Operational Nanosatellite Simulator: would it be applicable to Cubesat missions?

***Italo P. Rodrigues*[(1)], *Ana M. Ambrosio*[(2)], *Ronan A. J. Chagas*[(3)]**
[(1)] *Instituto Nacional de Pesquisas Espaciais, São José dos Campos - SP, italoprodrigues@gmail.com*
[(2)] *Instituto Nacional de Pesquisas Espaciais, São José dos Campos - SP,* aambrosio27@gmail.com
[(3)] *Instituto Nacional de Pesquisas Espaciais, São José dos Campos - SP, ronan.arraes@inpe.br*

Keywords: Space Systems Simulation; Space Systems Modeling; Operational Simulator; Simulations; Nanosatellite

Nanosatellite, including the CubeSat platform, is a low-cost and fast alternative for human resources training and technological validation in the space area. Over the years and with increasing numbers of nanosatellite missions, this alternative has demonstrated satisfactory. Also, it has attracted the attention of the space industry, involving more significant financial investment than academics CubeSat missions. After launch, the nanosatellite must complete its mission. Early loss of a company-built nanosatellite can be a considerable loss, mainly in terms of reputation damage. Developing tools to support the operation of CubeSat after its launch can be an excellent solution to this problem. In this sense, ground support tools, and operational satellite simulator can play an essential role by providing a mechanism that may mitigate the risk associated with satellite operation. This kind of simulator represents the final design solution of the satellite, and its development is expensive and time-consuming. On the other hand, it may improve the quality and lifespan of the operational phase. Because of the high development costs of such simulators and the low cost of CubeSats development, the operational satellite simulator has not been used for CubeSats. However, this kind of tool will strongly benefit the development and lifecycle of nanosatellite missions as it does for the long-life satellites with a model reuse approach from the design phase to the operation. Based on the lessons learned of developing the CBERS4 Satellite Simulator, this paper presents a simulator architecture, discusses model fidelity, issues of implementation, verification, and verification, showing how the project aspects can be adapted and reused to benefit also CubeSat missions. It is also discussed advantages of using an operational simulator for CubeSat missions, such as more confidence in the operation; support to adapt the operation to the current state of the nanosatellite, in case of malfunctioning or aging and anomalies, or both, aiming to increase its lifespan; moreover the trade-off of subsystems resources as memory availability, power supply, communication channel.

1. Introduction

Nanosatellites, especially CubeSats, have been changing the traditional space exploration scenario. The CubeSat is a miniaturized satellite standard built with cubes of 10 cm (1U) edge, which can be increased to larger sizes, for example, by placing two cubes together forming a CubeSat 2U, and so on. [1, 2, 3]

Firstly, CubeSats were conceived as a low-cost educational solution to enable the training of space scientists and engineers, as well as providing a platform for testing

and space qualification of new technologies [3,4,5].

However, in the last few years, the space industry has been showing a greater interest in the development of CubeSat, due to the potential high value related to scientific return and commercial revenue. One of the most formidable examples to illustrate the commercial and scientific interest in CubeSats is Planet (formerly Planet Labs), which has launched, since 2013, more than three hundred CubeSat-based satellites for Earth observation. [6,7,8,9,10,11]

There is a forecast of 1000 CubeSats launches in 2021 [1], which demonstrates an increase in the number of launches each year.

Even with advances in miniaturization, there are several technological challenges to be explored in CubeSats missions, such as [2, 8, 13, 14]: noise reduction in miniaturized components; attitude and orbit control; higher system reliability to guarantee reasonable lifetime in orbit; deep space telecommunications, power generation and propulsion; and Earth observations, space weather, astrophysics.

The increase in complexity in CubeSat missions requires the use of tools that allow mission control to ensure that experiments carried out on board reach their lifetime, as discussed in [15].

The use of a simulator during the lifecycle development may address several benefits, such as the risk and cost decreasing. The simulators enable system analysis and definition; software verification and validation, ground support equipment development; test procedures development; operation team training; operations procedures verification, and validation. [16]

This paper is organized as follows: Section 2 describes the objectives and general architecture of the operational satellite simulator. Section 3 discusses related work concerning tools for the operational phase after the CubeSat launch. The features of the CBERS-4 operational satellite simulator (SimCBERS) are presented in Section 4. Section 5 discusses the SimCBERS reuse regarding CubeSats missions. Finally, section 6 presents the conclusions.

2. Operational Satellite Simulator

An operational simulator is a software used to support operations activities after the launch of the satellite, which main objectives are to [16,17]:

- Develop and validate the Flight Control procedures.
- Train the flight control personnel.
- Validate the satellite control center software.
- Test and fix bugs of new versions of on-board software before it is uplinked to the satellite.
- Support troubleshooting and maintenance during operations, if necessary.

According to the ECSS Technical Memoranda [16], the architecture of an operational simulator comprises the following components, as illustrated in Figure 1:

- Virtual System Model: reflects the functions and behavior of the complete system at the level required to support the respective analysis and comprises the simulation infrastructure and the models of the space system.
- Simulation infrastructure: is an essential part of the architecture to allow model reuse to work smoothly.
- Spacecraft Model: is composed of the models that represent the parts of the satellite. These models must process the remote controls and generate the telemetry.
- Mission Control: is the model that generates the telecommand to send to spacecraft

and process the telemetry generated by the satellite.

- Facility M&C (Monitoring and Control): it is the interface between the user and simulator.
- Ground Models: simulates the visibility to the spacecraft, including acquisition and loss of signal and space to a ground link.
- Environment Models: calculates the orbit(s)and indicates ground station visibility.
- Spacecraft Dynamics Models: describes the dynamic behavior of spacecraft in space, for example, a rigid body model.

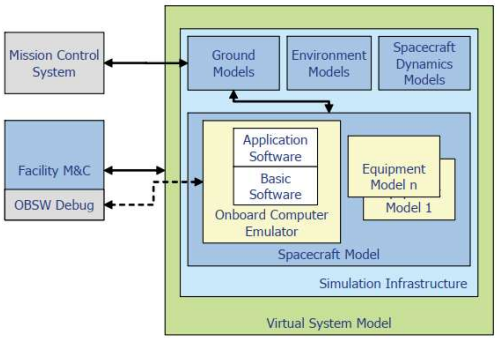

Figure 1: Generic Architecture of an Operational Simulator.

3. Related Work

Concerning to the existence of software to support the operation phase of Cubesats, in the literature, we found reports about tools to monitor the satellite, to analyze data and control, to test and few describing a satellite simulator.

In [18,19] are presented two tools developed to monitor the CubeSats during their operations. Despite these tools enabling the monitoring of CubeSat, they do not perform a simulation. One of these tools supported Prometheus CubeSat operations. It was developed by the Los Alamos National Laboratory, in their ground station context. This software tool performed the following functions: automated operations, orbit propagation, Two-lines elements (TLE) retrieval, state of health logging, and display [19]. The other tool, the MOVE-II Operations System, was developed to support the operation of CubeSat MOVE-II. The main function of the Move-II Operations System is to allow operators to analyze the data and control the satellite. The scope of the work did not consider the implementation of the operational simulator. However, the work presents relevant insight regarding the lessons learned in software development, such as the use of automatic tests to enable reusability and maintainability. [18]

The COSMOS (Comprehensive Open-architecture Space Mission Operations System) is a suite of software and hardware tools developed to improve the satellite operations. The COSMOS tool includes various functions, such as mission planning and scheduling, data management and analysis, simulations (including the operational testbed), ground network control, payload operations and flight dynamics. Moreover, the COSMOS Operations Test-Bed (OTB) uses an open-source system architecture that integrates (i) the actual Mission Operations Center (MOC) control tool, or MOST; (ii) the Ground Station Simulator (GSS); (iii) the Satellite System and Subsystem Simulator (SSS); (iv) the Test Bed engine (TBE); (v) the testbed controller tool (TBCT); and (vi) the testbed controller user interface for command scripting testing, personnel training, mission rehearsals and anomaly resolution. [20,21]

In [15], it is presented the NASA Operational Simulator for Small Satellites

(NOS3). Like a traditional operational simulator, it enables exercising operation of the simulated spacecraft, in nominal scenarios, using the ground software command and telemetry databases. It is a simulator with virtual models to enhance small satellite programs, and that was developed for the Simulation-to-Flight 1 (STF-1) 3U CubeSat mission. NOS3 integrated part of COSMOS into its architecture.

At least three initiatives [15,20, 21] in the literature reported the architecture of the operational simulators to CubeSats, in which, the models require high fidelity. However, no details of the modeling of the satellite subsystems are presented in the papers. Furthermore, these papers do not show whether there are tests to inform the model (power supply, communications, dynamics) correctness, or even whether there are model specifications which indicated what must be implemented and then tested.

4. SimCBERS: an operational simulator for a traditional satellite

SimCBERS is the operational simulator developed for the CBERS-3&4 satellite. CBERS (China-Brazil Earth Resources Satellite) is a long-term effort, between Brazil and China, to design, build, assemble, integrate and operate remote sensing satellites. The use of images obtained by CBERS enabling some analysis, such as controlling deforestation and burning in the Amazon Forest, monitoring water resources, and agriculture. [22,23]

Over the years, other operational simulators have been developed at INPE [20]. However, SimCBERS is the most sophisticated simulator delivered. Besides, to simulate variations in telemetry values similar to the real ones, SIMCBERS allows the simulation of anomalies scenarios. These anomalies can be set during a simulation run that has already started. Some of the SimCBERS requirements are [24]:

- The simulator shall allow the simulation-time step differs from one model to another, i.e., the rate shall be independent and parameterized for each model.
- The simulator shall allow configuring space system elements, namely, satellite, earth stations and space environment, which represent different solutions of space systems design.
- The simulator shall allow all the information that is part of a configuration to be parameterized and can be changed during a simulation run.
- The simulator shall perform on-the-fly parameter changes.
- The simulator shall be scalable, in the sense that models can be replaced over the simulator lifetime.
- The simulator shall reproduce the operational modes of all the subsystems in terms of power consumption
- The simulator shall reproduce the thermal variation in all the measurement points of the satellite.

The SimCBERS design solution consisted of a modular architecture designed to allow the software reuse for future simulators. Figure 2 shows the two-levels simulator: infrastructure and models.

The model level is divided according to the software technique adopted in the model implementation: DLL, C++, and XML. Table 1 describes these types.

In the current version, SimCBERS is configured for the CBERS-4 satellite, simulating the behavior of the 15 satellite subsystems, with 408 inputs (telecommand) and 629 outputs (telemetry). All subsystems are simulated near real-time.

The XML models type implements the behavioral models of the subsystems, described in terms of cause and effect rules, illustrated in Figure 3, and dynamic models described using mathematical equations. The parameters (Param A, Param B,

…, $Param_n$) may be remote control (TC), switch (SW), telemetry (TM), operating modes (OM) or working state (WK) of a subsystem or a piece of equipment, and external parameter (EX). These parameters may represent any satellite, which different functions, sizes, modes.

The SimCBERS infrastructure executes interpretation and evolution of models over time, the mechanisms of the software to allow for the simultaneous execution of threads or processes, triggering and canceling timers, dynamically changing values.

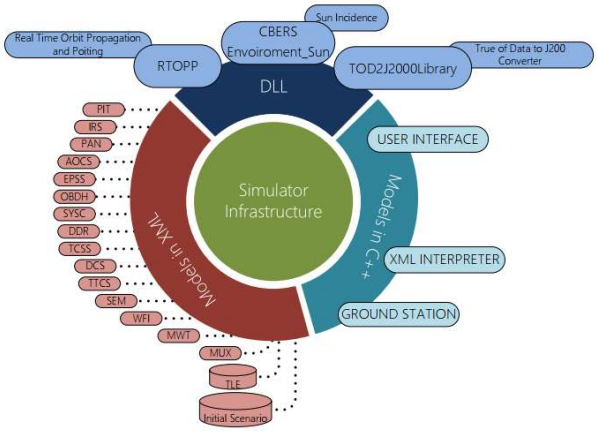

Figure 2: SimCBERS modular architecture [25].

Table 1 – Description of SimCBERS model types. [24,26]

Model Type	Description	SimCBERS Models
DLL (Dynamic-Link Libraries)	It contains the most computationally expensive models. The models can be implemented in a language-independent of the rest of the simulator, then converted into a DLL. The interfaces need to be well defined. Models will require adaptations if another satellite is simulated.	Real-time Orbit propagation and Pointing (RTOPP) Environment models True data to J200 converter
Models in C++	It includes software components not related to neither the satellite nor the infrastructure.	User interface and its facilities, ground station model and the XML interpreter
Models in XML	These are the models that represent the satellite to be simulated. These models can be easily modified, as they were designed in cause-effect rules and mathematical equations.	Models of the satellite service module, such as on-board computer and data handling (OBDH), attitude and orbit control (AOCS), power (EPS), TTCS (Telemetry, Tracking, and Command), among others. Models of satellite payload, such as MWT, MUX, WFI, DDR, DCS.

Preconditions			Effects		
Param A	Param B	Param C	Param X	Param Y	Param Z
a1	b1	c1	x1	y1	z1
a2	b2	> c2	x2	y2	z2
<= a3		-	x3	-	Z+C

Equivalent to:

```
if (A == a1 and B == b1 and C == c1)
   then (X = x1, Y = y1, Z = z1)
if (A == a2 and B == b2 and C > c2)
   then (X = x2, Y = y2, Z = z2)
if (A <= a3 and B == b2)
   then (X = x3, Z = Z+C)
```

Figure 3: Cause-effect table. [27]

In the execution, a trigger is required to modify the behavior of a model. This

trigger can be a TC or a monitored variable in other subsystems (EX). Triggers change the behavior of a switch (SW), which, consequently, will update the value of telemetry (TM). These changes, therefore, modify the operating modes (OM) of a subsystem and equipment levels, its working state (WK) parameters providing a complete mapping of valid, invalid, and foreseen data and unforeseen operating modes. [27]

5. Discussions on adopting an operational satellite simulator in Cubesat missions

Due to the limited transmission capabilities of a CubeSat, its operation needs to consider the visibility window for receiving on-board data and sending telecommands. Therefore, mistakes in operations are not tolerated to keep the Cubesat in good health. The operational simulator can play a fundamental role in turning safer CubeSat operations.

However, there are several features and issues, sometimes incompatible, to be balanced before deciding to produce an operational satellite simulator for a Cubesat mission. The topics highlighted below present some mutual shortcomings (SC) of the simulators presented in this article, and that could prevent the use of an operational simulator for CubeSat missions:

- SC1 – Operational Simulator Development Cycle: the more complex the simulator is in terms of functionality, the longer the time for its development. Several aspects are considered, such as architecture, database, testing, and software acceptance. In addition to software development, there are the models, wich represent the satellite, implementation.
- SC2 – Development cost: all issues presented in SC1 have an impact on the development cost, as they are time-consuming and demand expertise.
- SC3 – Fidelity of the models: despite the time of implementation of the most sophisticated models, an issue that must be considered is the degree of fidelity of the model. How close to reality the model must represent the real satellite, the space environment, and the ground stations. The models must meet the commitment to fidelity versus complexity for implementation. The model that is sufficiently faithful and that allows the specified analyzes, but not so complicated as to prevent its implementation in an almost real-time tool.
- SC4 – Tests: The criteria to finish testing activities to simulators acceptance are challenging to be defined for the infrastructure and the models. Once the criteria and tests are established, it is possible to know which ones can be reused for a new CubeSat mission to be configured in the simulator, as long as the simulator can be adapted to new missions.
- SC5 – Documents of a CubeSat mission: there is also no clear definition of what documents are needed to report all Cubesat solutions, such as equipment specifications. Similarly, the simulators, what materials are to be produced, such as model specifications, model test specifications, test reports. It is known that time is required to document and test.
- SC6 – Industrial secrecy or Copyrights: related to SC1 and SC3, the design and implementation of a model representing a satellite subsystem require knowledge of the subsystem specificities whose details can be denied by the owner of the subsystem due to industrial secrecy or Copyrights.
- SC7 – Multidisciplinary – a satellite simulator integrates different disciplines, like telecommunication/communication protocols, electrical balance, thermic, mathematic calculus, orbit, attitude control, all of them harmonized in a single

software product, in a distributed or one-processor execution.

Based on the simulators cited in the previous sections, we discuss the listed concerns.

For the first and second shortcomings, SC1 and SC2, respectively, reuse is an acceptable justification, which means that the time and cost of returning an operational simulator cannot be higher than the time and cost of developing a CubeSat, as it usually has a short lifespan. Therefore, developing a simulator that allows it to be applied for various CubeSat missions or reuse an existing simulator or part of it is a way to save time and does not give up a tool that supports operations, especially for more complex CubeSats.

For SC3, let's assume that the fidelity of the model can be met regardless of the tool adopted. The NOS3 [15] and COSMOS [21] simulators were developed directly for small satellites. However, no detailed descriptions were found about the modeling, the model implementation and the interfaces between these models. These details are important to become its use possible. Although SimCBERS, was not developed for use with CubeSats, its modular architecture allows easy adaptation for this purpose. Besides, in [27], the details of subsystem model implementation were presented, which allows their easy assimilation by the team involved in the development, and which can guarantee a high fidelity of the model.

Regarding SC4, testing and acceptance of the infrastructure can be dispensed in the reuse approach because it is expected that this occurred at some previous time and the models will not interfere in the infrastructure execution. Assuming that this is true, only testing of the models that describe the parts of CubeSat will be necessary.

There are no considerations about COSMOS simulator testing, and the authors state: "The ultimate success of the COSMOS approach will be measured by its acceptance into the space community. This will not be seen for a number of years and possibly well after the end of the EPSCoR[1] funding period" [21]. This statement indicates that the user community was testing the tool.

In the SimCBERS, due to the implementation of the models follows a cause-effect based approach, the tests can be directly specified from the specification given in the cause-effect tables. In this sense, the testing criteria are based on the coverage of the table. A solution based on testing models used in the SimCBERS to decrease test effort is presented in [28]. The dynamic models, such as the power supply subsystem, are tested comparing the results of simulation against the satellite specifications.

The SimCBERS is not open-source software, so currently, its use has been limited to INPE projects. Despite this disadvantage, the SimCBERS documentation is quite complete, which can facilitate the conduction of the project, minimizing the aspects presented in SC5. Besides, in the SimCBERS, XML models are implemented according to a standard adopted in the context of the project, taking into account telecommands, switches, telemetry, working states, operating modes, and external parameters.

In practice, the development of a CubeSat is more agile and shorter, so there is no major concern in documenting the project. However, for the development of the models, it is mandatory, at least, the subsystem description, the list of telecommands and telemetries. These minimum set of documents will be possible to create the models and derivate the tests for their validation.

The solutions adopted COSMOS [20,21] and NOS3 [15] regarding the implementation of the models and documentation is not provided in the published

[1] The Established Program to Stimulate Competitive Research

papers.

According to [18], the operational simulator does not include the execution of simulations with hardware-in-the-loop, but the COSMOS and NOS3 simulators have this characteristic. SimCBERS does not allow integration with hardware equipment in its current version. However, the next releases will enable integration with the on-board computer emulator. Both the hardware-in-loop and the emulator perform more accurate simulations in terms of the on-board computer functions but may require additional development costs.

Concerning the adoption of the SimCBERS solutions, modular architecture with ease implementation of the models, it is suggested that the following modifications, to enable simulations during operation CubeSats:

- Due to the small number of telecommands and telemetry, instead of the XML models representing each subsystem of a CubeSat, a single model can implement a complete CubeSat, significatively simplifying.
- Due to the multi-models support, several CubeSats may be configured in the simulator, each one as one model, so, is possible, the simulation of a Cubesat constellation, as illustrated in Figure 4. It is necessary to modify the simulator infrastructure to allow the simulation of CubeSats constellation at the same time.
- During the initialization of the simulator, the user can choose whether to simulate one or more satellites.
- When starting the simulator, the user chooses whether to simulate one or more CubeSat. This choice will depend on the type of analysis the user wants to perform.
- The ground station can be configured with the satellite. This feature is fundamental because, in the current solution, the ground station in the SimCBERS is general. In the case of more satellites and different ground stations for each one, each model must implement its station.
- The orbit propagation must also be optimized to allow the simulation of different sizes of CubeSats and orbit positions. Depending on the type of CubeSat, this model requires more fidelity.

For SC6, Industrial secrecy and Copyrights have to concern, but they will not limit the development of a satellite simulator for a Cubesat. At the same time, the multidisciplinary (SC7) may be easier to be dealt with once Cubesat, in general, is developed by reduced teams.

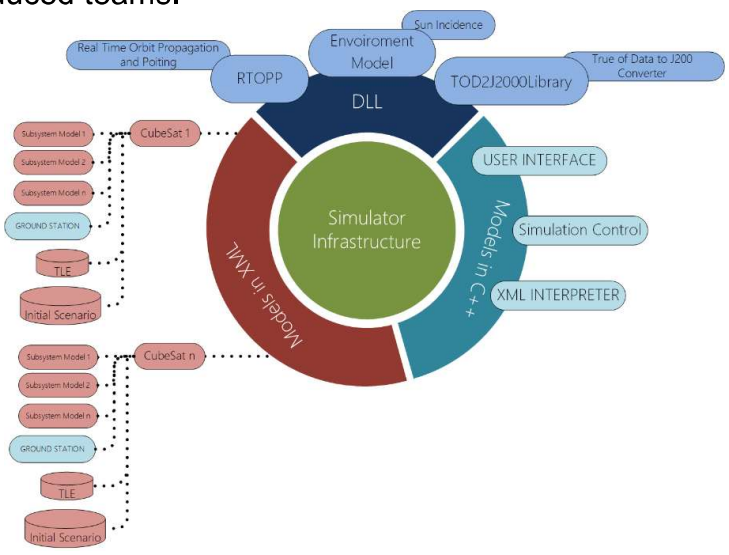

Figure 4: SimCBERS architecture adapted.

6. Conclusions

Despite being a widely used tool for traditional satellite missions [20], operational simulators have been rarely used in nanosatellite missions. This fact can be attributed to the development time of an operational simulator since the CubeSats development approach is faster and more agile.

In practice, projects involving CubeSats are very concerned with environmental testing. In an ideal scenario, one of the most coherent ways of conducting a project in the space area would be to use modeling and simulation throughout the development cycle. The models would evolve throughout the project until reaching maturity to be used in an operational simulator adopting this strategy.

The philosophy involving modeling and simulation is not yet used systematically. However, this scenario may change due to the remarkable sophistication of CubeSats projects, which requires more confidence in the artifacts that will be placed in orbit.

Thus, to perform a safer operation without compromising the mission and the possibility of extending the lifespan of the CubeSats in orbit, tools such as an Operational Simulator are fundamentals.

In this paper, three operational simulators are discussed. The COSMOS and NOS3 simulators were developed, especially for small satellite projects, including CubeSats. SimCBERS was designed for a traditional large satellite.

Due to the cost and development time of an operational simulator, we concluded that its use in a CubeSat mission is possible and justified with reuse, so as not to impact the project budget and schedule. The three solutions presented in this paper, COSMOS, NOS3, and SimCBERS, present a software architecture that can enable its reuse for different missions.

The implementation of the models to be used in COSMOS and NO3 simulations is not detailed, which can be a problem if it is difficult to interpret for the simulator's development and maintenance team. In the SimCBERS context, the implementation of the models is clarified, showing that the solution is simple and that it guarantees a degree of fidelity required by an operational simulator.

For future and more detailed studies, we suggest a comparison of the effort to reuse the three simulators presented, using the same CubeSat as a study case, considering:

- Simulation environment: know the tools and their limitations; configure these tools for use.
- Models: documents needed to specify the models; time to define and implement the models; challenges in implementation.
- Testing: documents needed to test the models; time to test; time to document the results; time to modify the model with a bug.

Acknowledgments

The authors wish to acknowledge the Instituto Nacional de Pesquisas Espaciais for the opportunity to conduct this research in one of its most recognized projects, to the simulator development team, particularly Márcio S. A. Branco and Denise N. R. Azevedo; to CAPES for the financial support.

References

[1] T. Villela, C. A. Costa, A. M. Brandão, F. T. Bueno, R. Leonardi, Towards the Thousandth CubeSat: A Statistical Overview, International Journal of Aerospace Engineering 6 (2019).

[2] R. M. Millan, R. von Steiger, M. Ariel, S. Bartalev, M. Borgeaud, S. Campagnola, J. C. CastilloRogez, R. Fleron, V. Gass, A. Gregorio, D. M. Klumpar, B. Lal, M. Macdonald, J. U. Park, V. S. Rao, K. Schilling, G. Stephens, A. M. Title, J. Wu, Small satellites for space science: A cospar scientific roadmap, Advances in Space Research 64 (2019) 1466 – 1517.
[3] CubeSat Design Specification Rev. 13: https://www.cubesat.org/
[4] CubeSat: A new Generation of Picosatellite for Education and Industry Low-Cost Space Experimentation
[5] I. Nason, J. Puig-Suari, R. Twiggs, Development of a family of picosatellite deployers based on the cubesat standard, in: Proceedings, IEEE Aerospace Conference, volume 1, pp. 1–1.
[6] M. N. Sweeting, Modern small satellites-changing the economics of space, Proceedings of the IEEE 106 (2018) 343–361.
[7] M. Weinzierl, Space, the final economic frontier, Journal of Economic Perspectives 32 (2018) 173–92
[8] A. Poghosyan, A. Golkar, Cubesat evolution: Analyzing cubesat capabilities for conducting science missions, Progress in Aerospace Sciences 88 (2017) 59 – 83.
[9] G. Denis, D. Alary, X. Pasco, N. Pisot, D. Texier, S. Toulza, From new space to big space: How commercial space dream is becoming a reality, Acta Astronautica 166 (2020) 431 – 443.
[10] CubeSat Tables | Nanosat DataBase, https://www.nanosats.eu/tables#constellations.
[11] Private Space Company Planet Lab launches 300 satellites in orbit, https://www.dailymail.co.uk/sciencetech/article-6639285/Private-space-company-Planet-Lab-launches-300-satellites-orbit.html.
[13] H. J. Kim, S. Hariharan, M. Julian, D. Macdonnell, Technology and opportunities of photon sieve cubesat with deployable optical membrane, Aerospace Science and Technology 80 (2018) 212 – 220.
[14] A. G. Guerra, F. Francisco, J. Villate, F. A. Agelet, O. Bertolami, K. Rajan, On small satellites for oceanography: A survey, Acta Astronautica 127 (2016) 404 – 423.
[15] D. Geletko, M. Grubb, J. Lucas, J. Morris, M. Spolaor, M. Suder, S. Yokum, S. Zemerick, Nasa operational simulator for small satellites (nos3): the stf-1 cubesat case study, Journal of Small Satellites 7 (2019).
[16] EUROPEAN COOPERATION FOR SPACE STANDARDIZATION (ECSS), Space engineering: System modelling and simulation (ECSS-E-TM-10-21A), Tech. rep., ESA-ESTEC, http://www.ecss.nl/ (2010).
[17] J. Eickhoff, Simulating Spacecraft Systems, 1st Edition, Springer Aerospace Technology, Heidelberg, Germany, 2009.
[18] A. Lill, T. Zwickl, C. Costescu, L. Patzwahl, C. Soare, M. Langer, Agile mission operations in the cubesat project move-ii, in: 2018 SpaceOps Conference.
[19] J. Michel, N. Dallmann, J. G. Delapp, S. L. Judd, M. Proicou, D. Seitz, R. M. Wheat, A portable autonomous ground station to support a constellation of cubesats, Small Satellite Conference, 2015.
[20] T. C. Sorensen, E. J. Pilger, M. S. Wood, M. A. Nunes, B. D. Yost, A university-developed cosmos to operate multiple space vehicles, in: Space Operations: Experience, Mission Systems, and Advanced Concepts, 2012, pp. 169–192.
[21] T. Sorensen, E. Pilger, M. Wood, M. Nunes, B. Yost, Development of a comprehensive mission operations system designed to operate multiple small satellites, in: 25th Annual AIAA/USU Small Satellite Conference, 2011.
[22] História. http://www.cbers.inpe.br/sobre/historia.php
[23] Lançamentos. http://www.cbers.inpe.br/lancamentos/cbers4.php
[24] A. M. Ambrosio, Especificação Geral de Requisitos do Simulador SIMC3 para os Satélites CBERS 3&4, Technical Report, INPE, 2014
[25] I. P. Rodrigues, L. F. M. Rego, T. d. S. Coimbra, G. P. Gruppelli, A. M. Ambrosio, Identification and control techniques applied to an operational satellite simulator, Congresso Brasileiro de Automatica, 22,. (CBA). 2018.
[26] A. M. Ambrosio, P. E. Cardoso, J. Bianchi Neto, Brazilian satellite simulators: previous solutions trade-off and new perspectives for the cbers program, in: Proceedings..., International Conference on Space Operations, 9th., 2006, p. 7.
[27] J. Tominaga, C. S. Cerqueira, J. Kono, A. M. Ambrosio, Specifying satellite behavior for an operational simulator, in: Proceedings..., Simulation and EGSE facilities for Space Programmes, (SESP 2012)., SESP 2012, 2012.
[28] P. D. B. d. Silva, A. M. Ambrosio, E. Villani, Model-based testing applied to software components of satellite simulators, Modelling and Simulation in Engineering 2018 (2018).

View publication stats