Satellite Simulator for Verification of Mission Operational Concepts in Pre-Phase A Studies

7th International Conference on Systems & Concurrent Engineering for Space Applications

- SECESA 2016 -

5-7 October 2016

Universidad Politécnica de Madrid (UPM)
Spain

Ronan Arraes Jardim Chagas[1], Arcélio Costa Louro[2], Fabiano Luis de Sousa[3], Willer Gomes dos Santos[4]

[1] ***Instituto Nacional de Pesquisas Espaciais (INPE)***
Av. dos Astronautas, 1758, Jd. Granja, CEP: 12227-010, Space Systems Division
São José dos Campos, SP, Brazil
Email: ronan.arraes@inpe.br

[2] ***Instituto Nacional de Pesquisas Espaciais (INPE)***
Av. dos Astronautas, 1758, Jd. Granja, CEP: 12227-010, Space Systems Division
São José dos Campos, SP, Brazil
Email: arcelio.louro@inpe.br

[3] ***Instituto Nacional de Pesquisas Espaciais (INPE)***
Av. dos Astronautas, 1758, Jd. Granja, CEP: 12227-010, Space Systems Division
São José dos Campos, SP, Brazil
Email: fabiano.sousa@inpe.br

[4] ***Instituto Nacional de Pesquisas Espaciais (INPE)***
Av. dos Astronautas, 1758, Jd. Granja, CEP: 12227-010, Space Systems Division
São José dos Campos, SP, Brazil
Email: willer.gomes@inpe.br

INTRODUCTION

We introduce in this paper a satellite simulator designed for verification of mission operation concepts during Pre-Phase A studies. In its present form, it can be used to verify the mission operational concept, as reflected in the dynamics of data and power usage inside the spacecraft, as well as data exchange between the spacecraft and ground stations.

The simulator consists of 4 modules that are described as follows. The orbit module propagates the satellite orbit showing the satellite position in both 2D and 3D views. This module is also responsible to compute the Sun position, to check if the satellite is in eclipse, and to verify if there is line-of-sight between the spacecraft and ground stations. The equipment module consists of a list containing all satellite devices together with their power usage patterns. This module outputs the current status of each device (turned on or off), the actual power consumption, and the memory usage rate. The power subsystem module computes the generated power in each solar panel and combines the information with the bus power consumption to provide the battery state of charge. Finally, the OBDH subsystem module computes the available mass memory considering the memory usage rate of the devices and the transmission rate of data to the ground stations.

The simulation core was coded in Julia language, which is an interpreted language providing an easy yet powerful integration with C++. The core handles all computations related to the simulation and continuously sends TCP/IP packages to the graphical user interface (GUI) responsible to show the results to the user. The GUI was coded using Qt and C++.

The simulator has been recently used successfully in studies carried out at INPE's Space Mission Integrated Design Center (CPRIME). It clearly enhanced the dynamic systemic awareness of the design team, also providing a more complete and interesting understanding of the mission. The code of the entire simulator is licensed under GPL and its release to the general public has been willingly planned.

ABBREVIATIONS

CPRIME	INPE's Space Mission Integrated Design Center
ECEF	Earth-Centered, Earth-Fixed
DoD	Depth of discharge
EOL	End-of-life
GUI	Graphical User Interface
SAG	Solar array generator
WGS	World Geodetic System

SIMULATOR DESIGN

The purpose of the simulator is to use the data from the mission analysis database of the design facility to configure the internal models in order to validate all the mission operation concepts. Hence, we defined the following guidelines during its development:

- The software must simulate the space environment and the required subsystems in such a manner that the designers can easily see the overall operation concept of the mission, improving the level of confidence in the proposed solution.
- It also must show the information in a friendly way for the stakeholders.
- It must be adaptable to be used in different missions with minor adjustments only.
- It must use only free software tools, since we are planning to release the entire simulator source code to the general public.
- The design must be simple to build and to maintain in order to avoid high utilization of manpower.

The simulator was divided in two parts: the simulation core and the simulator GUI, as it can be seen in Fig. 1. The former contains all the models and algorithms, handling the simulation computations. The latter is responsible to send commands to the simulation core, such as increase or decrease the simulation step, and to show to the user the simulation data received from the core in a friendly way. Furthermore, it is also possible for the core to send the simulation data to other GUIs simultaneously, as shown in Fig. 2. Hence, each specialist can see the simulation observing only what matters for his subsystem. Notice, however, that in this case only one GUI can send commands back to the simulation core.

The simulation core was coded in Julia language, whereas the simulation GUI was coded using C/C++ and Qt. The GUI main window can be seen in Fig. 3.

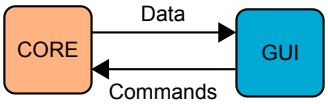

Figure 1: Simulation core and simulator GUI.

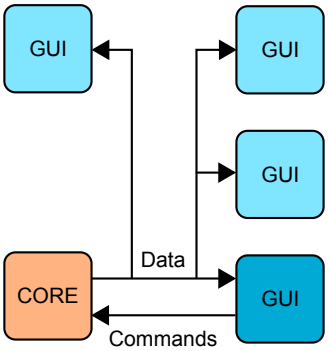

Figure 2: Simulator connected in a network.

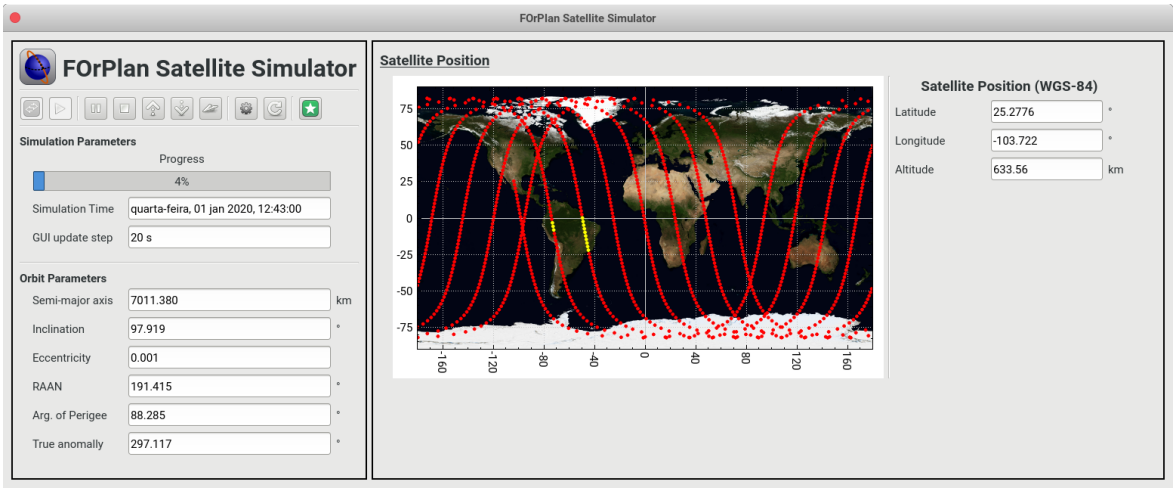

Figure 3: GUI main window.

IMPLEMENTED FEATURES

In this section, it is briefly listed all the features and modules that are currently implemented in the simulator.

Space Environment

The space environment module is responsible to:

- Propagate the satellite orbit using the algorithms obtained from [1] and [2];
- Compute the satellite position in many coordinate reference frames (J2000, ECEF, WGS-84, etc.);
- Compute the Sun position vector represented in the body reference frame as described in [3];
- Check if the satellite is inside the penumbral or umbral regions as shown in [4];
- Check if the satellite Nadir is pointing towards a specific country; and
- Check if the satellite is inside the visibility circle of the configured ground stations.

In the simulator GUI, the satellite position can be seen in both 2D and 3D views, as shown in Figs. 3 and 4.

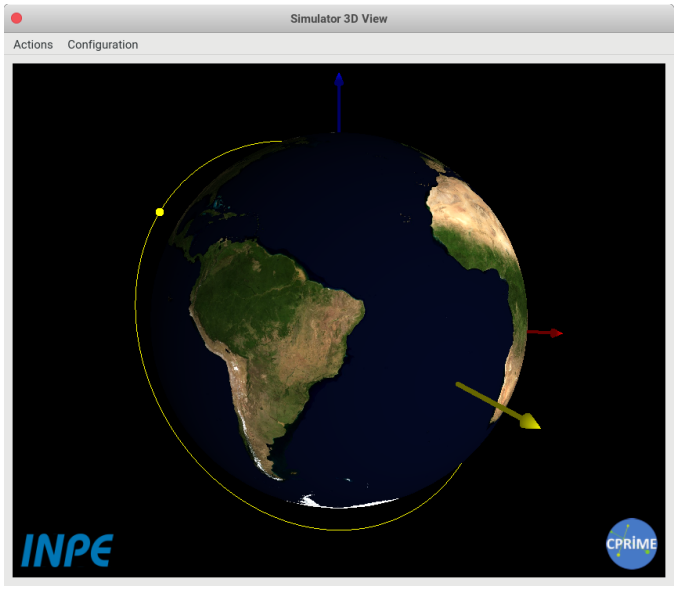

Figure 4: 3D view of the satellite position in the simulator GUI.

Equipment Module

The equipment module is responsible to simulate the behavior of the satellite equipments in terms of power usage and memory consumption. The user defines for each equipment the power consumption for both stand-by and operating modes and the equipment memory usage rate during the operation. Furthermore, it is also necessary to define how or when the equipments are turned on or off. In its current stage of the simulator, there are nine types of utilization patterns:

1. `EQUIPMENT_ALWAYS_OFF`: The equipment is always off.
2. `EQUIPMENT_ALWAYS_ON`: The equipment is always on.
3. `EQUIPMENT_ALWAYS_ON_TRANSIENT`: The equipment is always on but it is possible to define a transient in terms of power consumption. This is used to simulate, for example, the power consumption behavior of reaction wheels at the beginning of the mission. In this case, the wheels are used to remove the tip-off angular rate of the satellite body, which usually consumes much more power than that of the normal satellite operation.
4. `EQUIPMENT_ON_SUNLIT`: The equipment will be turned on when the satellite is not on eclipse.
5. `EQUIPMENT_ON_ECLIPSE`: The equipment will be turned on when the satellite is on eclipse.
6. `EQUIPMENT_ON_OVER_LOCATION`: The equipment will be turned on if the satellite is over a specific location.
7. `EQUIPMENT_ON_GSTATION`: The equipment will be turned on if the satellite is inside the visibility circle of a ground station.
8. `EQUIPMENT_ON_PERIODIC`: The equipment will be turned on periodically.
9. `EQUIPMENT_FUNCTION`: The user must provide a function written in julia that returns whether the equipment is turned on or off.

The equipment window in the GUI can be seen in Fig. 5, and the configuration file for the equipments can be seen in Fig. 6.

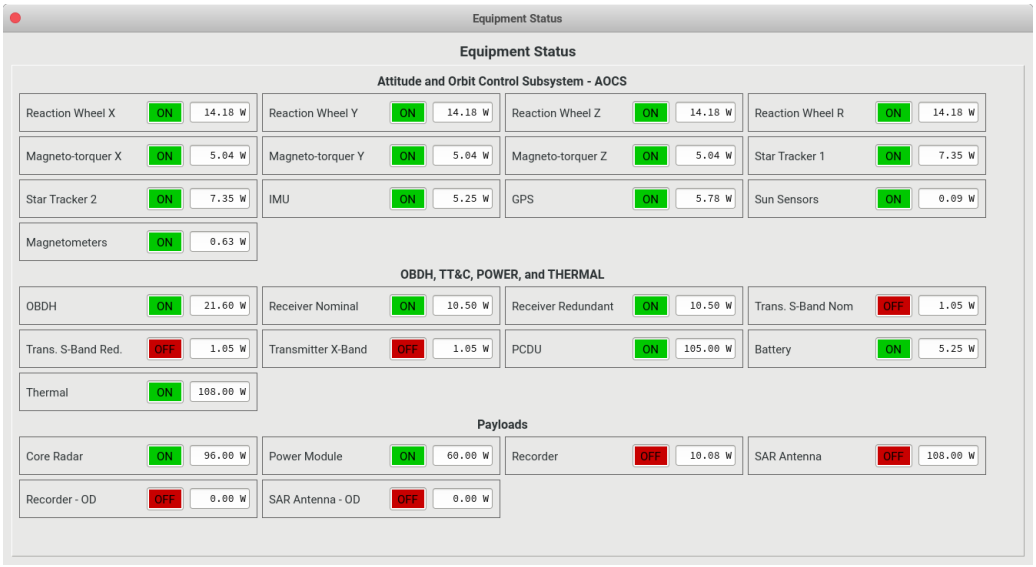

Figure 5: Window showing the equipment status in the simulator GUI.

Power Subsystem Module

The power subsystem module is responsible to verify if the generated power by the SAG and the amount of charge stored in the batteries are sufficient to attend the mission requirements. It obtains the Sun vector represented in the body coordinate frame to compute the theoretical irradiance in each configured solar panel.

The solar panels are defined by its type: fixed or rotating. In case of a rotating solar panel, the user must define the rotation axis and the simulation will position the surface to maximize the Sun irradiance. On the other hand, for fixed solar panels, the user must define the solar panel normal vector represented in the satellite body reference frame. Additionally, it is necessary to define the total area of each solar panel and its efficiency.

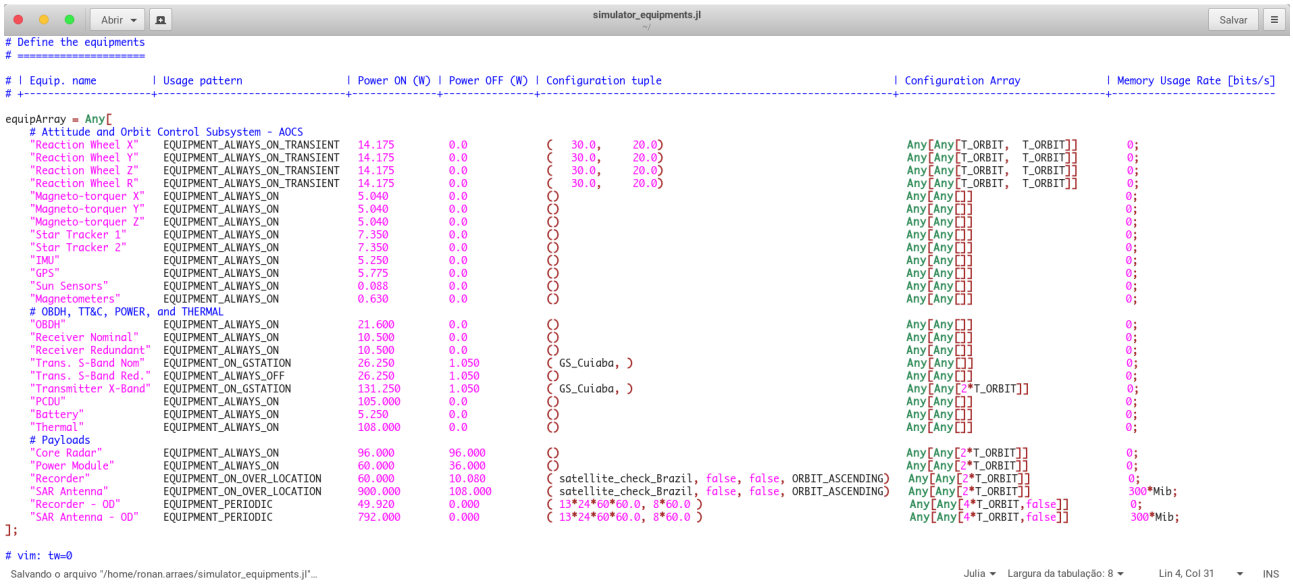

Figure 6: Example of the equipment configuration file.

With all this information, the power subsystem receives from the equipment module which equipment is on, computes the total power consumption of the satellite, computes the generated power by all the solar panels, and, finally, simulates the battery charge dynamics. Hence, if the battery DoD stays always below a limit, then the power subsystem design is validated. Notice that it is possible to provide a degradation factor to simulate the spacecraft condition at the EOL.

The power subsystem window in the GUI can be seen in Fig. 7.

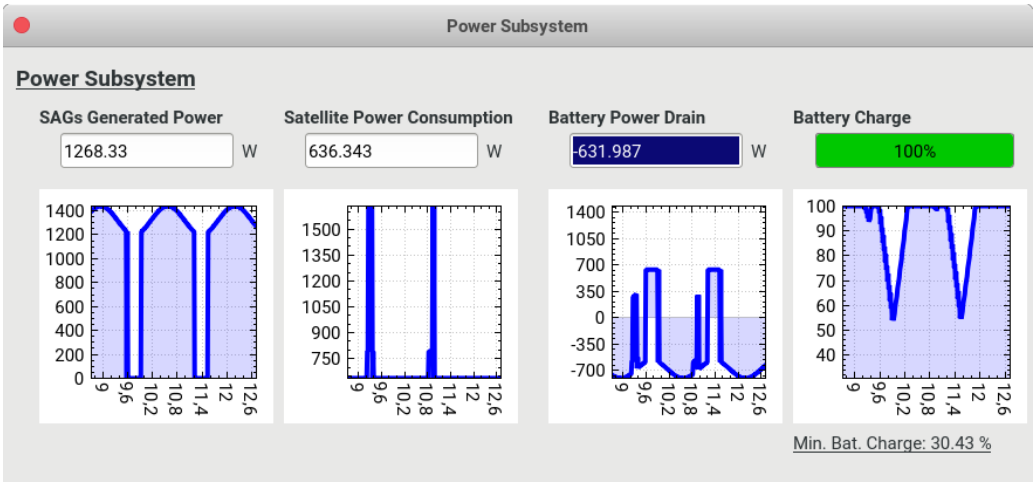

Figure 7: Power subsystem window in the GUI.

OBDH and TT&C Subsystem Modules

The OBDH and TT&C subsystem modules are responsible to validate the mass memory design. Each configured equipment defines its data generation rate during the operation, which is used by the OBDH subsystem to simulate the usage of the internal mass memory. Furthermore, the TT&C subsystem module simulates the data transfer to the ground stations by downloading the stored information with a configured mean transfer rate when the satellite is within the visibility circle. The latter is solely defined by the station geographic coordinates and the minimum elevation angle in which the communication is possible. If the maximum memory usage is lower than the total memory available, then the mass memory design is validated.

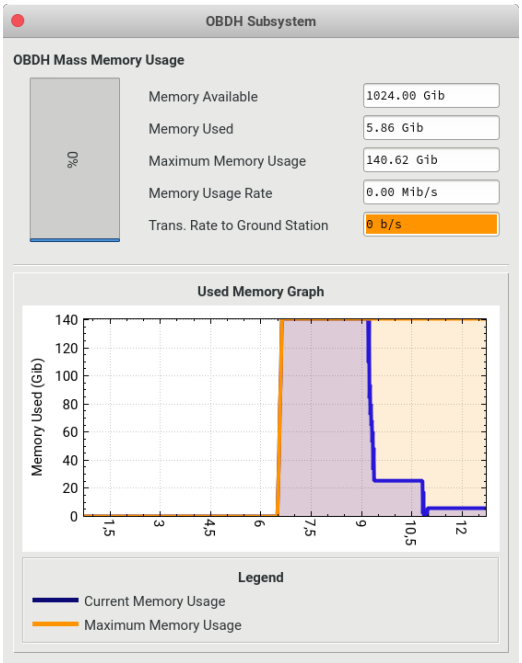

Figure 8: OBDH subsystem window in the simulator GUI.

The OBDH subsystem window in the simulator GUI can be seen in Fig. 8.

FUTURE WORKS

The simulator is still considered in early development stage and many features are planned to be implemented. Among those features, the most notable are described as follows.

- **AOCS Dynamics**: We are planning to add a simple rigid body model to validate the AOCS design in terms of reaction wheel momentum dumping by magneto-torquers and/or thrusters, and initial satellite pointing considering the launcher tip-off angular speed.
- **Flight plan:** It was requested to implement the capability to define a flight plan for the mission. In this case, the user will be able to create a script that will be sent from the configured ground stations to schedule actions such as turn equipments on or change the satellite pointing.
- **Model-based configuration**: The simulator shall be capable to configure every parameter based on the models in the mission database. Hence, it must send SQL commands to fetch all the necessary information, and configure itself automatically to simulate the studied mission. In its current form, the configuration is done manually by the simulation engineer.
- **Satellite constellations**: The simulator will be extended to handle satellite constellations.
- **Coverage analysis**: It is necessary to add the capability to compute the coverage for imaging missions to validate the payload and orbit designs in terms of swath and revisit.

CONCLUSIONS

In this work, we presented a simulator developed to improve the level of confidence in the mission operation concepts. Although it is still in an early development stage, it has already been successfully used in studies performed by INPE's CPRIME, enhancing the dynamic systemic awareness of the design team. The simulator is being built using free software tools only and we are planning to release its source code to the general public under GPL.

ACKNOWLEDGMENTS

The first author acknowledges the Instituto Nacional de Ciência e Tecnologia em Estudos do Espaço (INEspaço) for the financial support.

REFERENCES

[1] J. R. Wertz, *Spacecraft Attitude Determination and Control*. Boston, United States of America: D. Reidel Publishing Company, 1978.

[2] D. A. Vallado, *Fundamentals of Astrodynamics and Applications*. Hawthorne, CA, United States of America: Microcosm Press, 2013.

[3] U. S. Naval Observatory and Royal Greenwich Observatory, *The Astronomical almanac for the year 2000*, 1999.

[4] C. R. O. Longo and S. L. Rickman, "Method for the calculation of spacecraft umbra and penumbra shadow terminator points," Lyndon B. Johnson Space Center, Houston, Texas, USA, Tech. Rep. 3547, 1995.