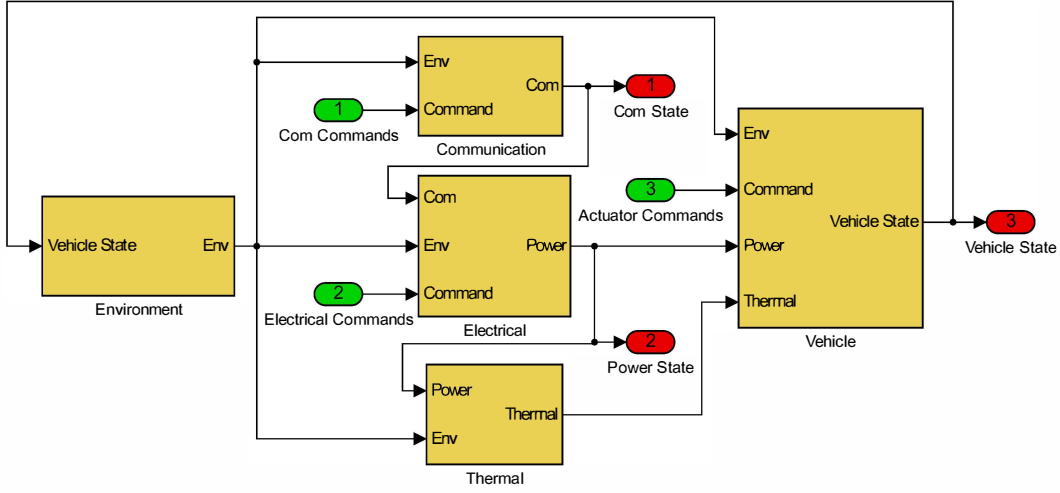


Figure 2. Simulation Architecture

provides the heartbeat counter of the simulation clock, which for the LADEE simulator is set to update at 50 Hz. That count is then turned into a time using the 'Tic to Sec' conversion block to transform the count into seconds since the start of the simulation. This signal is then fed to the 'Mission Elapsed Time' block to provide seconds since a user-specified mission time and can be reset to a specified value through a reset control signal. The mission elapsed time feeds both the 'MET to Time' block to compute the MET in days, hours, minutes and seconds and to the 'UTC Time' block to compute the UTC year, day number, hour, minute and second. The output of the UTC Time block is fed into the 'Julian Date' conversion block to compute the Julian date that corresponds to the input UTC time. The different time signals are collected into a Simulink bus signal and used by the Celestial Bodies and Rigid-Body Dynamics subsystem models.

Celestial Bodies

The Celestial Bodies model is a component of the Environment model and consists of six subsystems: Sun Position, Moon Position, Moon Libration, Greenwich Hour Angles, Occlusions, and Lunar Orbit Parameters.

Sun Position—The Sun Position model computes the position of the Sun in the ECI J2000 frame and the velocity of the Earth relative to the Solar System Barycenter (SSB) expressed in the J2000 frame. The Sun position and Earth velocity are calculated using the Jet Propulsion Laboratory (JPL) Development Ephemeris (DE) data set models [3]. The JPL planetary ephemerides are saved as Chebyshev polynomials fit to the Cartesian positions and velocities of the Sun, Moon and Earth. The model uses the Julian date from the Time model to extract the index and fraction of the Chebyshev polynomials stored in a lookup table. In order to express the Sun position relative to the ECI frame, the following equation is applied [4]:

$$\vec{r}_{\odot ECI} = \vec{r}_{\odot SSB} - \vec{r}_{EMB_{SSB}} + \frac{1}{1 + \mu} \vec{r}_{\text{Moon}} \quad (1)$$

where $\vec{r}_{\odot ECI}$ is the Sun position in the ECI frame, $\vec{r}_{\odot SSB}$ is the Sun position in the SSB frame (calculated directly

from the JPL Ephemeris data), $\vec{r}_{EMB_{SSB}}$ is the Earth-Moon Barycenter in the SSB frame (calculated directly from the JPL Ephemeris data), μ is the ratio of masses for the Earth and Moon and \vec{r}_{Moon} is the Moon position in the ECI frame.

The following equation is used to compute the velocity of Earth relative to the SSB frame and expressed in the ECI frame:

$$\vec{v}_{\oplus ECI} = \vec{v}_{EMB_{SSB}} - \frac{1}{1 + \mu} \vec{v}_{\text{Moon}} \quad (2)$$

where $v_{\oplus ECI}$ is the Earth velocity relative to the ECI frame (since the SSB frame and the ECI share the same J2000 orientation), $v_{EMB_{SSB}}$ is the velocity of the Earth-Moon Barycenter relative to the SSB frame and \vec{v}_{Moon} is the velocity of the Moon relative to the ECI frame.

Moon Position—The Moon Position model computes the velocity and position of the Moon in the ECI J2000 frame. The Moon position and velocity are calculated directly from the Chebyshev polynomials look-up table using the JPL Development Ephemeris (DE) data set models [3].

Moon Libration—This model computes the Moon libration angles and associated transformation matrix from the J2000 to Moon Principal Axis (MPA) frames. The Moon libration angles are calculated directly from the Chebyshev polynomial lookup table using the JPL Development Ephemeris (DE) data set models [3]. The resulting angles are used to compute the transformation matrix from the J2000 orientation to the MPA orientation using the 3-1-3 rotation sequence [6] as defined below:

$$\mathbf{M}_{J2000}^{MPA} = \mathbf{R}_z(\psi) \mathbf{R}_x(\theta) \mathbf{R}_z(\phi) \quad (3)$$

where ϕ , θ , and ψ are the Moon Libration angles.

Greenwich Hour Angle Model—The Greenwich Hour Angle model computes an approximation of the rotation of the Earth-Centered Earth-Fixed (ECEF) frame relative to the ECI J2000 frame about the z-axis. It also computes the

resulting transformation Direction Cosine Matrix (DCM) to transform a vector from ECI J2000 to the ECEF frame. This model assumes that the z-axis of the ECEF frame and the z-axis of the ECI J2000 frame are coincident and the relative rotation between these two frames is limited to be about the z-axis. This rotation is consistent with the right-handed direction about the z-axis and thus follows the Greenwich Mean Sidereal Time (GMST) polarity which is defined to be positive in the counter-clockwise direction[7].

Occlusion Model—The occlusion model determines the following four elements: whether the Sun is in sight of the LADEE spacecraft (not occluded by the Earth or the Moon), the unit vector of the Sun position in the body frame, the angles between normal vectors of the vehicle's external surfaces with the Sun vector and the angles between normal vectors of the vehicle's external surfaces with the Moon vector. For LADEE, the vehicle's external surfaces include each of the solar panels and an approximate surface area for the top and the bottom of the spacecraft. The surface angles relative to the Sun and Moon vectors are determined from the current position and orientation of the vehicle and the position of the Sun and the Moon. The occlusion and surface angles information is required for the external forces model (solar pressure), the power model (solar array power generation), the coarse sun sensor model and the thermal model.

Lunar Orbit Parameters Model—The Lunar Orbit Parameters model uses vehicle position and velocity in ECI J2000 and Moon position and velocity in ECI J2000 along with the J2000 to Moon Principle Axis (MPA) transformation to compute MPA and Moon-centered orbital parameters. The vehicle position and velocity relative to the Moon in the ECI J2000 frame is computed by subtracting the Moon position vector from the vehicle position vector and by subtracting the Moon velocity vector from the vehicle velocity vector, respectively.

Gravity Model

The Gravity model computes the gravitational force applied to the LADEE spacecraft in the ECI J2000 frame given the position of the spacecraft, Earth, Moon and Sun. Gravitational contributions from any other celestial body are considered to be negligible.

Sun Point-Mass Gravity Model—Due to LADEE's distance from the Sun, the Sun is treated as a point mass when determining gravitational force [4].

Earth/Moon Spherical Harmonics Gravity Model—A spherical harmonics gravitation model is used for both the Earth and the Moon gravitational forces. It is assumed that the spherical harmonics model order and degree are the same. The order and degree are tunable values that can be set during the model initialization step, depending on the level fidelity desired by the user. The maximum fidelity for real-time simulation of the LADEE model in lunar orbit was 8x8 for the Earth and 50x50 for the Moon. The moon is modeled at a higher order since its effects are obviously more dominant with the spacecraft in Moon orbit. The model is initialized with data loaded and processed from the EGM96 database for the Earth and LP150Q database for the Moon. The normalized spherical harmonics coefficients are loaded and converted back to the unnormalized coefficient form. This is accomplished by solving for the unnormalized coefficients

[7].

$$\begin{bmatrix} C_{nm} \\ S_{nm} \end{bmatrix} = \sqrt{\frac{(n-m)!}{(n+m)!}} (2 - \delta_{0m})(2n+1) \begin{bmatrix} \bar{C}_{nm} \\ \bar{S}_{nm} \end{bmatrix} \quad (4)$$

where n and m represent the degree and order respectively and δ_{0m} represents the Kronecker delta function defined as

$$\delta_{ij} = \begin{cases} 0, & \text{if } i \neq j \\ 1, & \text{if } i = j \end{cases} \quad (5)$$

A precomputed initialization script is used to extend the numerical precision and range and to minimize the calculation involved in determining ratios of factorials. The model uses these unnormalized coefficients to compute the acceleration experienced by the vehicle due to the specific celestial body.

The spherical harmonics fields are fixed to the reference body, therefore the vehicle position used to compute the resulting gravitational force needs to be expressed relative to the reference celestial body fixed frame and expressed in the reference celestial body fixed frame. The resulting gravitational acceleration is also expressed in the reference celestial body fixed frame, thus it needs to be transformed back in the J2000 ECI frame. The change in origin between these frames does not need to be accounted for since this is an acceleration vector and not a position vector.

The reference frame in which the accelerations are being used to propagate the vehicle state vector is the J2000 ECI frame. Since the J2000 ECI frame is not truly inertial (non-accelerating) relative to the Sun and Moon, additional acceleration terms need to be included.[7]. An augmented form of the gravitational acceleration equation which includes both the Sun and the Moon acceleration can be expressed as follows:

$$\ddot{\mathbf{r}}_{\oplus sat} = -\frac{\mu_{\oplus} \vec{\mathbf{r}}_{\oplus sat}}{r_{\oplus sat}^3} + \frac{\mu_{\odot} \vec{\mathbf{r}}_{sat \odot}}{r_{sat \odot}^3} - \frac{\mu_{\odot} \vec{\mathbf{r}}_{\oplus \odot}}{r_{\oplus \odot}^3} + \dots \\ \frac{\mu_{\text{J}} \vec{\mathbf{r}}_{sat \text{J}}}{r_{sat \text{J}}^3} - \frac{\mu_{\text{J}} \vec{\mathbf{r}}_{\oplus \text{J}}}{r_{\oplus \text{J}}^3} \quad (6)$$

where μ represents the standard gravitational parameters for each celestial body.

External Disturbances Model

The External Disturbances model includes the environmental forces and torques acting on the LADEE spacecraft computed in the body frame. The forces and torques modeled are earth gravity gradient, moon gravity gradient, and solar radiation. User-defined disturbance force and torque values can also be added as a tunable value. Additionally, each individual disturbance force and torque can be turned 'on' or 'off' from the initialization script.

Gravity Gradient Torque Model—The disturbance torque due to gravity gradient for both the Earth and the Moon is computed as follows [9]:

$$\vec{\mathbf{T}}_{gg} = \frac{3\mu}{r_{sat}^3} [\vec{\mathbf{r}}_{sat} \times \mathbf{I}_{sat} \vec{\mathbf{r}}_{sat}] \quad (7)$$

where $\vec{\mathbf{T}}_{gg}$ is the gravity gradient torque acting on the spacecraft, $\vec{\mathbf{r}}_{sat}$ is the geocentric position vector of the origin of

the body reference frame, \mathbf{I}_{sat} is the moment of inertia tensor, and $\mu \equiv GM$ is the gravitational constant of the reference body.

Solar Pressure Model—The force and torque due to solar pressure is defined as follows:

$$\vec{\mathbf{f}}_{i,\odot} = -\frac{F_e}{c} A_i \times \dots \left[(1 - c_s) \hat{\mathbf{s}}_i + 2 \left(c_s C_{\theta_i} + \frac{1}{3} c_d \right) \hat{\mathbf{n}}_i \right] C_{\theta_i} \quad (8)$$

$$C_{\theta_i} = (\hat{\mathbf{s}}_i \cdot \hat{\mathbf{n}}_i) \quad (9)$$

$$\vec{\mathbf{f}}_{\odot} = \sum_{i=1}^n \vec{\mathbf{f}}_{i,\odot} \quad (10)$$

$$\vec{\mathbf{t}}_{\odot} = \sum_{i=1}^n \vec{\mathbf{r}}_{i,c/m,c/p} \times \vec{\mathbf{f}}_{i,\odot} \quad (11)$$

where A_i is the area of each surface, $\hat{\mathbf{s}}_i$ is the unit vector from each spacecraft surface to the sun, $\hat{\mathbf{n}}_i$ is the unit outward normal of each spacecraft surface, c_s is the coefficient of specular reflection, c_d is the coefficient of diffuse reflection, $\vec{\mathbf{f}}_{\odot}$ is the total solar radiation force on each surface element, $\vec{\mathbf{t}}_{\odot}$ is the torque due to solar pressure and $\vec{\mathbf{r}}_{i,c/m,c/p}$ is the vector from the spacecraft center of mass to the center of pressure of the i th spacecraft surface.

Rigid-Body Dynamics

Translational Kinematics Model—The Translational Kinematics model computes the linear position and velocity of the LADEE vehicle in the ECI J2000 frame. A configurable subsystem is used for the model, allowing the user to specify whether interpolation or state propagation is used to calculate the vehicle's position and velocity. The trajectory interpolation option uses a predetermined table of time and position with cubic spline interpolation while the state propagation option uses the accelerations experienced by the vehicle and integrates them using Adams-Bashforth integration. Simulation results demonstrated the Adams-Bashforth integration updating at 50 Hz able to run at least 30 times faster than real-time with WSIM and at real-time on the PIL and HIL without any degradation in performance.

The interpolation option uses time (in the units of Julian day UTC) to select the spline polynomial coefficients for the position curve fit corresponding to the time before the specified time. The fraction of the time interval between consecutive control points and the specified time is used to compute the cubic polynomial powers which are then multiplied by the spline coefficients to calculate the splined curve value at the specified time. To compute the velocity at the time, the cubic polynomial is differentiated to obtain the [3 2 1] multipliers for the cubic, square, and linear terms of the polynomial while using the same spline coefficients.

The propagation algorithm uses resettable two-step Adams-Bashforth integration, as follows [11]:

$$\begin{bmatrix} \vec{p} \\ \vec{v} \end{bmatrix}_{i+1} = \begin{bmatrix} \vec{p} \\ \vec{v} \end{bmatrix}_i + \frac{T}{2} \left(3 \begin{bmatrix} \vec{p} \\ \vec{v} \end{bmatrix}_i - \begin{bmatrix} \vec{p} \\ \vec{v} \end{bmatrix}_{i-1} \right) \quad (12)$$

where \vec{p} is the position vector \vec{v} is the velocity vector and T is the time step.

Translational Dynamics Model—The Translational Dynamics model sums the external (disturbance) forces and the actuation forces in the vehicle Body frame. It then transforms them in the ECI J2000 reference frame using the quaternion that expresses the relative orientation between these two frames. The resulting ECI force is then divided by the current vehicle mass to compute the force-induced acceleration in the ECI J2000 frame. This acceleration is added to the total acceleration due to gravity experienced by the vehicle.

Rotational Kinematics Model—The Rotational Kinematics model computes the angular velocities of the LADEE vehicle and the angular velocities of each of the four reaction wheels by integrating the respective angular accelerations using a resettable two-step Adams-Bashforth integrator (see the Translational Kinematics model). This model also computes the quaternion of the vehicle Body frame relative to the ECI J2000 frame and the corresponding ZYX Euler angles in degrees. The quaternion of the LADEE Body frame relative to the ECI J2000 frame is computed using the quaternion discrete propagation method [9]. The angular momentum in the Body and ECI J2000 frames of the complete spacecraft (vehicle + reaction wheels) is also computed by this model.

The propagated attitude quaternion at time t_{n+1} , denoted by $q(t_{n+1})$, is defined as follows:

$$q(t_{n+1}) = \left[\cos\left(\frac{\omega T}{2}\right) \mathbf{I} + \frac{1}{\omega} \sin\left(\frac{\omega T}{2}\right) \boldsymbol{\Omega}_n \right] q(t_n) \quad (13)$$

where $q(t_n)$ is the quaternion at t_n , T is the sampling interval $T = t_{n+1} - t_n$, \mathbf{I} is the identity matrix and $\omega \equiv \sqrt{\omega_x^2 + \omega_y^2 + \omega_z^2}$

$$\boldsymbol{\Omega} = \begin{bmatrix} 0 & \omega_z & -\omega_y & \omega_x \\ -\omega_z & 0 & \omega_x & \omega_y \\ \omega_y & -\omega_x & 0 & \omega_z \\ -\omega_x & -\omega_y & -\omega_z & 0 \end{bmatrix} \quad (14)$$

where ω is angular rate.

Rotational Dynamics Model—The Rotational Dynamics model computes the angular accelerations of the LADEE vehicle. The angular accelerations include the vehicle angular acceleration in the body reference frame and the angular acceleration of each of the four reaction wheels. The rotational equations of motion are defined as follows:

$$\begin{aligned} \dot{\vec{\omega}}_{\text{sat}} &= (\mathbf{I}_{\text{sat}} - \mathbf{A}_{\text{rw}} \mathbf{I}_{\text{rot}} \mathbf{A}_{\text{rw}}^T)^{-1} \dots \\ &[\vec{\mathbf{t}}_{\text{act}} + \vec{\mathbf{t}}_{\text{dist}} - \mathbf{A}_{\text{rw}} \vec{\mathbf{t}}_{\text{motor}} - \dots \\ &\vec{\omega}_{\text{sat}} \times (\mathbf{I}_{\text{rot}} \mathbf{A}_{\text{rw}} \vec{\omega}_{\text{rw}} + \mathbf{I}_{\text{sat}} \vec{\omega}_{\text{sat}})] \end{aligned} \quad (15)$$

$$\vec{\omega}_{\text{rw}} = \mathbf{I}_{\text{rot}}^{-1} \left(\vec{\mathbf{t}}_{\text{motor}} - \mathbf{I}_{\text{rot}} \mathbf{A}_{\text{rw}}^T \dot{\vec{\omega}}_{\text{sat}} \right) \quad (16)$$

where \mathbf{I}_{sat} is the spacecraft moment of inertia that includes the moments of inertia of the reaction wheels, $\vec{\omega}_{\text{sat}}$ is the spacecraft angular velocity along the spacecraft body axis, \mathbf{I}_{rot} is a diagonal matrix that contains the scalar moments of inertia of the reaction wheels along their rotational axes, \mathbf{A}_{rw} is the reaction wheel orientation matrix, $\vec{\omega}_{\text{rw}}$ is a 4x1 vector that contains the angular speeds of the four reaction wheels relative to the Body frame, $\vec{\mathbf{t}}_{\text{act}}$ is the sum of the actuator torques, $\vec{\mathbf{t}}_{\text{dist}}$ is the sum of the disturbance torques and $\vec{\mathbf{t}}_{\text{motor}}$ is the sum of the reaction wheel motor torques.

4. VEHICLE MODEL

The Vehicle model contains the models of the actuators and sensors onboard the spacecraft.

Coarse Sun Sensor

The Coarse Sun Sensor (CSS) model determines the current out of each of the 12 Sun sensors mounted on the external surfaces of the LADEE spacecraft, and is composed of two subsystems.

CSS Geometry—Determines the angle between the boresight of each sensor and the Sun. The model uses the information from the Sun Position model and the attitude of the spacecraft from the Rigid Body Dynamics model.

CSS Hardware—Takes the sensor boresight to Sun angle and uses it as the independent variable to evaluate a quadratic polynomial for each sensor to calculate the microamp reading from the sensor. The output is gated by the whether the Sun is in sight, then measurement noise is added. The final output measurement is limited by the defined min/max output of each sensor.

Inertial Measurement Unit

The Inertial Measurement Unit (IMU) model is a representation of the sensed accelerations and rates experienced by the IMU sensor and consists of three subsystems.

IMU Kinematics—Determines the rotational rates and translational accelerations sensed at the IMU location in the IMU frame. IMU misalignment and the effects of distance relative to the CG are also accounted for in this model.

IMU Hardware—Calculates the measured linear accelerations and rotational rates as observed by the IMU. This calculation computes a moving average of the sensed accelerations and rates and then applies the sensor scaling, bias, and noise (rotational rate include a random walk term) followed by a dynamic response transfer function. The resulting signals are then quantized and limited.

IMU Software—Prepares the actual IMU message. The signals from the IMU Hardware model are multiplied by Δt to produce delta velocities and delta angles which are then scaled into counts. A null message is also created for the output when the IMU is turned off and a message valid flag is also included based on the state of the device.

Star Tracker

The Star Tracker (ST) subsystem models the characteristics of the star tracker used on the LADEE spacecraft. This model does not simulate the imaging, image processing, or the star recognition software. Instead, this model reproduces the effects that degrade the quality of the estimated attitude by taking the truth attitude quaternion, accounting for the effects that limit the star tracker field of view, and adding noise. The model is composed of 5 subsystems.

ST Events—Determines if the star tracker is on or off and gates the output message accordingly. The transition from off to on also introduces an additional delay to the generation of the output message to simulate the effects of attitude acquisition.

ST kinematics—Resolves the Earth, Moon, Sun direction relative to the boresight of the star tracker. Aberration terms due to vehicle velocity relative to the Earth and to Earth

velocity relative to the Sun are computed to produce an aberration axis and rotation angle.

ST FOV Obstruction—Calculates the angles of the Earth, Moon, and Sun in the local star tracker field of view, along with the angle associated with the celestial body projection in the star tracker field of view.

ST hardware—Determines if the star tracker can calculate its orientation by checking if the field of view is impacted by the Earth, Moon, or Sun. The star tracker field of view is modeled by using the rectangular shape of the associated baffle. The orientation is also gated by excessive angular velocity. The measurement is degrade with a noise model provided by the star tracker vendor.

ST software—Assembles the star tracker message for each star tracker, latches and delays the message, and sets the message valid flag as determined in the ST Events model.

Reaction Wheels and Sensors

This model is constructed out of two components, one for the reaction wheels model and one for the Micro-Electro-Mechanical system (MEMs) rate sensor model. The wheel speed is propagated in the Rigid Body Dynamics of the Environment models where it is coupled with the overall vehicle angular rates. The Reaction Wheel model is composed of two subsystems and the MEMs Rate Sensor model is composed of three subsystems. An additional model combines the two components into a single bus message.

Reaction Wheel Software—Models the reaction wheel speed sensor, control software, and electrical output using subsystem blocks for each of the four reaction wheels. These models handle all the reaction wheels in a vectorized form, resulting in a single model representation that passes the signals for each wheel as vectors throughout the diagram. The Wheel Speed Sensor subsystem models the speed sensors with no noise and the measured rate is generated in counts per sample units. The Wheel Control Mode subsystem models the wheel controller state and associated control law for each wheel independently. Speed control, acceleration control, and idle are the three control modes modeled. The Wheel Controller subsystem models the control and estimation performed by the control software within the reaction wheel units. The Wheel Electrical subsystem models the reference voltage and generated motor voltage, current, and power as a result of the commanded control voltage.

Reaction Wheel Hardware—Models the reaction wheel motor and wheel friction terms and returns the net torque and current generated.

MEMs Kinematics—Calculates the rotational rates and angular accelerations experienced by the MEM gyros.

MEMs Hardware—Models the coarse and fine rate measurement process, including noise, random walk, and bias terms. The measured rates are then filtered and quantized. Noise terms associated with temperature are also added to the filtered measurement prior to quantization.

MEMs Software—Provides filtered and unfiltered rates for both coarse and fine measurements along with tunable gyro state and enable parameters.

Reaction Wheel Output—Constructs the reaction wheel assembly output from the Reaction Wheel model and the MEM

Rate Sensor model output into a single reaction wheel message. This model also passes the reaction wheel net torque and net power draw after gating these signals with the state of the reaction wheel (setting the values to zero when the reaction wheel is turned off). This model also provides "hooks" for the test conductor by maintaining the state of the reaction wheel. This includes failed ON or failed OFF states. Each signal in the message can also be overridden with user defined values.

Propulsion

The propulsion model is comprised of models of each of the components of the propulsion system, including two pressurant tanks, two fuel and oxidizer tanks each, pressure regulator, three latch valves, four reaction control system (RCS) units and one orbit control system (OCS). These are modeled to sufficient fidelity to approximate the time-dependent behavior of the system under changing thermal conditions and thrust demands, and include some fault management failure modes, such as pressure regulator leakage and over/under prediction of thrust levels. In this system, thrust is computed from the RCS and OCS modules as a function of time-on, temperature and pressure of the fuel and oxidizer tanks. The thrust modules also compute the demanded mass flux from the system. The demanded mass flux is propagated through the tanks, along with the heat transfer to/from the tanks to estimate the pressure, temperature and mass of the fluid components in each tank. The algorithm for this modeling is similar to that of Purohit and Prickett [5].

5. ELECTRICAL SYSTEM MODEL

Commands—The Commands subsystem of the Electrical model processes flight software command messages and converts them into switch commands. The output of this model is a set of four arrays containing the commands to the Switches model. The order of these arrays is based on the order of the switches as defined in an Interface Control Document (ICD) database. The four arrays represent the following information: new command is ready for processing, commanded state of the switch, commanded on time for the switch, and reset command for current protected switches. The model allows for each current protected switch to have its own reset signal.

Switches—The switches model is divided into functional areas that manage the states of various devices. The fuse states (resettable fuses and fixed fuses), the switch states (manual switches, variable time switches, and fixed time switches) and the board states are all managed. Device switch states (IMU, Star Tracker, ect.) are also managed by this model. An important feature of the Switches model is that the model itself is agnostic to the number of switches and to the order of the switches. The initialization script will parse an ICD database to determine this information. Therefore, when a change is made to the ICD (which was done frequently with LADEE) it is automatically propagated to the model. This feature also makes the electrical system adaptable to new missions because there is no LADEE specific data embedded in the model file.

Solar Array—There are 30 solar panels on the spacecraft which are controlled by 12 solar array (SA) switches. The SA switches can turn OFF or ON a set of solar panels to limit the amount of electrical current that is used to charge the battery. The electrical current of each solar panel is calculated by multiplying the associated solar array switch state with the fractional current output of the panel and then

multiplying that by the maximum output current of the panel. The fractional current output is calculated by using the cosine of the Sun angle (angle between the normal vector to the panel surface and the sun) to the specific solar panel and using a Kelly cosine interpolation table to compute the fraction of the total possible current being produced by the panel at its current orientation relative to the sun.

The Kelly cosine interpolation table is generated by evaluating the Kelly cosine function at specific angles. These angles are selected in the ranges of 0 to 90 degrees with different spacing depending on the changes in slopes in a specific range of the curve. The independent variable for this table is then reordered from largest angle to smallest angle and the cosines of these angles are used as the final table interpolation values for the independent axis (the reordering is performed to provide a monotonically increasing independent variable).

Loads—The electrical Loads model calculates the electrical currents generated by all of the active loads on the spacecraft. The currents are determined by the state of the switches and the corresponding loads, by the contribution of fixed and variable loads, and by the state of the automated thermistors. The currents are summed to represent the total current consumed by the loads on the spacecraft. The currents are also associated back to the related switches and boards to determine if there is an overcurrent condition. A simple switch network is used to determine whether the circuit is closed for the related loads. The network is defined by a master switch in series with a set of switches in parallel with each other.

Battery—There are two parts to the calculations performed by the Battery model. The first part is to compute the net inflow of charge into or outflow of charge out of the battery, thus maintaining the electrical current state of charge of the battery. The second part is to compute the voltage output of the battery along with the current in or out of the battery. To compute the inflow or outflow of battery charge, the electrical current output of the solar panels and the current demands of the loads are used along with a series of efficiency parameters.

Sensors—The electrical Sensors model constructs the electrical system status messages sent to the flight software using the appropriate conversion (example: Amps to Count) defined in the Command and Telemetry Dictionary (C & T). This model also allows a Verification and Validation Engineer to inject faults into the system. The V&V Engineer can override devices, switches, or fuses as failed 'ON' or 'OFF'. For example failing the IMU 'OFF' would command the IMU model to power off regardless of the commanded state from the flight software. This allows the V&V Engineer to ensure the flight software responds appropriately to system failures as defined by mission requirements.

6. THERMAL MODEL

The Thermal model is comprised of two subsystems, the Thermal Power generation model and the Thermal Propagation model.

Thermal Power

The Thermal Power model computes the external thermal power for each spacecraft external node due to the space environment. There is a node for each spacecraft surface panel including the top and bottom of the spacecraft. The sources of environment thermal power are lunar infrared

radiation (IR), lunar albedo, and solar radiation.

Solar Radiation—Calculates power due to solar radiation incident on a spacecrafts surface and defined as follows:

$$\dot{q}_{\odot} = \frac{F_e}{\|\vec{r}_{\odot} - \vec{r}_{sat}\|^2} \quad (17)$$

$$Q_{\odot_i} = \alpha_{s_i} \dot{q}_{\odot_i} A_i \cos \theta_i \quad (18)$$

where \dot{q}_{\odot} is the thermal solar radiation, F_e is the solar constant, Q_i is the power of the i th node, α_s is the surface absorptance, A_i is the area of the panel of i th node, and $0 \leq \theta_i \leq 1$ is the angle between Sun and the surface normal of the i th node. If the Sun is occluded the solar radiation is zero.

Moon Albedo—Calculates power due to solar radiation reflected by the surface of the Moon to the spacecraft. If the spacecraft is in Moon's shadow then the lunar albedo is zero.

$$\dot{q}_{\mathcal{D}alb} = \frac{F_e}{(\|\vec{r}_{\odot} - \vec{r}_{sat}\| - r_{\mathcal{D}})^2} \quad (19)$$

$$Q_{\mathcal{D}alb_i} = \alpha_{s_i} \alpha_{\mathcal{D}alb_i} \dot{q}_{\mathcal{D}alb_i} A_i \cos \theta_i \quad (20)$$

where $\dot{q}_{\mathcal{D}alb}$ is the thermal lunar albedo radiation, $Q_{\mathcal{D}alb_i}$ is the lunar albedo power of the i th node, $\alpha_{\mathcal{D}alb_i}$ is the lunar albedo, $r_{\mathcal{D}}$ is radius of the moon

Moon Infrared Radiation—Calculates power due to lunar Infrared Radiation (IR). The infrared radiation acting on the spacecraft due to the Moon is corrected for the altitude above the lunar surface, , and the angle between each spacecraft surface and the Moon [10].

$$Q_{IR_i} = S_{IR} A_i \epsilon F_G (h_m, \theta_i^m) \quad (21)$$

$$S_{IR} = (\cos \gamma \cos \beta + b_{IR,min}) \dots$$

$$\left[\frac{F_e}{(\|\vec{r}_{\odot} - \vec{r}_{sat}\| - r_{\mathcal{D}})^2} m_{IR} - b_{offset} - b_{IR,min} \right] \quad (22)$$

$$\beta = \sin^{-1} \left[\hat{r}_{\odot} \cdot (\hat{r}_{\mathcal{D}} \times \hat{r}_{\mathcal{D}}) \right] \quad (23)$$

$$\gamma = \cos^{-1} \left[\hat{r}_{\mathcal{D} \rightarrow \odot} \cdot \hat{r}_{\mathcal{D} \rightarrow sat} \right] \quad (24)$$

where $\hat{r}_{\mathcal{D} \rightarrow \odot}$ is unit vector of the Moon to the Sun, $\hat{r}_{\mathcal{D} \rightarrow sat}$ is the unit vector of the Moon to the spacecraft, \hat{r}_{\odot} is the unit vector of the Sun to the Earth, $\vec{r}_{\mathcal{D}}$ is the position of the Moon relative to the ECI frame, $\vec{r}_{\mathcal{D}}$ is the velocity of the Moon relative to the ECI frame, m_{IR} is the slope of the lunar surface radiant intensity from the dark side terminator to the subsolar point, b_{offset} is the lunar surface radiant intensity calibrated offset, $b_{IR,min}$ is the surface IR at the dark side terminator, ϵ is the spacecraft surface emittance, and Q_{IR_i} is the power of the i th node due to lunar IR.

Thermal Propagator

The Thermal Propagator model propagates the temperatures of the lumped thermal mass nodes in time, given heat flux input and heat flux output to space. The lumped mass thermal model is defined in an external text file and multiple thermal model files can be used to define the thermal model at different levels of fidelity. An initialization script reads

the text file containing information about each node and the thermal connectivity among these nodes for a specific lumped mass thermal model and organizes this data to be efficiently used by the simulation thermal propagation engine.

Thermal nodes are identified by an associated ID number. This number is used both inside the thermal model and outside the thermal model. Differing resolution models will vary in the number of nodes, but the node ID associated with a specific element or component on the spacecraft does not change. This allows for other subsystem models to reference specific node IDs. These node IDs are then resolved into array indices that correspond to the location of the specified node in the array that contain the node temperatures maintained in this model. The index of particular node allows for fast access to the values associated with that node, and, since the nodal model cannot change while the simulation is running, the index is fixed for the duration of the run. Node information includes node ID, initial temperature, and capacitance. There are three types of nodes available in the models: diffusion, arithmetic, and boundary. The diffusion nodes represent normal materials with thermal mass (capacitance) and store and release energy over time. The arithmetic nodes represent the surface of material and/or thermal blankets. Arithmetic nodes do not have thermal mass and their temperature represent a steady state balance temperature with the neighboring nodes. Boundary nodes represent constant temperature sources or sinks (such as space at large).

The connection between nodes in terms of heat transfer is called a link. Each link has an associated transport parameter. The transport parameter has different meanings and units depending of the transport mechanism associated with the link. Generally there are three transport mechanism considered in thermodynamics: conduction, convection, and radiation. For the purposes of the design in the LADEE spacecraft, the convection mechanism can be ignored (involves fluids). The transport parameter associated with conduction is called conductance (linear link) and has units of $W/^{\circ}C$. The transport parameter associated with radiation is called radiative exchange factor (radiation link) and has units of $W/^{\circ}C^4$. Each link has an associated link ID which is a number that when positive indicates a conductive link and when negative indicates a radiation link.

7. COMMUNICATION MODEL

The Communications Model reports the state of the transmitter. When the transmitter link is determined to be blocked (for example when the spacecraft is behind the Moon), the transmission ON flag is set to false regardless of the state of the transmitter. By default the receiver is always ON, but, if the link is determined to be blocked, the command reception on signal is set to false.

The state of the selected antenna is also modeled and maintained in this model. The initial state is defined by a tunable parameter. The state of the switches from the Power model of the OMNI and MGA antennas are passed as input to this model, which then determines the final state of the two antennas by also considering the fault state of these devices which is controlled by the test conductor.

This model also computes the state of the link using a user defined method from a pool of 5 available options. The first option (mode 1) is to manually force the link to the ON state. The second option (mode 2) is to manually force

the link to the OFF state. The third option (mode 3) is to allow a user defined table of times for Loss of Signal (LOS) and Acquisition of Signal (AOS) to be used to determine the current state of the link. The fourth option (mode 4) is to allow the model to automatically determine the state of the link based on the current geometry and the Earth visibility from the current spacecraft position. There is also a null mode (mode 0) that allows the link to be ON while not being specifically commanded ON or OFF.

An additional parameter is available in the output bus that collects all of the output signals from this block. This tunable parameter is a trigger mechanism to allow the injections of additional failures in the transmission and reception of commands in the PIL and HIL configurations. For WSIM environment this parameter has no effect.

8. PROCESSOR AND HARDWARE IN THE LOOP

Heritage Software

In order to minimize costs the LADEE flight software leverages previously developed government software from NASA Goddard Space Flight Center (GSFC). In particular, the product line Core Flight Software System (CFS). CFS is a platform independent, mission-independent, reusable flight software environment. Two components of CFS are core Flight Executive (cFE) and Operating System Abstraction Layer (OSAL). Time, event, file and table services are provided to software applications by cFE. OSAL abstracts operating system details so software can run under Linux, VxWorks or other operating systems.

The simulator models and flight software models are auto-coded using the Simulink Code Generator to create cFE compliant applications. OFSW models become tasks which run on the Flight Processor. Simulator models (environment, thermal, power, vehicle) become a "Sim Task" which runs on a separate simulator processor. A template target language compiler developed with Mathworks creates cFE compatible functions for initialization and execution.

A Simulink Interface Layer (SIL) provides a standard interface to all Simulink generated modules. This interface includes a standardized command set (no-op, reset counters, clear queue, etc.) for each task. Bus modules that describe Simulink interfaces become cFE Software Bus compliant data structures for task I/O.

WSIM

The WSIM model consists of the components listed in the preceding sections and is primarily used early in the development process to perform algorithm development, requirements analysis, and unit testing of model performance. The WSIM model does not contain any CFS, cFE or OSAL software.

PIL

The simulator models and flight software models are auto-coded and integrated with the C&DH software to run on the real-time processors to provide a processor-in-the-loop simulator (PIL). The PIL simulator represents a higher fidelity simulator over WSIM because it incorporates all of the C&DH software on an inexpensive "flight like" processor. The interface to the simulator is the same ground telemetry software, Integrated Test and Operations System (ITOS), utilized by the Mission Operations which provides

early development and testing of the command and telemetry interface.

HIL

The highest level of fidelity simulator is the Hardware-in-the-Loop Simulator. On the HIL the flight code runs in real-time on the Flight Avionics Engineering Design Unit (EDU) while the Sim runs on its own dedicated processor. The HIL allows testing of flight software with the Avionics I/O and provides definitive answers on resource utilization. Like the PIL the interface to the HIL is the command and telemetry software utilized in operations.

A mobile version of the HIL known as the Travelling Road Show (TRS) was an EDU integrated with the flight software in a mobile chassis. The Flight Software team members were able to travel to sites where payloads were being developed to test the interfaces before integration of the hardware on the spacecraft. These tests with the TRS allowed confirmation of the ICD and payload requirements and caught defects early in the development process.

Table 1. Overview of different simulators

Simulator	Platform	Description
WSIM	Simulink on Windows, Mac, or Linux computers	<ul style="list-style-type: none"> Models of GNC, Prop, Power, Thermal Used by FSW to generate and test algorithms. Provided to MOS for full sequence verification. Much faster than real time (10-50x) depending on selected fidelity of models and platform.
PIL	PPC750 Processors in Standalone chassis	<ul style="list-style-type: none"> Includes all flight software functionality. Runs on 2 processors. Multiple copies maintained by FSW as inexpensive system for real time software & fault management development. Multiple copies provided to Mission Ops for Training, GDS development, and Operations. Runs in real-time.
HIL	Avionics EDU with simulated vehicle hardware.	<ul style="list-style-type: none"> Highest fidelity simulator includes hardware interfaces. One copy maintained in FSW lab for software & fault management development and characterization. Inexpensive version (no power cards) provided to MOS and I&T. Runs in real-time.

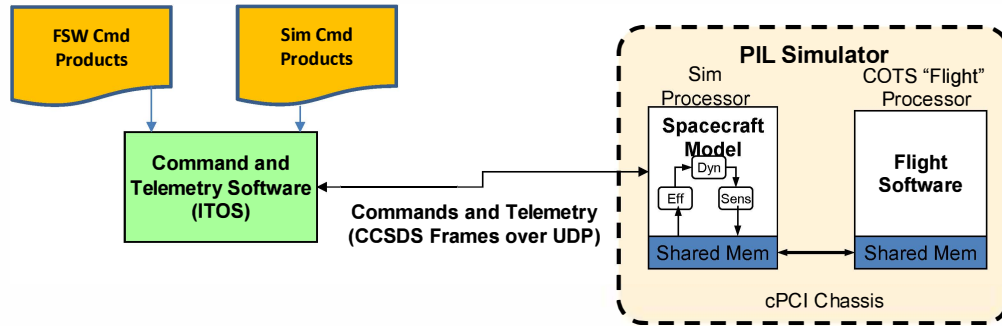


Figure 3. Processor in the Loop Simulator

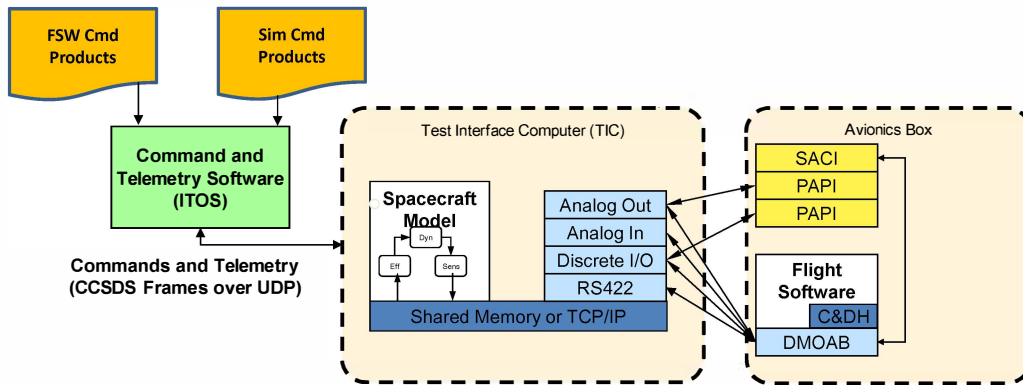


Figure 4. Hardware in the Loop Simulator

9. SOFTWARE DEVELOPMENT AND TESTING

Software Development

The simulator models developed in sections 3-7 were developed to the fidelity necessary to develop and test the flight software algorithms and verify command sequences during mission operations (discussed in further detail in section 10). The fidelity for each simulator model is quantified in requirements documents for both flight software testing and mission operations. The requirements are then tested and verified by unit tests and system integrations tests. Figure 5 illustrates the design cycle used for the LADEE flight software development. The flight software was divided up into four Build/Release cycles and the simulator enhanced at each Build to support the development and testing. Table 2 shows the functionality that was included in each software build and the relevant simulator updates made.

The development environment of WSIM also provided the capability to perform Monte Carlo simulations on the integrated system model. Sensitivity analysis of the Monte Carlo runs provided developers insight to key spacecraft parameters and assisted in the development of operational flight rules. For example, to create a flight rule for the maximum spacecraft momentum before a momentum dump procedure was required, a Monte Carlo simulation was performed. The Monte Carlo simulation determined the maximum spacecraft

moment such that if the spacecraft was in "Safe Mode" for 24 hours (in safe mode operators can not command a momentum dump) the increase in momentum due to lunar gravity gradient would not exceed the control authority of the reaction wheels. Without an integrated system model of the spacecraft environment it would be difficult to quantify this flight rule.

Software Verification and Validation

In general the later in the development process a software bug is found the more expensive it is to fix. For this reason, an early emphasis was placed on the unit test suite infrastructure with an eye toward permanently capturing the unit test effort as a regression test suite. The development effort on LADEE separated testing into two different types: unit tests and system integration tests. Low level requirements were tested at the unit level because it is usually hard to test and debug internal modes and interfaces at a system level. Higher level requirements were tested with an integrated model because it is impossible to test system integration issues at a unit level.

Testing on the LADEE program is enhanced by the modular and layered system architecture. To maintain bidirectional traceability between code/models, requirements, design and test artifacts a system of naming conventions was enforced. For example, for the modeling environment, pertinent naming conventions are:

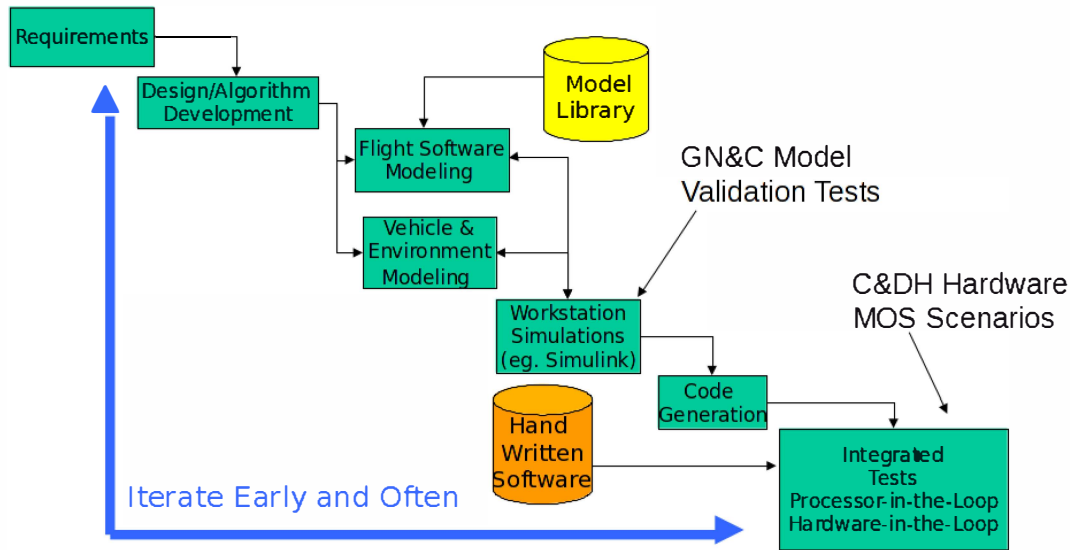


Figure 5. Model Based Development and Testing Approach

Table 2. Build Cycle and Simulator Updates

Build	Simulator Enhancement
<ul style="list-style-type: none"> Attitude Control with Reaction Wheels. Basic and Stored Command. 	<ul style="list-style-type: none"> Rigid Body Dynamics Gravity Model Reaction Wheel Model Star Tracker Model
<ul style="list-style-type: none"> Orbital Maneuvers. 	<ul style="list-style-type: none"> Propulsion Model IMU Model Mass Model Slosh Model Orbital Mechanics
<ul style="list-style-type: none"> Trajectory Correction Maneuvers Health Monitoring Payload Data Nominal End to End Ops 	<ul style="list-style-type: none"> Thermal Model Electrical Model Communications Model
<ul style="list-style-type: none"> Off Nominal Recovery & Safing 	<ul style="list-style-type: none"> Coarse Sun Sensor

- <name>_lib.mdl: Simulink Model library
- <name>_hrn.mdl: Simulink test harness
- <name>_test.m: Test script associated with model library.

Where the <name> included a unique identifier for that model library that could be cross-referenced with requirements and other external documents. Each developer was responsible for providing all the necessary artifacts and developing the unit test suite associated with their model libraries. By adhering to the naming conventions, higher-level regression test suites could interrogate the LADEE model and generate a report of test artifacts. This report was used to exercise the entire test suite under development and evaluate the progress on verifying requirements. Simulink Report Generator was used as a platform to both drive the test suites and capture the resulting information in a manner that provided bi-directional traceability between low-level requirements, models, test suites and metrics.

This test suite also exercises the unit tests on simulator side of the LADEE model. Assumptions are documented as well as ranges of acceptable operation for the model. Depending on the type of model, refinement studies, sensitivity analyses, comparison with analysis, or subsystem performance specifications are included.

The unit test suites are used as early indicators that pertinent requirements are being met, but they do not test the integrated functionality of the software. That is done in a PIL/HIL environment through the use of scenario-based testing. A spacecraft concept of operations was developed that looked at each phase of the life cycle and scenarios were developed to test anticipated operations. These included separation and activation, science operations, orbital maneuvering and fault management related scenarios.

Once the scenarios have been run, a custom Simulink report generator script is used to post process the data and capture the output. The script first reads external imported data, such as requirements spreadsheets and Interface Control Documents. It then processes each of the data files to extract needed variables for the test suite [2].

10. MISSION OPERATIONS

One of the benefits of the LADEE spacecraft simulator in terms of time and cost savings was while the simulator was built by the flight software group, the LADEE Mission Operations group was able to use the PIL, HIL and WSIM with virtually no changes. The three areas in which the LADEE simulator was used in Mission Operations were:

- 1) Development and testing of spacecraft command scripts using PIL and HIL simulators before and during the mission
- 2) Operator training during mission simulations and readiness testing using the HIL simulator before the start of the mission
- 3) Verification of all tactical command sequence files uploaded using WSIM and PIL simulators

For an in depth description of the LADEE Mission Operations see [12].

Development and Testing

There were two types of command sequence files used in operations to command the spacecraft: Absolute Time Sequence (ATS) file and Relative Time Sequence (RTS). As the names imply an ATS would execute a command at a specific spacecraft clock time while a RTS would execute commands a specified number of seconds relative to the last command. RTS files were primarily developed before flight and all acceptance testing was done by testing RTS files using the PIL or HIL simulator. ATS files are verified during the uplink verification process described below.

Additional operations products that utilized the PIL and HIL simulators for tested included templates for maneuvers sequences (lunar orbit insertion and orbit maintenance). During flight, values on the limit checker would occasionally need to be updated and before any change was uploaded it first needed to be verified using a PIL simulation. Lastly, before flight the PIL simulator was used to develop the display pages of the command and telemetry software (ITOS) used by the spacecraft engineering team.

Operator Training

In order to train console operators a series of Simulations and Operational Readiness Testing (ORT) scenarios utilized the HIL simulator in or to provide a flight like training environment [13]. The ORT campaign was the most flight like mission simulation of four critical phases of the LADEE mission: (1) Launch and Activation (2) Checkout and Phasing Loop Maneuvers (3) Lunar Orbit Insertion and (4) Science with an Orbital Maintenance Maneuver. The ORTs were conducted in real-time with operators on console 24 hours a day for 3-5 days depending on the ORT.

One feature of simulator that was critical for operator training was the ability to inject faults into the system. During the flight software development the ability to inject faults was created to ensure the software satisfied requirements on fault tolerance. The mission operations test conductor was able to inject faults using the same feature in order to test console operators response faults. The test conductor has the ability to initialize the start of the simulation with faults (example: scale factor on thrust out of main axial thruster) or inject faults while the simulation is running (example: over-current temperature sensor failure).

As an example of the capability of the simulator to inject faults the following is a list of actual faults injecting during the training campaign.

- Reversed polarity of reaction wheel.
- Increased friction on reaction wheel.
- Multi-bit EDAC error.
- Damaged solar panel
- Temperature sensor failure
- Heater failure
- Large temporary thruster leak
- Faulty RF switch
- Star-tracker switch overcurrent
- IMU failure
- Software reboot
- "Hot" axial thruster burn

Tactical Product Verification

Once LADEE was launched commands to the spacecraft were primarily sent through Absolute Time Sequence (ATS) files. The files generated by the Mission Planning and Sequencing group, in consultation with the Science and Orbit Estimation groups, contained a sequence of commands to be executed by the onboard flight software. Before the command file was uploaded by the Real Time Operations group the ATS was verified by an Uplink Verification Engineer using WSIM.

The benefit of using the WSIM simulator is that it can run approximately 30 times faster than real-time. Thus a typical 3 day Science command sequence would take approximately 1.5 hours to verify. An automatically generated report would be created after a run was completed as evidence that the command sequence would execute as intended and that no flight rules were violated. For ATS containing a spacecraft maneuver (considered a critical event) the ATS was also loaded on the PIL simulator and the maneuver sequence was executed. The PIL simulator was used because it is a more flight like simulator, however it would not be feasible to run an entire ATS using the PIL, as is done with WSIM, because the PIL simulator runs in real time.

11. RESULTS AND CONCLUSIONS

Example Simulator Use Case in Flight

One instance where the simulator proved to be especially valuable in terms of time savings and risk reduction was in debugging unexpected behavior of the star-tracker shortly after activation. LADEE's GNC system merged measurements from attitude rate sensors, an IMU, and two star tracker cameras to estimate the spacecraft attitude. Shortly after launch and activation, performance errors were seen from the state estimator which triggered the on-board fault management to command the spacecraft into Safe Mode due to sudden large changes in estimated attitude and attitude rate.

The initial assessment of the IMU and rate sensors indicated no sudden changes in attitude. Further investigation of the star tracker measurement messages revealed that the star tracker would occasionally delay reporting of its attitude solution or report an inaccurate solution. These delayed measurements were especially likely to occur when a bright object (such as the Sun, Earth, or Moon) entered the star tracker field of view. The longer than expected internal calculations of the star tracker presented "stale" attitude measurements to the State Estimator software.

Once the potential cause of degraded estimator performance was identified, the star tracker model was updated to incorporate a delay model that accurately represents the observed flight data. A variable time delay using biased Gaussian distribution (positive only) to latch to the star tracker messages when a "Bright Object" entered the star-tracker field of view. The operations verification WSIM simulation run was repeated for portion of the mission where estimator anomalies were observed with the new time delay model and compared to flight data.

After it was evident the new star tracker model mimicked the behavior seen from flight data, the GN&C group began to develop a patch for the state estimation algorithms to handle delayed star tracker measurements. While this development went on, the new star tracker model was immediately available to the WSIM and PIL simulators used by Mission Op-

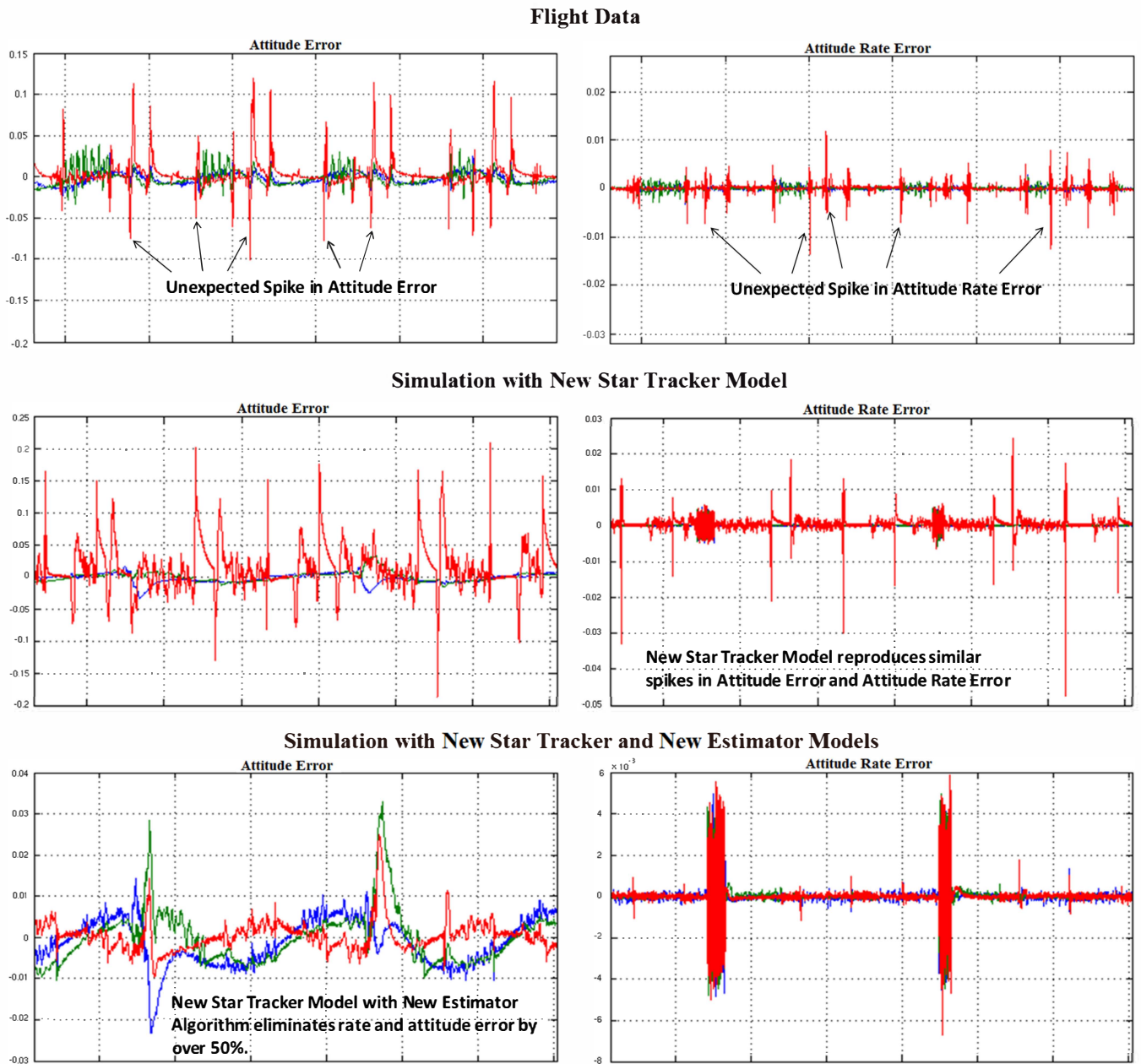


Figure 6. Attitude Error from flight data

erations command verification to detect in advance potential problems and allow mission planners to avoid troublesome spacecraft attitudes.

The first row of Figure 6 shows the Attitude Error and Attitude Rate Error from flight data during an early phase of the mission. The large spikes in the error occurred when a delayed star tracker measurement is updated to the state estimator. After the time delay model was added to the simulator the same sequence of commands was executed on the simulator shown on the second row in Figure 6. Spikes in the attitude error similar to ones observed in the flight data can be seen from the simulated data. After the estimator was updated to account for these delayed star tracker measurements the simulator was once again run on the same sequence of commands. The third row of Figure 6 shows

the spikes in attitude error due to delayed measurements have been eliminated. This was the evidence presented to the mission operations management to support the request upload and updated state estimator.

This example illustrates the strength of the simulator. The same model used to develop and design flightsoftware algorithms is also used to verify and validate requirements, define flight rules, train mission operators, verify uploaded commands, and debug anomalies.

Conclusions

To summarize the benefits of the model-based multipurpose spacecraft simulator developed for the LADEE Mission

- Design & Analysis environment is merged with the FSW Development & Verification environment.
- Elimination of a "translation" phase where FSW engineers interpret and implement algorithm description documents.
- More early and frequent testing of software source.
- Compared to WSIM, PIL and HIL simulators from auto-code create higher fidelity simulations for full onboard software because the executive applications from cFE/CFS are also included.
- Using the same telemetry interface as Operations and I&T allows simulator reuse and early testing
- Ability to test software by injecting "faults" provides a natural interface for the Operations Test Conductor to train mission operators.
- Provides a tool to debug flight anomalies and verify flight commands prior to upload.
- Monte Carlo simulations provide a tool to identify critical spacecraft parameters and develop flight rules.

Future Work

As discussed in the introduction, the Modular Common Spacecraft Bus (MCSB) utilized by LADEE was designed to be agnostic of a specific mission. This means the MCSB can be used for another mission with little to no changes other than the science instruments. The simulator models were designed with the same modularity in mind, so that the models can be used on future missions that utilize the MCSB or on an entirely new spacecraft. For example, the power model can be utilized by imputing a new switch/load map, or the thermal model can be reused by referencing a new node index file. Models of the hardware components may need to be swapped out or parameters modified if a future mission utilizes different hardware such as a different star-tracker or IMU. Current and proposed projects are making use of the simulator models including missions involving cube sats, mars orbiters, rovers, and landers. In order to maximize the use of the multi-purpose simulator for future projects a rigorous comparison of the simulator models with LADEE flight data is currently being performed

ACKNOWLEDGMENTS

The authors would like to thank the members LADEE flight software team at NASA Ames for their contributions to development of the LADEE simulator as well the many LADEE spacecraft subteams for their valuable input in making the simulator better represent reality.

REFERENCES

- [1] B. Hine, S. Spremo, M. Turner, and R. Caffrey. "The Lunar Atmosphere and Dust Environment Explorer Mission," Proceedings of the 2010 IEEE Aerospace Conference, Big Sky, MT, March 6-13, 2010
- [2] K. Gundy-Burlet. "Validation and Verification of LADEE Models and Software," 2014 Workshop on Spacecraft Flight Software, Pasadena CA, December 16-18, 2014.
- [3] NASA JPL FTP site: <ftp://ssd.jpl.nasa.gov/pub/eph/>
- [4] O. Montenbruck and E. Gill. Satellite Orbits: Models, Methods, Applications. Berlin: Springer, 2005.
- [5] G. P. Purohit and R. P. Prickett. "Modeling of the Intelsat VI Bipropellant Propulsion System", AIAA 93-2518.
- [6] P. L. Seidelmann, et al. "Report of the IAU IAG Working Group on Cartographic Coordinates and Rotational Elements: 2006". Celestial Mechanics and Dynamical Astronomy, 98 155-180, 200.
- [7] D. A. Vallado. Fundamentals of Astrodynamics and Applications. Hawthorne, CA: Microcosm Press. 2007. pp 192.
- [8] R. B. Roncoli. "Lunar Constants and Models Document." Jet Propulsion Laboratory, JPL D-32296. 2005
- [9] J. Wertz. Spacecraft Attitude Determination and Control. Norwell, MA: Kluwer Academic. 1978.
- [10] J. Wertz. Space Mission Analysis and Design: 3rd Edition. Netherlands: Springer. 1999. pp 908.
- [11] H. Klee and R. A. Klee. Simulation of Dynamic Systems with MATLAB and Simulink. Boca Raton, FL: CRC Press. 2011.
- [12] M. D'Ortenzio, J. Bresina, A. Crocker, R. Elphic, A. Hawkins, R. Hunt, B. Owens, L. Plice, and L. Policastri, "Operating LADEE: Mission Architecture, Challenges, Anomalies, and Successes," 2015 IEEE Aerospace Conference Proceedings, March 7-14, 2015.
- [13] Brandon D. Owens and Alan R. Crocker, "SimSups Loop: A Control Theory Approach to Spacecraft Operator Training," 2015 IEEE Aerospace Conference Proceedings, March 7-14, 2015.

BIOGRAPHY



from Stanford University.

Nathaniel Benz is a member of the Common Avionics and Software Technology group at NASA-Ames Research Center and was previously a member of the flight software and mission operations groups for the LADEE mission. He graduated with a B.S. in Mechanical Engineering from Rochester Institute of Technology and a M.S. in Computational and Mathematical Engineering



ing, and supporting business development activities.

Danilo Viazzo received his B.S. in Aerospace Engineering from the California State Polytechnic University, Pomona in 1987 and his M.S. in Aero/Astro Engineering from Stanford University in 1989. He is currently a technical director at Cummings Aerospace. His current professional activities involve developing simulations, performing system engineering, and supporting business development activities.



Karen Gundy-Burlet is the group lead of the Common Avionics and Software Technology at NASA-Ames Research Center and was the flight software lead for the LADEE mission. Dr. Gundy-Burlet graduated with a B.S. in Mechanical Engineering from U.C. Berkeley and obtained M.S. and Ph.D. degrees in Aerospace Engineering from Stanford.