FEDERAL UNIVERSITY OF SANTA CATARINA
TECHNOLOGY CENTER
AUTOMATION AND SYSTEMS DEPARTMENT
UNDERGRADUATE COURSE IN CONTROL AND AUTOMATION ENGINEERING

Lucas Camargo da Silva

Development of a satellite simulator test bench for the AEROSAT mission

Florianópolis
2022

Lucas Camargo da Silva

Development of a satellite simulator test bench for the AEROSAT mission

Final report of the subject DAS5511 (Course Final Project) as a Concluding Dissertation of the Undergraduate Course in Control and Automation Engineering of the Federal University of Santa Catarina.
Supervisor: Prof. Ricardo José Rabelo, Dr.
Co-supervisor: Nadie Rousse, Eng.

Florianópolis
2022

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Camargo da Silva, Lucas
Development of a satellite simulator test bench for the AEROSAT mission / Lucas Camargo da Silva ; orientador, Ricardo José Rabelo, 2022.
79 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Engenharia de Controle e Automação, Florianópolis, 2022.

Inclui referências.

1. Engenharia de Controle e Automação. 2. Nanossatélite. 3. Cubesat. 4. Modelagem. 5. Simulação. I. Rabelo, Ricardo José. II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Controle e Automação. III. Título.

Lucas Camargo da Silva

Development of a satellite simulator test bench for the AEROSAT mission

This work was evaluated in the context of the subject DAS5511 (Course Final Project) and approved in its final form by the Undergraduate Course in Control and Automation Engineering.

Florianópolis, December 14, 2022.

Prof. Hector Bessa Silveira, Dr.
Course Coordinator

Examining Board:

Prof. Ricardo José Rabelo, Dr.
Advisor
UFSC/CTC/DAS

Nadie Rousse, Eng.
Supervisor
Centre national d'études spatiales

Brenda Fernandes Ribeiro, M. Eng.
Evaluator
UFSC/PosAutomação

Prof. Eduardo Camponogara, Dr.
Board President
UFSC/CTC/DAS

To my foundation,
Eliane, Jairo and Maria Clara,
and to my love,
Milena.

ACKNOWLEDGEMENTS

I start this report by expressing my gratitude towards my tutors **Nadie Rousse** and **Stéphane Fredon** for giving me the opportunity to do this internship, and for always helping me with their expertise whenever necessary throughout it.

I also thank CentraleSupélec professor **Clément Elvira**, who accepted to be my academic advisor for this internship.

I thank **Nicolas Verdier** for his advices on the development of the simulator as Nanolab-Academy's project manager, and everyone else who was part of the project I had the pleasure of meeting: **Paul**, **Albane**, **Laetitia**, **Antoine**, **Audrey**, **Romain** et **Louise**. I also thank the previous intern who worked on the AEROSAT test bench, **Manuel Amouroux**, for being available at the beginning of my internship to help me understand the work that had already been done.

Many thanks to everyone who is part of Spacebel's team and helped me advance in my internship, especially **Guilhem Roussel**, who kindly answered all questions I had about BASILES and related subjects, **Corentin Rossignon**, who gave me an introductory lesson about SMP and helped to solve some simulator problems, **Nicolas Monségur**, who presented me in detail the EyeSat test bench, and **Pierre Verhoyen**, who made much of Spacebel's help possible.

Finally, I thank UFSC professor **Ricardo José Rabelo** for having accepted to be my academic advisor for the writing of this work.

“Felizes aquelles que sabem lêr,
mesmo soletrando, os caractéres luminosos que
a noite insculpiu na abóboda celeste!”
(José Brazilício de Souza, 1902)

ABSTRACT

Space test benches are simulators used to test satellites before their launch and to do their maintenance throughout the duration of their missions. The objective of this work was to continue the development of a test bench for the AEROSAT nanosatellite in the context of the Centre national d'études spatiales's project Nanolab-Academy. The test bench is necessary for the validation of AEROSAT and will also play the role of generic model for French university space centers in the development of their own cubesats. For accomplishing this project, the test bench of the EyeSat mission was studied, the versions of architecture for the AEROSAT test bench were defined and the design of the simulator and its models in BASILES and SIMSAT was started. The work resulted in an easily generalizable test bench that includes an orbit extrapolator simulator, called AEROSIM 0, and a simulator containing AEROSAT's Attitude and Orbit Control System equipment models, called AEROSIM 1.

Keywords: Nanosatellite. Cubesat. Modeling. Simulation.

RESUMO

Plataformas de teste espaciais são simuladores utilizados para testar satélites antes de seu lançamento e manter seu bom funcionamento durante a duração de suas missões. O objetivo deste trabalho foi continuar o desenvolvimento da plataforma de testes do nanossatélite AEROSAT no contexto do projeto Nanolab-Academy do Centre national d'études spatiales. A plataforma é necessária para a validação de AEROSAT e também cumprirá o papel de modelo genérico de desenvolvimento para que os centros espaciais universitários franceses possam realizar seus próprios projetos de cubesats. Para a realização deste projeto, a plataforma de testes da missão EyeSat foi estudada, as versões de arquitetura para a plataforma de AEROSAT foram definidas e a concepção do simulador bem como de seus modelos nos softwares BASILES e SIMSAT pôde começar. O trabalho resultou em uma plataforma facilmente generalizável que compreende um simulador extrapolador de órbita, chamado AEROSIM 0, e um simulador contendo modelos de equipamentos do Sistema de Controle de Órbita e Atitude de AEROSAT, chamado AEROSIM 1.

Palavras-chave: Nanossatélite. Cubesat. Modelagem. Simulação.

RÉSUMÉ

Les bancs de test spatiaux sont des simulateurs servant à tester les satellites avant leur lancement et à maintenir leur bon fonctionnement pendant la durée de leurs missions. L'objectif de ce travail a été de continuer le développement du banc de test pour le nanosatellite AEROSAT dans le contexte du projet Nanolab-Academy au Centre national d'études spatiales. Le banc de test est nécessaire pour la validation d'AEROSAT et jouera également le rôle de modèle générique de développement pour que les centres spatiaux universitaires français puissent réaliser leur propres projets de cubesats. Pour la réalisation de ce projet, le banc de test de la mission EyeSat a été étudié, les versions d'architecture pour le banc AEROSAT ont été définies et la conception du simulateur ainsi que de ses modèles sur les logiciels BASILES et SIMSAT a pu commencer. Le travail a abouti à un banc de test facilement généralisable qui comprend un simulateur extrapolateur d'orbite, appelé AEROSIM 0, et un simulateur contenant des modèles d'équipements du système de contrôle d'attitude et d'orbite d'AEROSAT, appelé AEROSIM 1.

Mots-clés : Nanosatellite. Cubesat. Modélisation. Simulation.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AIT	Assembly, Integration and Test
AOCS	Attitude and Orbit Control System
CSU	Centres spatiaux universitaires
ECSS	European Cooperation for Space Standardization
EM	Engineering Model
FS	Flight Software
HDRM	Hold Down and Release Mechanism
HMI	Human-Machine Interface
IRF	Inertial Reference Frame
LEO	Low Earth Orbit
MAS	Mode d'Acquisition en Survie
MNO	Mode Nominal d'Opération
OBC	On-Board Computer
QM	Qualification Model
SCC	Simple Control Center
SCF	Spacecraft Frame
SMP	Simulation Model Portability
TID	Total Ionizing Dose

CONTENTS

1 INTRODUCTION

This document describes the work done during my internship at the Centre national d'études spatiales (CNES) from June 28 to December 24, 2021. The internship followed the end of my second year of engineering school at CentraleSupélec as part of a double degree program between the Federal University of Santa Catarina (UFSC), in Brazil, and CentraleSupélec, in France. The internship took place at the Centre spatial de Toulouse site within the Nanolab-Academy project. I was attached to the DSO/AVI/VS department, responsible for space simulators at CNES.

A satellite simulator test bench for the AEROSAT mission

Space agencies have been designing and developing satellites for decades now, and a solid simulator and test bench infrastructure for that conventional technology has already been established. The use of nanosatellites in large scale, on the other hand, is a recent development. In France, only one commercial nanosatellite has been developed to this day. ANGELS, which was designed by Hemeria, Thalès Alenia Space and Syrlinks in a partnership with CNES, was launched on December 18, 2019 (BOIVINET, 2019). This means that space simulator test benches made specifically for nanosatellites are also a new kind of technology whose development is starting now. A satellite simulator test bench is very important during all phases of a satellite's life, since it is used to develop its Flight Software (FS), to test software updates before they are released to the real satellite and to help the mission to be readapted in case of satellite failures. Therefore, test benches are useful for any kind of satellite missions, including nanosatellites. That is why CNES has decided that AEROSAT, a nanosatellite carrying several environment sensors and technology demonstrators developed in the context of the Nanolab-Academy project, will need a proper satellite simulator test bench for its mission.

A need for a generic nanosatellite test bench platform

Meanwhile, in France, several Centres spatiaux universitaires (CSU), university space centers, have been created in recent years focused on helping students to start their path into the space industry mostly through the development of nanosatellites. A common problem identified by CNES in these CSU is that oftentimes they start their projects from scratch and with little help from space industry experts. This is why CNES started the Nanolab-Academy project, which aims to raise awareness of the development logic, techniques and implementation of space projects among students from French schools and universities. In this context, one of the main objectives of Nanolab-Academy is to create generic platforms on which the CSU can base their

projects. This includes both a platform for the satellite itself and for its space simulator test bench.

Nanolab-Academy has already developed a first nanosatellite test bench, which was built for the EyeSat mission. However, creating a generic test bench platform for nanosatellites was not yet the objective of the project at that time. At the end, most of the simulator was never fully developed, and it was not used much in the mission.

Aiming to create a functional generic nanosatellite test bench platform, Nanolab-Academy decided to start this work within the context of the AEROSAT mission. The development of a test bench for the AEROSAT nanosatellite started at the end of 2020 by the intern Manuel Amouroux, who made a global analysis of the architecture of the test bench and its needs (AMOUROUX, 2021a). The final goal for this test bench is to be released to every CSU so that it will serve as base model for the development of their own test benches.

1.1 OBJECTIVE

The objective of this work was to continue the development of the AEROSAT test bench by designing and implementing the first two versions of its space simulator: an orbit extrapolator and an Attitude and Orbit Control System (AOCS) simulator. In order to reach this goal, the development of AOCS simulation models was also necessary. The objective was not to finish the entire test bench because more than 6 months are necessary for the conclusion of the whole project.

1.2 METHODOLOGY

In order to continue this development, firstly all the software tools and standards that are necessary for the development of simulators, in particular the BASILES development platform, the SIMSAT software and the Simulation Model Portability (SMP) model standard, were studied. Then, an analysis of the state of the art of the EyeSat mission test bench was done, since it was known that some aspects of that test bench could be reused for AEROSAT's. After that, an orbit extrapolator simulator provided by DSO/AVI/VS was adapted with the parameters and characteristics of the AEROSAT mission. This became AEROSIM 0, the first version of simulator for AEROSAT. Finally, the models of the AOCS equipment magnetoboard, solar sensor and reaction wheel were designed and a second simulator was created with these models starting from the orbit extrapolator simulator, called AEROSIM 1. In parallel, the development of CNES's new hardware communication interface brick for test benches was closely followed, since its design influences the architecture of the simulators themselves.

1.3 DOCUMENT STRUCTURE

This section presents the structure of this document.

Chapter 2 - The Centre national d'études spatiales (CNES)

The host organization, CNES, also known as the French space agency, is presented, as well as the main aspects of its structure and its goals.

Chapter 3 - Theoretical foundations

The theoretical foundations of this work are addressed: the definition of nanosatellite, the concept of satellite test benches, the BASILES platform, the Tcl programming language, the SMP model standard, the SIMSAT software, the EyeSat mission, the AEROSAT mission and the hardware communication interface that is needed in hybrid test benches.

Chapter 4 - Developing the AEROSAT simulator test bench

This chapter presents in detail the development and designing of the AEROSAT simulator test bench by focusing on its architecture, the defined hardware communication interface and the necessary simulation models. The work described in this chapter was developed by the author.

Chapter 5 - Results

The final results obtained with each one of the designed models as well as with the two versions of AEROSIM developed during this internship are presented in this chapter.

Conclusion

A conclusion about the outcomes of this project for the organization and for the student is given in this final chapter.

2 THE CENTRE NATIONAL D'ÉTUDES SPATIALES

CNES is the French space agency. It was established under president Charles de Gaulle in December 19, 1961. It is in charge of the development of space activities in France. CNES is also one of the main contributors to the European Space Agency (ESA). CNES's activities are divided into 5 areas (LES. . . , 2021):

- **Launch vehicles**: CNES works for the autonomy of Europe in space by developing launch vehicles, such as Ariane 6, and by operating the European spaceport Centre spatial guyanais (CSG).
- **Sciences**: it supports space exploration by participating in several European and international programs, such as the Perseverance Mars rover's camera SuperCam (SUPERCAM. . . , 2021).
- **Earth observation**: it has participated in the development of several satellites for Earth observation on its own and in cooperation with other space companies and agencies.
- **Telecommunications**: together with other actors of the space industry, CNES works on the development of innovative TV and telecommunications solutions.
- **Defense**: CNES's contribution to the French space defense strategy is fundamental for the security and independence of France. The Syracuse 4A military communications satellite, launched in October 2021, is a good example of this important role (LANCEMENT. . . , 2021).

CNES is composed of four centers, which work in cooperation. They are:

- **Paris Les Halles**: the headquarters.
- **Paris Daumesnil**: the launch vehicles direction.
- **Centre spatial guyanais**: the European spaceport, where rockets are launched.
- **Centre spatial de Toulouse**: the Toulouse space center, a main technical and operational site, where this internship took place. Its north entrance is shown in Figure 1.

2.1 NANOLAB-ACADEMY

Nanolab-Academy, formerly Janus, is a project created in 2012 by CNES with the objective to promote space activities among students of French schools and universities (NANOLAB-ACADEMY. . . , 2021). In order to achieve this goal, it offers technical

Figure 1 – North entrance of the Centre spatial de Toulouse.

Source: Gyrostat, Wikimedia, CC BY-SA 4.0.

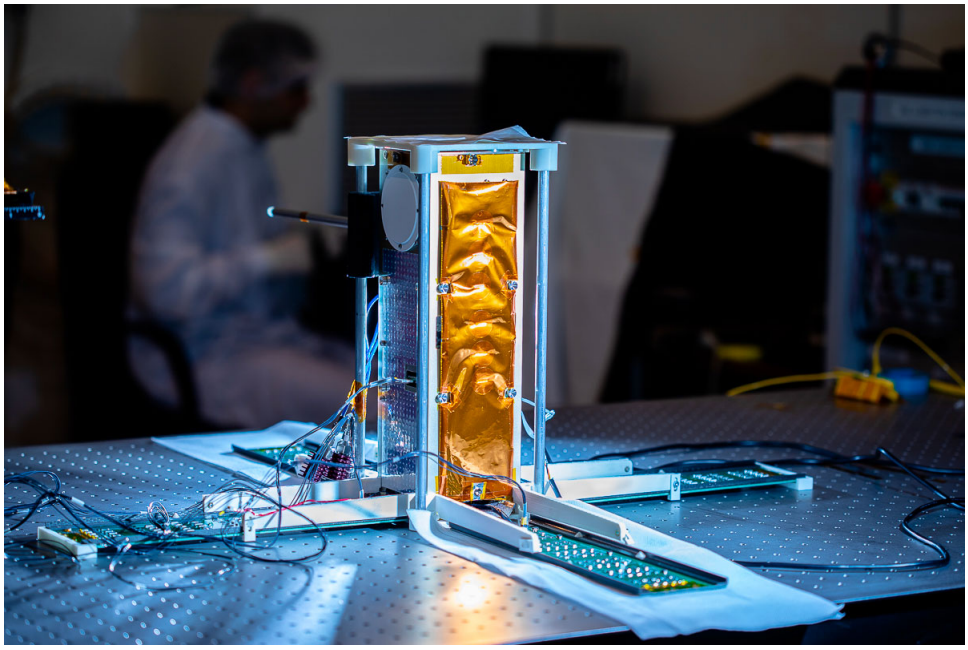
expertise in several areas, such as project management, development plan, launch, reception of telemetry and transmission of remote control messages, data exploitation, among others, to help students develop their own nanosatellites. This also involves experimental activities related to satellite thermal control, mechanics, attitude control, avionics and energy systems. The satellites developed by the project are of great interest for the scientific and industrial communities, as they carry new satellite technologies for testing and equipment for scientific experiments as their payload.

Nanolab-Academy's main achivements

As of 2021, more than 10 French schools and universities develop their nanosatellites with the help of Nanolab-Academy. They are spread throughout the country, and most of them have created their own CSU, where the satellites are developed.

One of Nanolab-Academy's greatest accomplishments is the EyeSat mission. This nanosatellite was developed entirely by students during their internship at CNES. EyeSat is in orbit since December 2019 and allowed the testing of several new space technologies, such as its On-Board Computer (OBC), its antennas and its telemetry and telecommand system (DEB, 2020). A model of EyeSat is shown in Figure 2.

Figure 2 – The EyeSat nanosatellite.



Source: © CNES, Nicolas Tronquar, 2019.

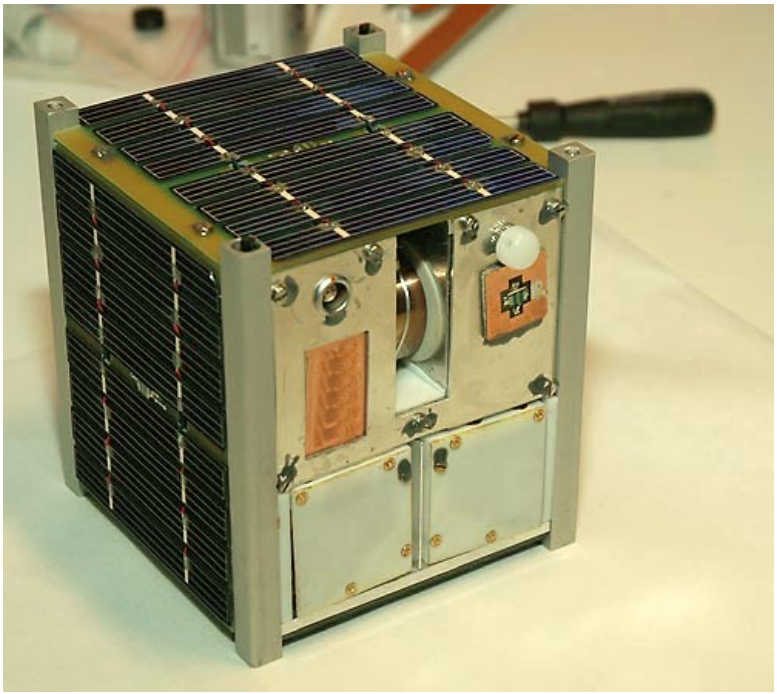
3 THEORETICAL FOUNDATIONS

This chapter describes the theoretical foundations on which this work was based: the definition of nanosatellite, the concept of satellite test benches, the BASILES simulation platform, the Tcl programming language, the SMP model standard, the SIMSAT software, the EyeSat mission, the AEROSAT mission and the hardware communication interface that is needed in hybrid satellite test benches.

3.1 NANOSATELLITES

Nanosatellites are small satellites used in the space industry. They can follow a format called cubesat, whose specifications are defined by the California Polytechnic State University. A cubesat's size is measured in a unit called U. One U is a cube whose edges are 10 cm long and whose mass is less than or equal to 2 kg (JOHNSTONE, 2022). One can stack several U units to form bigger nanosatellites. EyeSat and AEROSAT are both cubesats of type 3 U. Figure 3 shows an example of a 1 U nanosatellite.

Figure 3 – The 1 U Norwegian nanosatellite NCube-2.



Source: Bjørn Pedersen, CC BY 1.0.

One of the main characteristics of nanosatellites is that they tend to have a very low cost compared to satellites of bigger dimensions, especially thanks to their small size and weight. These satelittes are either deployed by the International Space Station

(ISS) or launched as secondary payloads on a launch vehicle. Once in orbit, they are used mostly to test new space technologies and perform scientific research.

3.2 SATELLITE TEST BENCHES

Satellite test benches are systems made to test the behavior of a satellite through a simulated environment. For AEROSAT, it is specifically meant to test its FS. This is done by connecting it to a simulation platform where the orbit, environment, attitude dynamics, equipment and payloads of the satellite are digitally emulated. The communication between the FS and the satellite's equipment and payloads simulation is also emulated through specific hardware reproducing the connections through the corresponding communication protocols.

3.3 THE BASILES SIMULATION PLATFORM

BASILES is a simulation platform developed and maintained by CNES and Spacebel[1]. It allows the designing and implementation of advanced simulators of all kinds. It has been CNES's main framework for the development of simulators for space missions, which includes satellites and spacecraft in general. It is also Nanolab-Academy's main tool for the development of the AEROSAT test bench.

This section summarizes some of the key concepts for understanding a BASILES simulation and its use for the AEROSAT simulators. A detailed and complete manual of the BASILES platform can be found in its installation package and within the software itself. It contains all the essential information on its operation and its several Human-Machine Interface (HMI) windows.

3.3.1 Simulation structure

In BASILES, a simulation is composed of a simulator, models, variants and scenarios.

Simulator

A BASILES simulator is as a set of models that can be used in a simulation. The simulator also defines which services will be used in the simulation, such as the VTS software and the SMP standard.

[1] Spacebel is a Belgium company specialized in the development of space systems, which include space simulators.

Model

Models are the main components of a simulator. A model corresponds to a piece of code that is run during a simulation by instances of the model. Models are analogous to classes in object-oriented programming, and instances correspond to objects. Models can have inputs, outputs and functions, generate simulation events and contain other models. An instance of a model connects to other instances through its inputs and outputs.

Variant

A variant of a simulator is a set of instances of the models contained in the simulator. The connections between instances and the parameters of each instance are also defined by the variant. A simulator may be linked to several variants.

Scenario

A simulation scenario defines global simulation parameters, such as starting time and duration. It is also used to define what data will be generated in specific result files at the end of the simulation. A variant may be linked to several scenarios.

3.3.2 User interface

BASILES's user interface has four main windows of HMI: Conception, Configuration, Exécution and Exploitation. It is important to note that, depending on the use of Tcl files or files generated by the SIMSAT software, some or all features of some windows may become unnecessary.

Conception

The Conception window allows the user to create and compile simulators. It is also possible to create native BASILES models, but only SMP models will be used for AEROSIM, as explained in Section 4.3.1.3.

Configuration

The Configuration window is where simulation variants can be created and configured.

Exécution

The Exécution window is used to create, configure and run simulation scenarios.

Exploitation

The Exploitation window can be used to analyze simulation results.

3.3.3 BASILES and the AEROSAT simulators

Two simulators were used as a starting point for developing the AEROSAT simulators: a very basic one, called "bouncing ball", and a more complex one, called "orbit extrapolator".

The bouncing ball simulator

"Bouncing ball" is a simulator designed by the EyeSat intern Quentin Rohan, and its documentation is also available to AEROSAT interns (ROHAN, 2016). The objective is to simulate a very simple physics problem: a ball that starts its path with a specific position and an initial velocity and bounces over time. The simulator is described in more detail in Appendix A.

This simulator was also used to put into practice the conversion of a model from the BASILES standard to the SMP standard, which is needed for some of the AEROSAT models since a few of them will be the same used for the EyeSat test bench. Since the EyeSat models follow the BASILES standard, the conversion to SMP is necessary in these cases.

The orbit extrapolator simulator

An orbit extrapolator is a simulator capable of calculating the state of a satellite[2] over time from an initial state vector. It contains several models divided into the following three categories:

- **Orbit models**: calculate and propagate the satellite state.
- **Environment models**: simulate the environment that surrounds the satellite, such as the solar radiation, the infrared radiation that comes from Earth, the atmospheric drag[3], Earth's magnetic field, among others; this is necessary because the environment influences significantly the satellite state.
- **Dynamics models**: compute the result of the interaction between the environment models and the satellite's current state, attitude and internal forces.

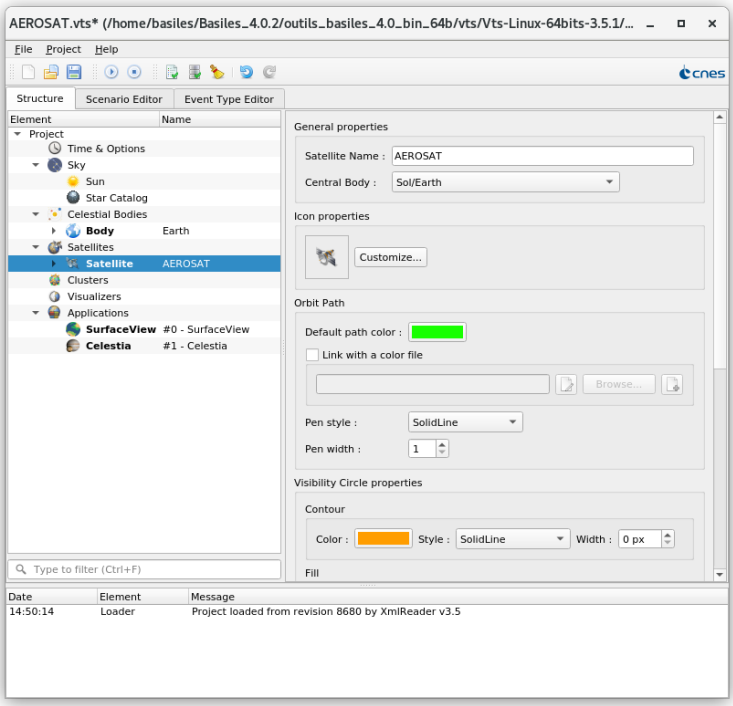
This set of models is essential for any kind of simulator whose objective is to test the FS behavior when the satellite is in orbit. Therefore, it serves as a basis for all simulator versions necessary for the AEROSAT mission.

[2] The state of a satellite is represented by its position and velocity.
[3] The atmosphere at Low Earth Orbit (LEO) is still significant enough to be considered.

3.3.4 VTS

VTS is a CNES software that can be used within the BASILES platform to interact with data generated by a given simulation. It is mostly used to generate 2D and 3D animations of the satellite being simulated and its environment. VTS uses another software, Celestia, to generate a 3D space simulation including planets, stars, the Sun and the Moon. Figure 4 shows VTS's main window, where the animation parameters, such as the satellite's 3D model, are configured, and a VTS project is created to be used in a BASILES simulation.

Figure 4 – VTS's main configuration window, VTS 3.5.1.



Source: Author.

3.4 THE TCL PROGRAMMING LANGUAGE

Tcl is an acronym for "tool command language". It is an interpreted programming language designed to be simple and easy to learn while serving diverse types of applications (LANGUAGE. . . , 2021). Tcl can be used in BASILES to write simulation scenarios. This includes scripts not only for the scenario itself, but also for the creation of model instances, the definition of the connections between them, their schedule, among other functionalities. One has significantly more control over a simulation when using Tcl to design scenarios instead of the native BASILES interface and its automatically

generated .sce files. For this reason it has been decided that all AEROSIM simulation scripts will be written in the Tcl language with .tcl files.

3.5 THE SMP MODEL STANDARD

SMP is a standard developped by the European Space Agency (ESA) that defines design rules and principles for developing models to be used in space simulators. Its main objective is to reduce development effort by enabling reuse and portability of simulation models. SMP2 is the last major release of the SMP standard, and all models that will be made for the AEROSIM simulators will be designed following it.

SMP2 was promoted in 2020 to an European Cooperation for Space Standardization (ECSS) standard by the name of ECSS-E-ST-40-07C, and its first issue fully explains the standard and its definitions (ECSS-E-ST-40-07C. . . , 2020). The SMP service user manual written by Spacebel's Simulator Team is another good source of information to understand the standard (USER. . . , 2020). This user manual also explains how to make a model implement the dataflow extension, which makes all the inputs of an instance update automatically as soon as the corresponding outputs are updated. This is necessary because most of the model connections of the simulators for AEROSAT will be made this way, as explained in more detail in Section 4.3.1.3.

3.6 THE SIMSAT SOFTWARE

SIMSAT is a software developed by ESA for the designing of simulation models following the SMP2 standard. SIMSAT can be used to generate four types of files related to simulation models:

- **Catalog** (.cat): used to design models. The entire model is described in this file: its fields, functions and SMP features. Several models can be grouped together under the same catalog. This is the most important file as it is used to generate the skeleton of the source code of the models. This code is written in C++.
- **Package** (.pkg): used to describe the SMP implementation of models, arrays, SMP features, among other catalog-related characteristics. It is necessary for generating other pieces of code that are required for the SMP standard.
- **Assembly** (.asb): used to create and connect instances and initialize the values of their fields.
- **Schedule** (.scd): used to create and schedule simulation tasks and events.

In practice, if the creation and scheduling of instances are made through Tcl scripts, .asb and .scd files are not needed. However, they can be useful for designing simple simulators, such as *recette* tests.

3.6.1 SIMSAT and the SMP2 standard

SIMSAT allows one to generate the skeleton and all the pieces of code that are needed for a model to be SMP2 compliant. An excellent source of information about SMP2 and its use within SIMSAT is a manual written by the Spacebel's Simulator Team (HELP. . . , 2014). It is a guide through building a simple SMP2 simulator from the creation of its models in SIMSAT to its implementation in BASILES. The simulator in question is a simple counter. It contains two models: a "Master" and a "Counter". The Master asks the Counter to increment its counter variable according to a time step. Even though this is a simple simulator, it deals with several key SMP2 concepts, such as entry points, interface links, interfaces, references, among others. As such, it is a good starting point for creating more complex SMP2 models with SIMSAT.

3.6.2 Converting a model from the BASILES standard to SMP2

A few of the models that will be used for the AEROSAT test bench will come from EyeSat models. However, they all follow the BASILES model standard, but the models made for AEROSAT must be SMP2 compliant. This means that it is necessary to convert models from the BASILES standard to SMP2 for developing models for AEROSAT. A useful source of information about this process is a guide explaining the basics of how to make this model conversion using SIMSAT written by previous AEROSAT test bench engineering intern Manuel Amouroux (AMOUROUX, 2021b). It is important to note that this process is very model-specific, so each model needs specific adaptations according to its characteristics and features.

3.7 THE EYESAT MISSION

EyeSat is a 3 U nanosatellite developed by students at Nanolab-Academy from 2013 to 2019. It was launched from the Centre Spatial Guyanais on December 18, 2019, on a Soyuz rocket. The objective of its mission was the study of the zodiacal light by means of a telescope with an image detector it carries as payload. As of this date, EyeSat is still operating, but its main mission has been compromised by failures in its system of reaction wheels. Despite these difficulties, EyeSat has still been able to capture images of deep-sky objects, such as the Lagoon Nebula, as shown in Figure 5.

The importance of EyeSat for the AEROSAT mission lies on the fact that AEROSAT's platform and some of its equipment are the same used for EyeSat. The EyeSat test bench has been studied for the development of the AEROSAT test bench as well, and some simulation models will be readapted for AEROSAT.

Figure 5 – The Lagoon Nebula by EyeSat in 2020.

Source: © CNES.

3.7.1 The EyeSat test bench

The EyeSat test bench was developed by a total of 10 interns between 2015 and 2019. This section describes its main features for it serves as a starting point in the development of the AEROSAT test bench.

One notices that the this test bench was not continuously developed, with interns focusing on very different aspects of it often without continuity between their work. Several versions of the simulator were created, but not all of them are complete or reusable as is. The last version of the test bench, developed in 2019, is the only one known to be functional today. All information described here was gathered and summarized throughout the duration of my internship.

3.7.1.1 Two test bench architectures

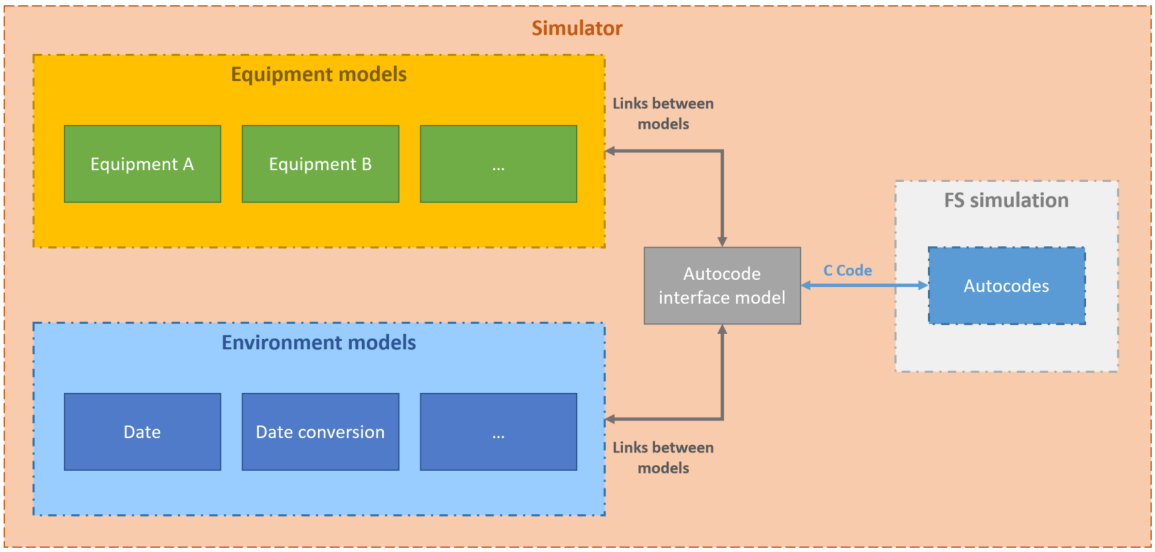
Two types of architecture were used for the development of the EyeSat test bench: one where the FS behavior is modeled by using autocodes[4], called "fully digital test bench", and another one where the FS is uploaded to an electronic board, called "hybrid test bench". These architectures are presented in the following paragraphs.

[4] An autocode is a piece of code that has been automatically generated by a software program, in this case to simulate a portion of the FS.

The fully digital test bench

In the fully digital version of the test bench, all testing and validation is done in the simulator. The autocodes are added to the list of external codes needed for the simulation, and the models interact with these autocodes as if they were the real FS. Figure 6 illustrates this architecture.

Figure 6 – The fully digital EyeSat test bench architecture.



Source: Author, 2021.

The hybrid test bench

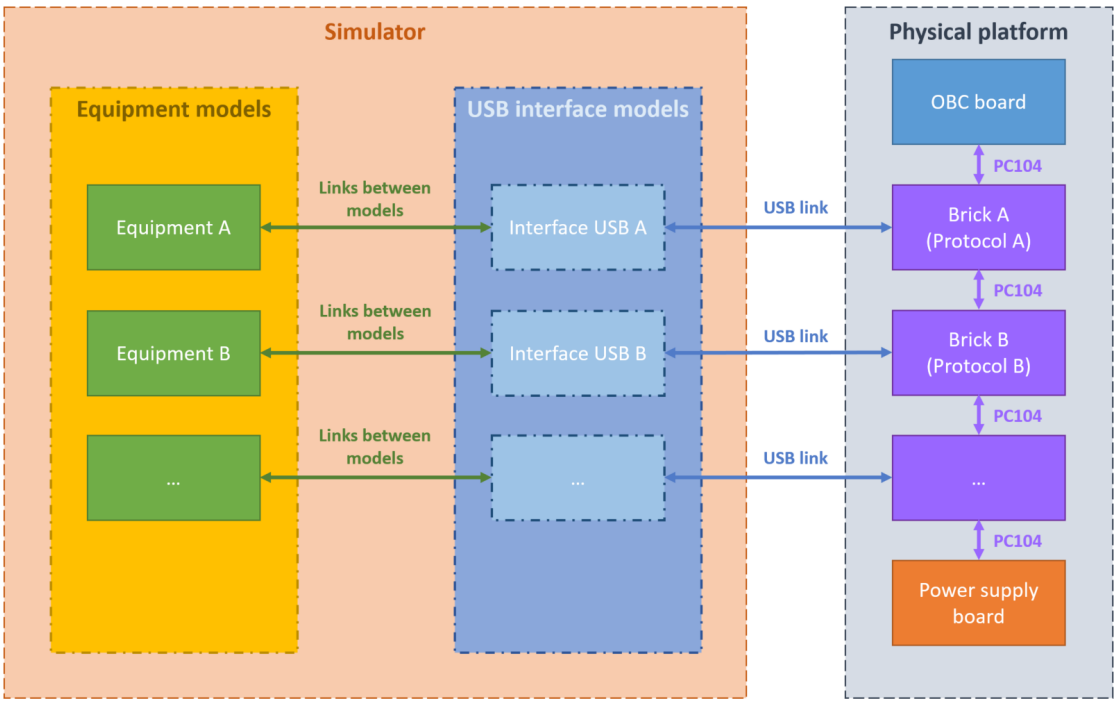
The hybrid version of the test bench includes electronic boards that are connected to the computer where the simulator is running through USB cables. This type of simulation architecture can be referred to as "hardware-in-the-loop". The boards are connected to each other using a PC104 bus which reproduces the connections of the real satellite. Three types of board exist:

- **Power supply board**: distributes the power to the other boards; it is connected to an external power supply.
- **Interface brick**: makes the communication link between the equipment models, which run in the simulator, and the OBC; there is one interface brick per modeled equipment; each brick implements the communication protocol of the equipment to which it corresponds.

- **OBC board**: contains the processor and memory that make up the OBC; this is where the FS runs; the FS is uploaded to it via a special USB cable with a JTAG connector that is plugged to the board.

Figure 7 presents this architecture.

Figure 7 – The hybrid EyeSat test bench architecture.



Source: Author, 2021.

3.7.1.2 The final version of the EyeSat test bench

The last version of the test bench was designed to validate elements of the FS, in particular the AOCS. It can work in a fully digital or hybrid configuration. As such, it is composed of several models, variants and scenarios and a few electronic boards, which are described in the following paragraphs.

EyeSat simulator models, variants and scenarios

All models used in the final version of the EyeSat test bench are presented in Table 1.

Table 1 – List of models used in the last version of the EyeSat test bench.

Begin of Table 1.	
Model	**Description**
Interface_Autocode_MAS	Makes the interface between the simulator models and the Mode d'Acquisition en Survie (MAS) autocode (fully digital configuration only)
Interface_Autocode_MNO	Makes the interface between the simulator models and the Mode Nominal d'Opération (MNO) autocode (fully digital configuration only)
ABDO	Calculates the elementary albedo fluxes
ACC	Calculates the total linear acceleration applied to the satellite
CONVERSION_JJ_SEC	Converts a CNES Julian day into seconds
COUP	Sums the external torques and transports them from a point O to a point G
DATE_BASILES	Provides the simulator date
DATE_CALCUL_SIMU	Calculates the elapsed time in seconds since the start date of the simulation based on January 1, 1950 at 0h taking into account the simulation frequency
DATM_DTM78_MSI86	Calculates the environmental atmospheric density based on the Julian date, the geodetic coordinates and solar activity parameters
DriverSwitcher	Loads the driver that allows the USB communication for the hybrid bench to work
DYN	Models the attitude dynamics of the satellite
ECLA	Calculates the illumination of the satellite by the Sun and the Moon as well as the phase of the Moon as seen from the satellite position
FrameDecoder	Analyzes the input frames and outputs the communication protocol data
FrameEncoder	Encodes the bytes received as input in a format that the interface brick understands
FrameIdentifier	Detects if the data read at the input of the interface brick contains frames that are readable by the computer and segments them correctly

Continuation of Table 1.	
Model	**Description**
FSOL	Provides the value of the solar flux illuminating the satellite taking into account a possible eclipse configuration
IMTQ_Board	Models the iMTQ board
IMTQ_MAC	Models the actuators of the iMTQ board
IMTQ_MTM	Models the magnetometer of the iMTQ board
Interface_SLV_S4_IMTQ	Converts iMTQ and environmental model data into inputs for the iMTQ interface brick
Interface_SLV_S9_RWS	Converts data from the reaction wheel model and the environment models into inputs for the reaction wheels interface brick
InterfaceReg_RWS	Models the data registers of the reaction wheels interface brick
InterfaceRegistre	Models the data registers of interface bricks in general
MAGT_IGRF	Calculates Earth's magnetic field and expresses it in the inertial reference frame according to geocentric coordinates
MCI_INER_C	Models the inertia matrix of EyeSat
ORBT	Integrates the position and velocity of the satellite in the inertial reference frame
PERT_AEROSOL	Models solar and aerodynamic disturbances
POSA	Provides the direction vector and the Earth-Moon and Earth-Sun distances in the Veis Gamma50 frame if the input quaternion is the identity quaternion
POT_LUNSOL	Calculates the gravitational potential due to the attractions of the Moon and the Sun
POT_TERRE	Calculates Earth's gravitational potential at the satellite position in the inertial reference frame
RWS_Hyperion_BASILES	Models a set of 4 reaction wheels
TGDG	Models the gravity gradient torsor
TMAG	Calculates the magnetic disturbance torque

USBCommunication	Allows the communication between the simulator and the interface bricks
End of Table 1.	

Source: Author (CAMARGO DA SILVA, 2021c).

The simulator variants corresponding to the final version of the EyeSat bench are described in Table 2.

Table 2 – List of variants of the final version of the EyeSat test bench.

Variant	**Configuration**	**Description**
Var_MAS	Hybrid	EyeSat's MAS
Var_MAS_NUM	fully digital	EyeSat's MAS
Var_MNO	Hybrid	EyeSat's MNO
Var_MNO_NUM	fully digital	EyeSat's MNO
Var_MNO_Veille_NUM	fully digital	EyeSat's MNO sub-mode Veille
Var_OpenLoopCommunication	Hybrid	Test of the communication with the interface bricks in open loop
Var_OpenLoopCommunication SLV_S4_IMTQ	Hybrid	Test of the communication with the interface brick S4-IMTQ1
Var_OpenLoopCommunication SLV_S9_RWS	Hybrid	Test of the communication with the interface brick S9-RWHI2
Var_OpenLoopEquipmentMTQ	Hybrid	Test of the iMTQ board models with the interface brick S4-IMTQ1 in open loop
Var_OpenLoopEquipmentRWS	Hybrid	Test of the reaction wheels model with the interface brick S9-RWHI2 in open loop

Source: Author (CAMARGO DA SILVA, 2021c).

Finally, the corresponding scenarios are presented in Table 3.

EyeSat test bench boards

The electronic boards that make up the last version of the EyeSat test bench in its hybrid configuration are presented in Table 4.

Hybrid test bench assembly

The physical part of the hybrid configuration of the test bench is shown completely assembled and connected to the simulation computer in Figure 8. Its parts are

Table 3 – List of scenarios of the final version of the EyeSat test bench.

Scenario	**Variant**
MAS.sce	Var_MAS
MAS_NUM.sce	Var_MAS_NUM
MNO.sce	Var_MNO
MNO.tcl	Var_MNO
MNO_Veille_NUM.sce	Var_MNO_Veille_NUM
OpenLoopCommunication_SLV_S4_IMTQ.sce	Var_OpenLoopCommunicationSLV_S4_IMTQ
OpenLoopCommunication_SLV_S9_RWS.sce	Var_OpenLoopCommunicationSLV_S9_RWS
OpenLoopEquipmentIMTQ.sce	Var_OpenLoopEquipmentIMTQ

Source: Author (CAMARGO DA SILVA, 2021c).

Table 4 – List of electronic boards of the final version of the EyeSat test bench

Reference	**Type**	**Description**
SN122250	OBC simulation board	Used to emulate the OBC; to be connected onto an Enclustra board
ODB1	Enclustra board	Makes the interface between the OBC and the other boards
S4-IMTQ1	Interface brick	Simulates the communication with the iMTQ board
S9-RWHI2	Interface brick	Simulates the communication with the reaction wheels
CP-1	Power supply board	Provides power to all boards

Source: Author, 2021.

also indicated in the image. The author has written a complete and detailed guide explaining how to assemble the EyeSat test bench and how to run simulations with it (CAMARGO DA SILVA, 2021c).

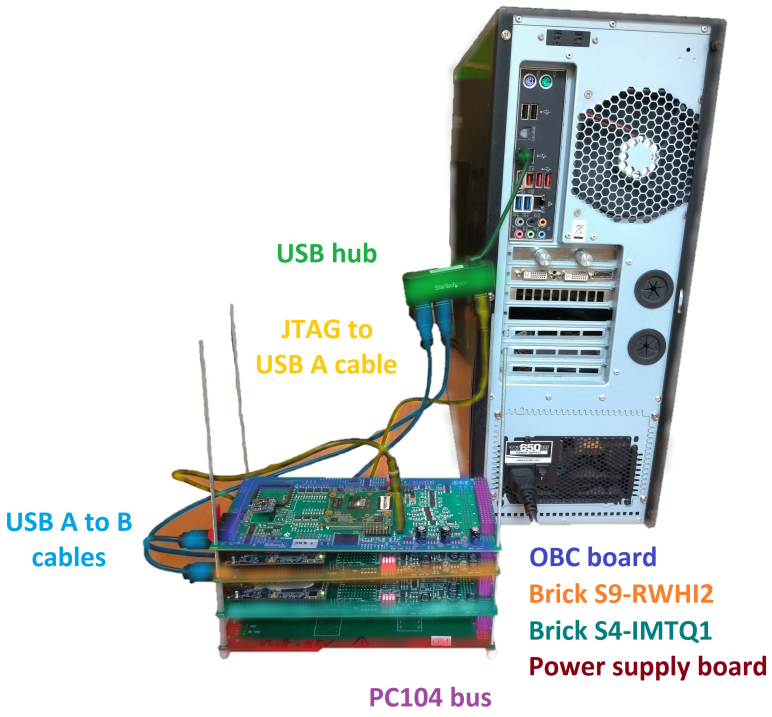
3.7.1.3 The EyeSat test bench inventory

Other models and simulators were created throughout the development of the EyeSat test bench, but it is not known whether they were validated or if there is any documentation about them. There are also power supply boards, OBC boards, among others. The entire inventory is described in detail by the author in a previous work (CAMARGO DA SILVA, 2021c).

3.8 THE AEROSAT MISSION

AEROSAT is a 3 U nanosatellite currently under development at Nanolab-Academy. Figure 9 shows a 3D model of AEROSAT.

Figure 8 – The physical part of the hybrid configuration of the EyeSat test bench.



Source: Author, 2021.

As opposed to EyeSat, which had a single and specific scientific mission, AEROSAT will carry 7 payloads, each one serving a different objective. These payloads are divided in two categories: environment sensors and technology demonstrations.

3.8.1 Environment sensors

DREAM

DREAM is a detector of protons and electrons. It will allow the analysis of electronic particles trapped by Earth's magnetosphere at LEO.

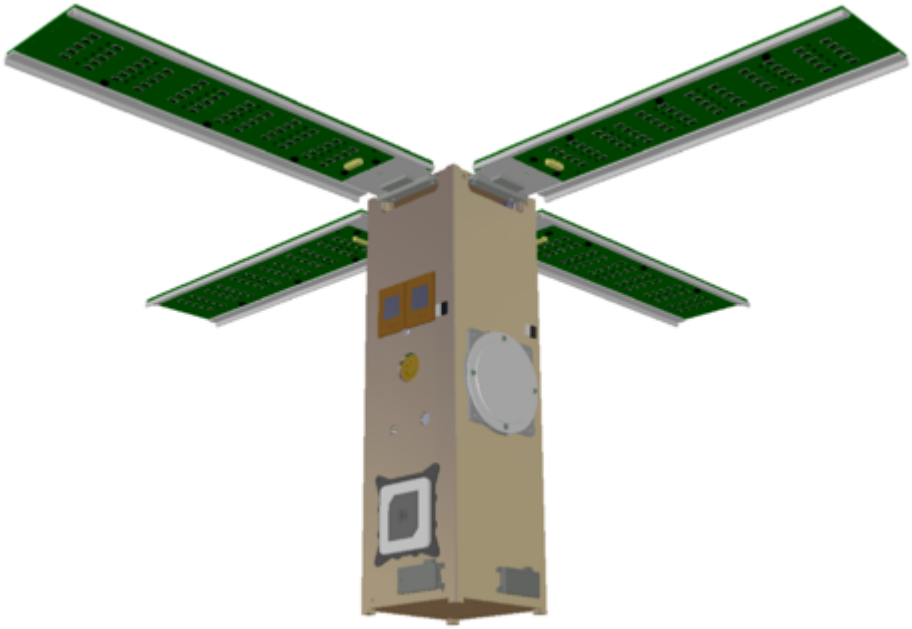
GLOWRIA

GLOWRIA is an optic fiber used as a radiation detector. It will be able to measure the Total Ionizing Dose (TID) at the orbit of AEROSAT.

ERS

ERS consists of a set of radiation detectors by laser. It will be used to observe the radiation coming from Earth at the orbit of AEROSAT. The resulting data will be

Figure 9 – 3D model of AEROSAT, FreeCAD 0.19.



Source: Author.

used to analyze Earth's energy balance, which is important for the study of climate change.

COAST

COAST is a payload divided in two parts: Resistack and Microbalance. **Resistack** is a detector of monoatomic oxygen, which is a gas that is significantly present in LEO and is corrosive for spacecraft material. **Microbalance** is a contaminant detector that will be used in the study of outgassing[5].

3.8.2 Technology demonstrations

LOAC is a powerful aerosol counter and measurement tool. It will be used in future missions to explore and study the atmosphere of other planets. AEROSAT will enable the validation of its operation in the space environment.

SESAM is a digital payload. It is a piece of software that will use artificial intelligence algorithms to monitor and detect anomalies in the satellite's telemetry data.

[5] Outgassing is the release of gas that was trapped in some material.

3.8.3 Platform

AEROSAT's platform is based on EyeSat's, since the operation of several components has already been validated by the EyeSat mission.

Attitude and Orbit Control System (AOCS)

AEROSAT's AOCS is composed of two sensors and two actuators. A **Sun sensor** will measure the position of the Sun, and a **magnetometer** will be used to measure Earth's magnetic field. **Magnetorquers** will use Earth's magnetic field to adjust the satellite's attitude, and a **reaction wheel** will keep the satellite pointed to a given direction despite disturbance torques.

Communication

The communication system of AEROSAT is made of two components. A **S-Band transceiver** will be responsible for managing the communication itself, and two **S-Band antennas** will perform the physical reception and transmission of data.

Global Navigation Satellite System (GNSS)

AEROSAT will carry a GNSS so that orbit restitution will be autonomously made by the satellite itself.

On-Board Computer (OBC)

The OBC is responsible for controlling several aspects of the satellite operation, such as the AOCS, payloads, thermal management and data storage.

Mezzanine & Interface Electronics (MIEL)

MIEL is an electronic board that will perform the communication between the OBC and the payloads ERS and COAST. It will also be used to switch all payloads on and off when necessary.

Power

AEROSAT's power system include four **solar arrays** to collect solar energy, **batteries** to store this energy and a **power board** to manage this entire system.

Hold Down and Release Mechanism (HDRM)

AEROSAT will carry four pieces of Hold Down and Release Mechanism (HDRM), which are responsible for holding the solar arrays attached to the satellite during its launch phase and releasing them once in space.

3.8.4 Orbit

In order to maximize the availability of solar energy, AEROSAT will follow a Sun-synchronous orbit (SSO), so that the Sun will almost always illuminate its solar arrays. The orbit characteristics are as follows:

- Semi-major axis: 600 km.
- Eccentricity: less than 0,005 (circular orbit).
- Inclination: 97,78°.
- Local time of the ascending node: 6h.
- Period: 96,68 minutes, which corresponds to almost 15 orbits per day.

3.8.5 Mission schedule

In december 2021, the AEROSAT mission was finishing its feasibility study and entering its preliminary definition phase. An Engineering Model (EM) is planned for 2022, and a Qualification Model (QM) is expected in 2023. The satellite is scheduled to be launched from the end of 2023 on.

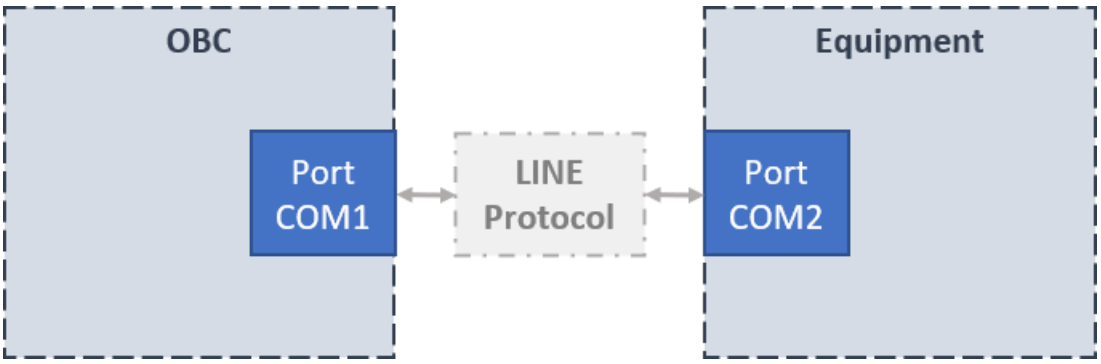
3.9 THE HARDWARE COMMUNICATION INTERFACE

Hybrid satellite test benches need specific hardware for emulating the communication interface between the satellite's OBC and its equipment and payloads. This section describes in detail the requirements this hardware must fulfill in order to reproduce reliably the real satellite operation. The old CNES hardware responsible for this operation up to the beginning of this work is also described.

3.9.1 The communication interface in a real satellite

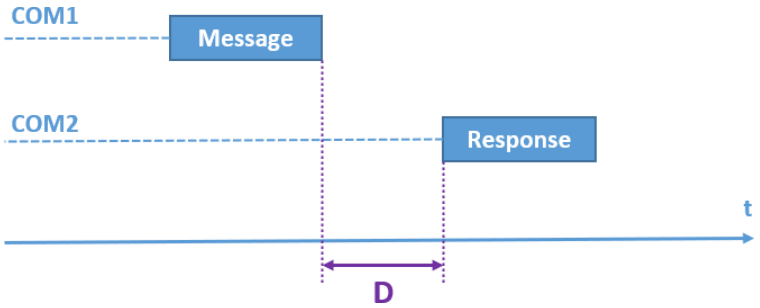
Figure 10 presents the real communication interface architecture, which satellite test benches reproduce.
The OBC and each equipment communicate through a physical link, here referred to as LINE Protocol. The communication is subject to a delay D between the emission of a message and its corresponding response, which is illustrated by Figure 11.
This delay is not equally significant for all communication protocols. Table 5 presents qualitatively the significance of D for some of the most common protocols used in satellites from the highest to the lowest significance.

Figure 10 – The hardware communication interface architecture in a real satellite.



Source: Author, 2021.

Figure 11 – The hardware interface communication delay.



Source: Author, 2021.

3.9.2 The old CNES communication brick

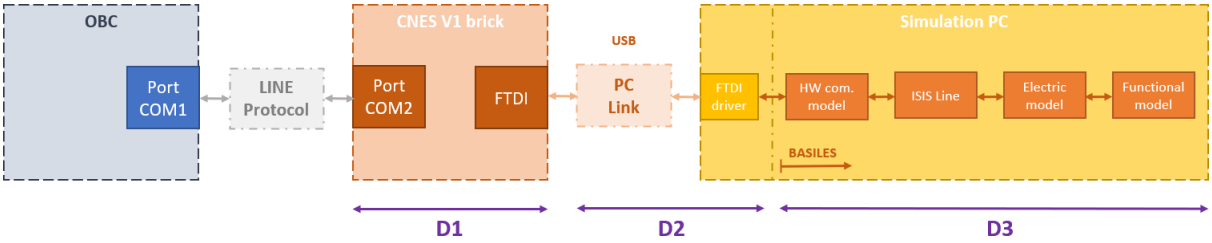
Up to the beginning of this work, missions in which CNES is involved use an old CNES hardware communication brick. The test bench architecture corresponding to this brick is presented in Figure 12.

Table 5 – Significance of D for common communication protocols

Protocol	Significance of D
1553	High
I²C	Medium
1-Wire	Medium
CAN	Low
UART	Very low
SpaceWire	Very low
TMTC	Very low

Source: Author, 2021.

Figure 12 – Test bench architecture with the old CNES brick.



Source: Author, 2021.

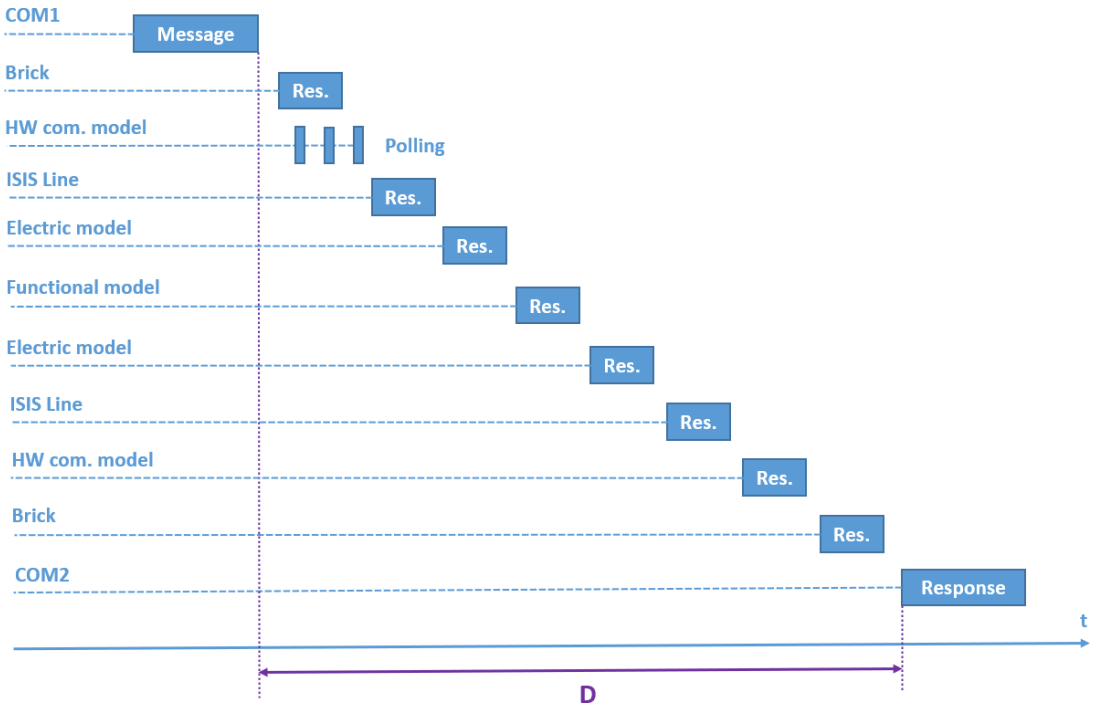
This old design presents three main problems :

- It is obsolete: CNES cannot produce new bricks following this design anymore.
- It implies three different delays, which can become significant when added up:
  - D1: the delay of the message inside the brick, which depends on the communication protocol.
  - D2: the delay of the PC link and the driver in the simulation computer.
  - D3: the delay of the polling process[6] to receive new messages plus the delay of the message to transit between simulation model instances (hardware communication, ISIS Line, electric and functional models). This delay is often very short, but this depends on each model. The total delay generated by this design is graphically represented in Figure 13.
- It is not easily manageable: protocol errors are not detectable, it is not possible to confirm whether all connections have been established, the brick cannot be reset, among other management problems.

Given all the problems related to the old brick design, it has become clear that CNES needs a new brick for future test benches. CNES's department DSO/TB/ET started the development of a new brick, called *briquette*, which is intended to replace the old design. At the beginning of this work, the briquette's design was not yet completely defined. Some meetings involving Nanolab-Academy, Spacebel and CNES's departments DSO/AVI/VS and DSO/TB/ET have taken place with the objective to develop a design that meets all the simulation needs and solves the old brick's problems. A few design approaches were discussed, and the chosen one is described in Section 4.2. The designs that were considered but not chosen are described by the author in a previous work (CAMARGO DA SILVA, 2021b).

[6] Polling is a method of receiving messages in which the receiver actively checks if there is a new message periodically.

Figure 13 – The communication delay generated by the old CNES brick.



Source: Author, 2021.

4 DEVELOPING THE AEROSAT SIMULATOR TEST BENCH

This chapter describes the development of the AEROSAT test bench by presenting the designed test bench architecture, the new CNES communication brick and the models that are implemented by the digital simulators AEROSIM 0 and 1. Furthermore, the design of the AOCS models is presented in detail. The work described here was developed by the author.

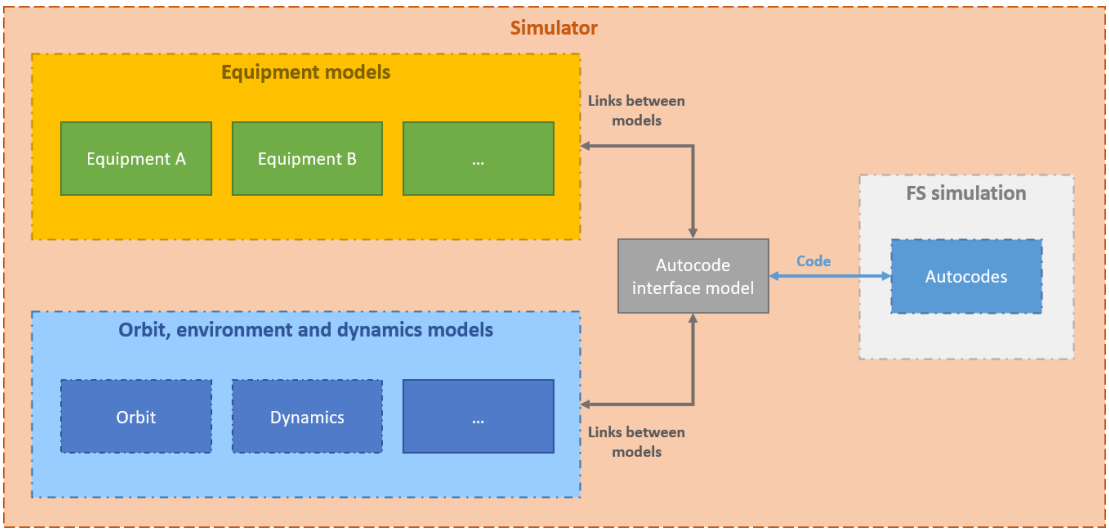
4.1 THE AEROSAT TEST BENCH ARCHITECTURE

The AEROSAT test bench will include hybrid and fully digital versions. This section describes the architecture designed for both of them.

4.1.1 Fully digital test bench

The objective of this version of the test bench is to test some specific FS algorithms, such as the AOCS ones. In the fully digital version of the test bench, all the testing is done in the simulator and the FS behavior is emulated by autocodes, exactly as in the fully digital version of the EyeSat test bench. Figure 14 illustrates this architecture.

Figure 14 – The fully digital AEROSAT test bench architecture.



Source: Author, 2021.

4.1.2 Hybrid test bench

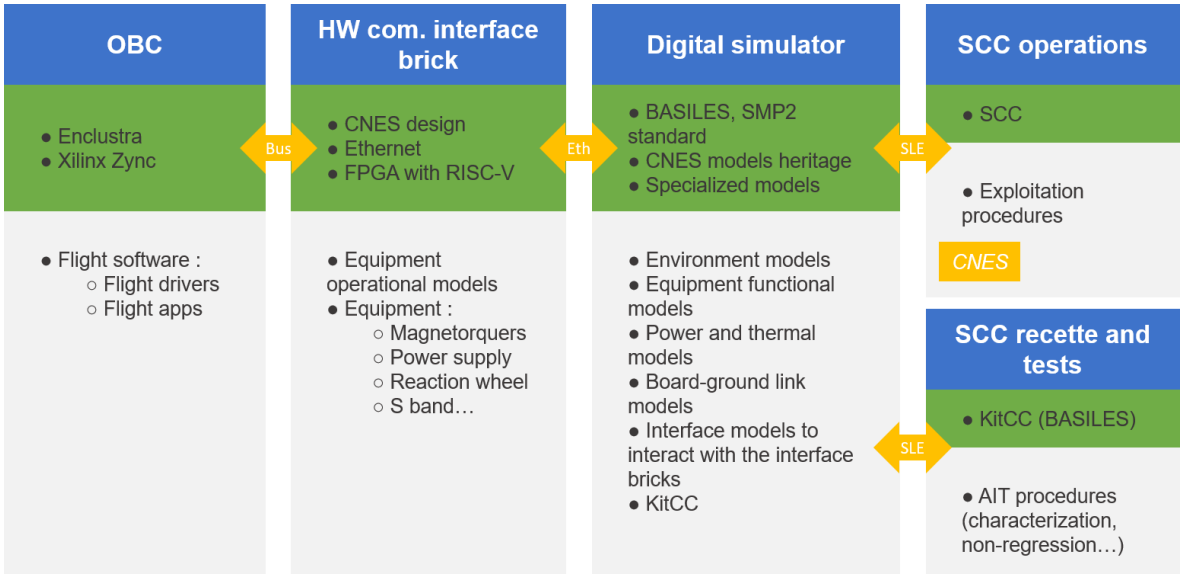
The complete hybrid test bench is composed of 4 main parts:

- **OBC**: the satellite's computer, where the FS runs.

- **Hardware communication interface brick**: the electronic hardware that simulates the communication between the satellite equipment, which are modeled in the digital simulator, and the OBC.
- **Digital simulator**: where the functional models of the equipment and the models of the satellite environment, orbit and dynamics run.
- **Simple Control Center (SCC)**: the command and control center, which is the system that enables communication between the ground teams and the satellite. It can be simulated (*recette* and tests) or the real center (operations).

Figure 15 presents a global view of the hybrid architecture.

Figure 15 – The hybrid AEROSAT test bench architecture.



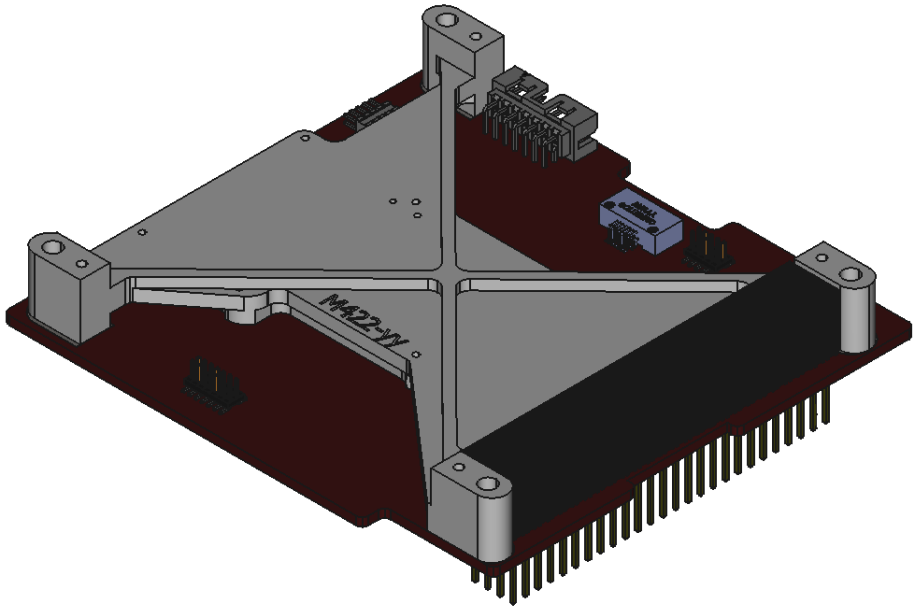
Source: Author, 2021.

The communication between the interface bricks and the OBC is done through a PC104 bus, which reproduces the connections of the satellite flight model. The bricks are connected to the computer where the simulator runs through an Ethernet connection. The simulator and the SCC communicate using the SLE protocol.

4.1.2.1 The On-Board Computer

The OBC is the brain of the nanosatellite, managing all of its platform aspects (such as its AOCS, thermal control, communication, among others) as well as its payloads. AEROSAT uses the Ninano board, developed by Steel Electronique. It includes an ARM9 processor, 512 MB of RAM and 15 GB of mass storage. It is the same board used by EyeSat. Figure 16 shows a 3D model of the Ninano board.

Figure 16 – 3D model of the Ninano board by Steel Electronique, FreeCAD 0.19.



Source: Author, 2021.

For testing, one can either use the real OBC or emulate it through an FPGA. For the latter, the AEROSAT test bench will use an Enclustra board to which the OBC simulator board containing the FPGA is connected, exactly as in the EyeSat test bench.

4.1.2.2 The hardware communication interface

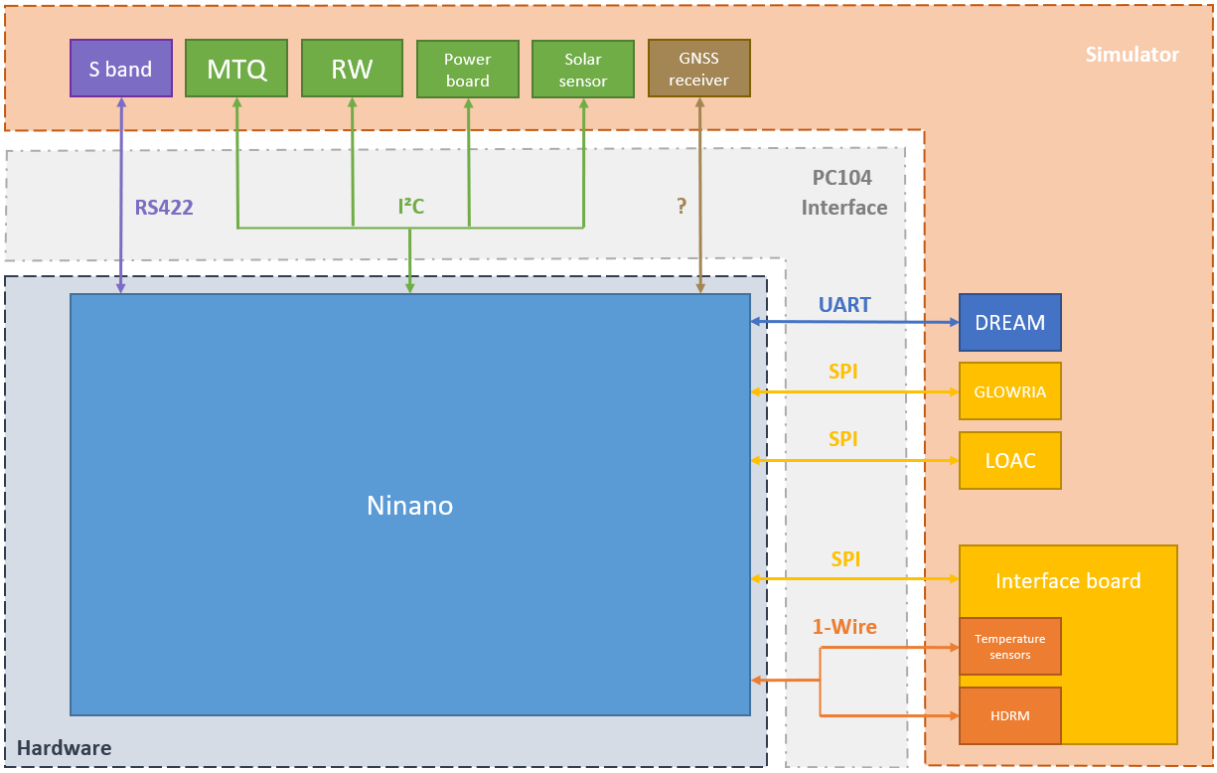
Hybrid test benches need an electronic interface that simulates the communication between the equipment and payloads of the satellite, modeled in the digital simulator, and the OBC. This interface implements the communication protocols of each equipment.

The AEROSAT needs for this interface are based on the two proposed architecture versions of the hybrid test bench: one version where all payloads are simulated, called the "digital payload test bench", and another version where some real payloads are used, called the "AIT test bench". These versions are presented in the following paragraphs.

Digital payload test bench

The digital payload bench only includes the OBC board as its physical part. Figure 17 shows the interfaces of this configuration.

Figure 17 – The hardware interfaces of the digital payload test bench.



Source: Author, 2021.

AIT test bench

The AIT bench includes the MIEL interface board and one or more payloads as hardware. Figure 18 shows an example of an AIT bench where the real LOAC payload is used. Figure 19 presents the AIT bench with the DREAM payload.

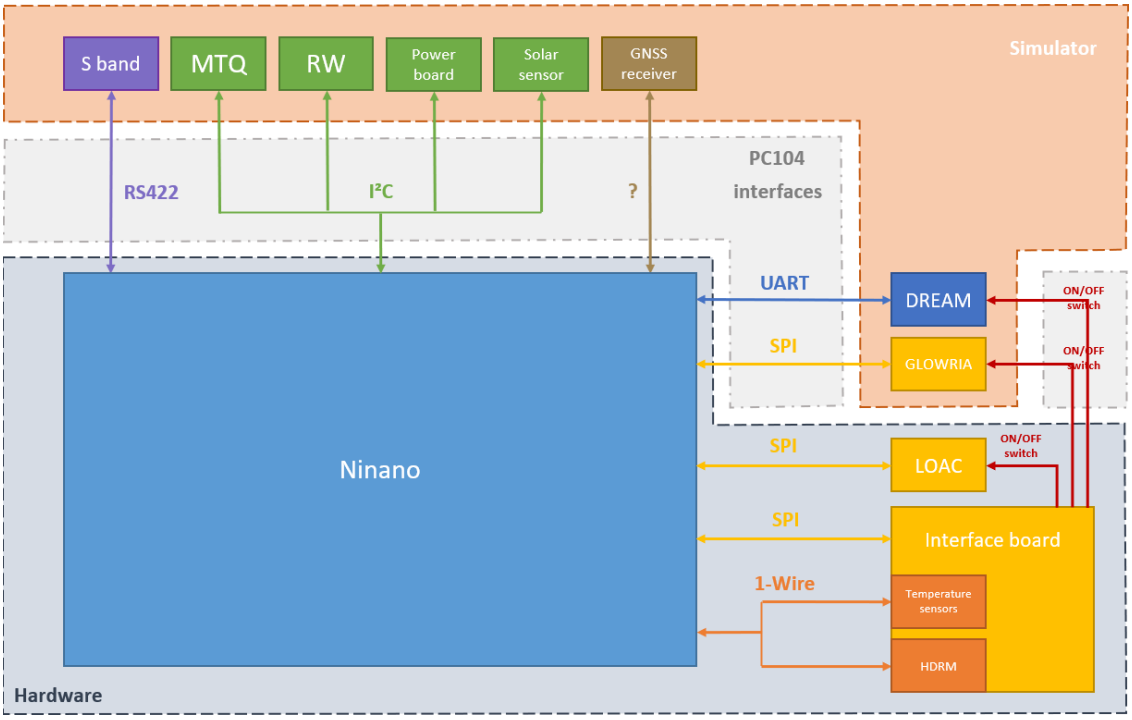
AEROSAT test bench interface protocols

Table 6 summarizes all the interface protocols necessary for meeting the needs of the AEROSAT test bench. Let it be noted that, to the end of the internship, some information related to the GNSS receiver, the temperature sensors and the HDRM was still missing.

Two solutions for the hardware communication interface were considered: the reuse of the electronic boards designed for the EyeSat test bench or the design of new interface bricks by CNES's department DSO/TB/ET[1]. DSO/TB/ET's solution was chosen because it is meant to be CNES's standard for all future missions. This choice is explained in detail by the author in a previous work (CAMARGO DA SILVA, 2021a).

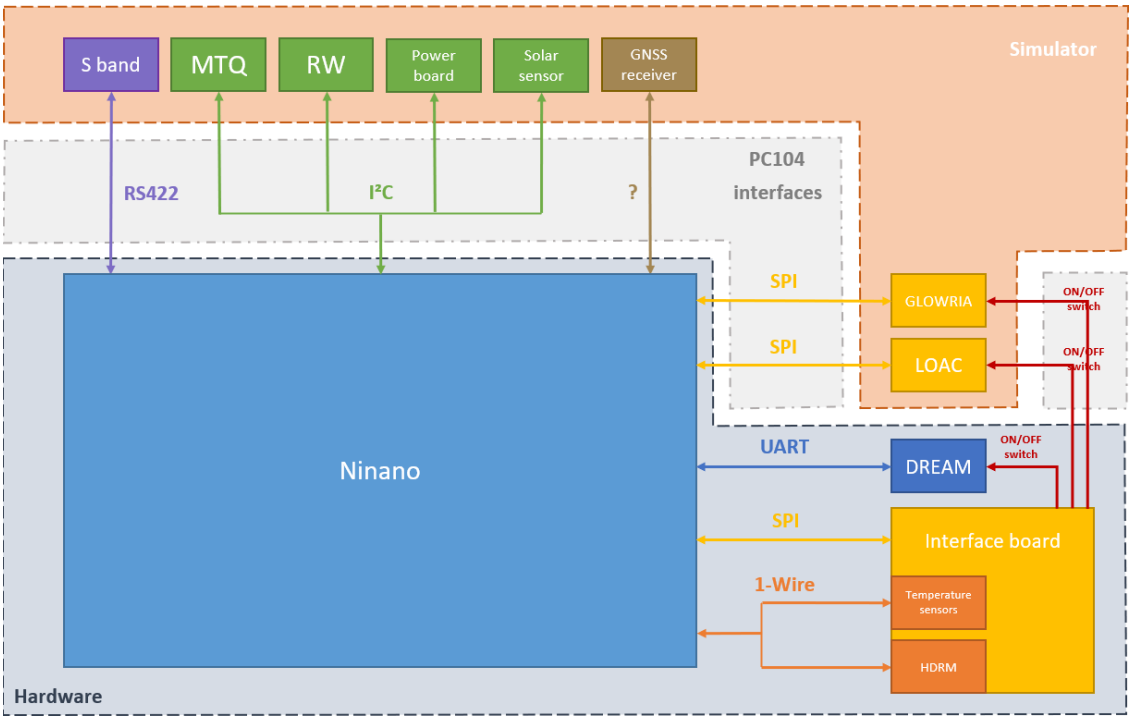
[1] DSO/TB/ET is CNES's department responsible for the design and development of electronic hardware.

Figure 18 – The hardware interfaces of the AIT test bench, LOAC example.



Source: Author, 2021.

Figure 19 – The hardware interfaces of the AIT test bench, DREAM example.



Source: Author, 2021.

Table 6 – Hardware interface needs for the AEROSAT test bench

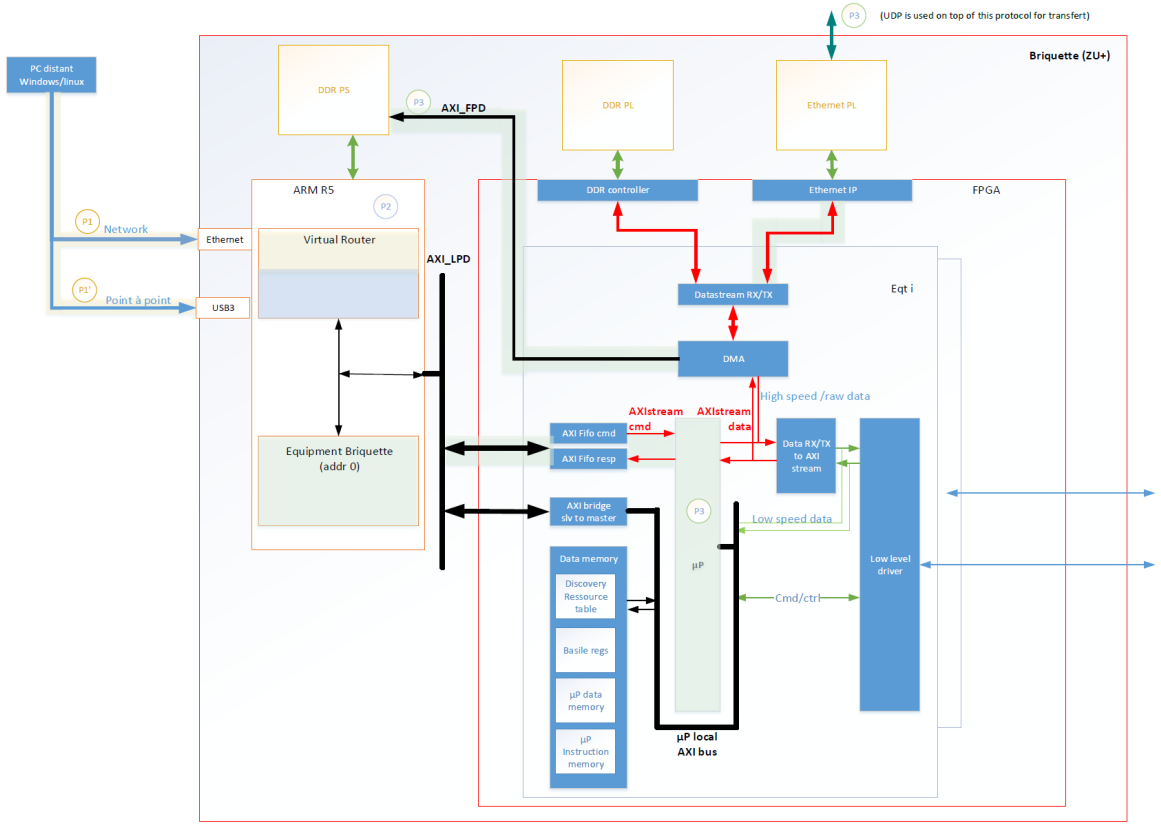
Equipment	**Protocol**	**Description**
Syrlinks EWC31	RS422/TMTC	Bande S
ISISPACE iMTQ Magnetorquer Board	I²C	Magnetorquers
Sinclair Interplanetary RW-0.003	I²C	Reaction wheel
Power board	I²C	Power supply board
Solar MEMS nanoSSOC-D60	I²C	Solar sensor
NovAtel OEM719	?	GNSS receiver
?	1-Wire	6 to 10 temperature sensors
?	1-Wire	4 HDRM
Interface board	SPI	Interface board between Ninano and the ERS, Resistack and Microbalance payloads
GLOWRIA	SPI	The GLOWRIA payload
LOAC	SPI	The LOAC payload
DREAM	UART	The DREAM payload
GLOWRIA	ON/OFF	GLOWRIA's ON/OFF switch
LOAC	ON/OFF	LOAC's ON/OFF switch
DREAM	ON/OFF	DREAM's ON/OFF switch

Source: Author, 2021.

4.1.2.3 DSO/TB/ET's solution

The new hardware communication interface board developed by DSO/TB/ET is called *briquette* and is an evolution of the old CNES hardware communication brick. It consists of a main brick, called *carte mère*, "motherboard", which communicates with the simulator via an Ethernet connection, and smaller bricks called *cartes filles*, "daughterboards", which connect to the OBC. The briquette includes an FPGA connected to an ARM processor. Figure 20 shows a global view of the architecture of the briquette as of 19/10/2021.

Figure 20 – Architecture of the briquette as of 19/10/2021.



Source: DSO/TB/ET, 2021.

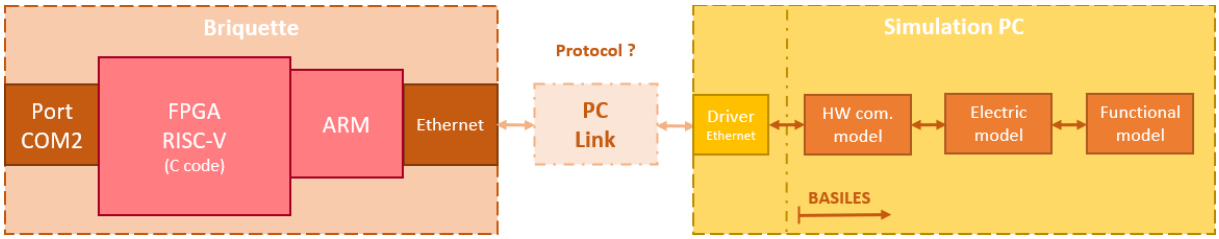
Let it be noted that this is not the final version of the briquette, as some design changes and definitions were still needed when this version was presented. The implementation of the briquette and the necessary developments are better discussed in Section 4.2.

4.2 THE NEW CNES BRICK

This section presents the new hardware communication brick that was proposed to replace the old CNES brick and that will be used in the AEROSAT test bench. The

design chosen for the new CNES brick is based on the implementation of a RISC-V microprocessor (ABOUT. . . , 2022) in the briquette's FPGA. Figure 21 presents the proposed architecture for test benches using the new brick.

Figure 21 – Test bench architecture with the new briquette.



Source: Author, 2021.

4.2.1 The new design's advantages and disadvantages

The **advantages** of this design are:

- The briquette is 100% generic.
- The briquette's behavior can be programmed in C thanks to the RISC-V microprocessor. This is an advantage because the same person who develops the model is also able to write the briquette code. If the program had to be written in VHDL, an extra developer, specialized in VHDL, would be necessary to write it.
- All delay problems are solved.
- There is no need for the simulation models to be synchronized with the C code in the briquette.

Some possible **disadvantages** were also listed:

- The debugging of programs running in the RISC-V microprocessor could be relatively difficult.
- The developer of the model would be responsible for the logging of physical messages.
- The communication protocol between the FPGA and the electric model would be specific to each model.
- The transition to a fully digital simulator in which even the OBC is emulated may require changes to simulation models.

4.2.2 Necessary changes for the implementation of the briquette

There are also a few necessary changes to the version of the **briquette** presented by DSO/TB/ET:

- The FPGA is flashed with a RISC-V microprocessor.
- The ARM processor must implement a server allowing to upload a RISC-V program.
- The ARM processor must implement a server allowing to retrieve information about the physical link, such as IsConnected for Spacewire and SetSlaveId for I^2C.
- The ARM processor must implement a server allowing to configure and retrieve generic information, such as reception buffer size and link speed.

Finally, some changes to the **simulation computer** are necessary comparing to what was needed for the old CNES brick:

- An Ethernet driver is necessary.
- The hardware communication model must be modified to implement the communication protocol to be used with the briquette.
- The electric model must be completely rewritten.
- The program to be run in the RISC-V microprocessor must be written.
- The simulation computer must have a RISC-V compiler that is compatible with the chosen processor installed.

4.2.3 The briquette communication protocol

For the communication protocol between the briquette and the simulation computer, two solutions have been analyzed: UDP and TCP. On the one hand, the advantage of UDP is that one message per packet can be sent, whereas with TCP several messages are sent in the same frame, which would be much more difficult to implement in the briquette since it would be necessary to manage the separation of these messages. On the other hand, the advantage of TCP is that it ensures that a packet has arrived at its destination, which is essential for simulators. The solution proposed by DSO/TB/ET in order to take advantage of both protocols is to create an "enhanced UDP" protocol with message acknowledgment. This could be done through acknowledgment messages. In this case, it would have to be ensured that the acknowledgment messages are also received. A UDP solution is not a problem for digital simulators as

long as it is possible to guarantee an ordered reception of messages, that is, as long as there is a mechanism to handle the loss of a message and the loss of an acknowledgment message. DSO/TB/ET will still do some tests with TCP to analyze its performance. It has been agreed that at least an enhanced UDP protocol is needed to detect the loss of an acknowledgment packet.

4.3 DIGITAL SIMULATOR DEVELOPMENT

The digital simulator is one of the most important parts of a satellite test bench, as this is where model instances that simulate the space environment, the satellite's orbit, its equipment and its payloads run. AEROSIM, short for AEROSAT simulator, has two versions up to now: AEROSIM 0, which includes only environment, orbit and dynamics models, and AEROSIM 1, which complements the previous version with AOCS models.

4.3.1 Development strategy

This section describes some practices, procedures and choices followed throughout the development of AEROSIM and its models.

4.3.1.1 Git

Git is one of the most used version control systems, and it has also been used in the development of the AEROSAT test bench. A *Group* dedicated to the development of AEROSIM and its models was created in Nanolab-Academy's space in the CNES GitLab. Specific *Projects* for AEROSIM 0, AEROSIM 1 and each one of the AOCS models were also created. All changes have been tracked following a semantic versioning scheme in which new versions receive a tag in the vMAJOR.MINOR.PATCH format. This scheme is explained in detail in Appendix B.

4.3.1.2 Meld

Meld is a software development tool that allows one to easily compare and merge code (WHAT..., 2022). This is especially useful for the development of simulation models, since one can use Meld to fill the skeleton code generated by SIMSAT with any changes made to the previous version of the model. It can also be used to compare the Tcl code used for different versions of a simulator. The use of Meld made the development of AEROSIM and its models considerably faster than having to change entire pieces of code manually.

4.3.1.3 Development choices

Some choices were made as a way to facilitate and standardize the development of AEROSIM and its models. They are described in this section.

Simulation scenarios in Tcl

All versions of AEROSIM have their scenario configuration written in Tcl instead of the native BASILES .sce files. This allows the simulator developer to have a more precise control over the behavior of the simulation.

Models follow SMP2

All models developed for AEROSIM follow SMP2. This way it is easier to reuse SMP models developed for previous projects, and the models created for AEROSAT will also be available for future missions. These characteristics are essential for a generic nanosatellite test bench platform.

Automatic data propagation and the Dataflow extension

The Dataflow extension is used by all AEROSIM models in order to make the data propagation from outputs to inputs easier. This is necessary because, by default, one needs to schedule the update of every single connection in an SMP simulator by means of simulation tasks and events. With this extension, this update is made automatically at each simulation step without the need for scheduling.

4.3.2 Orbit, environment and dynamics models

AEROSIM's orbit, environment and dynamics models are reused from the orbit extrapolator simulator described in Section 3.3.3. Almost all of these models come from PATRIUS, a library of space simulator models developed by CNES written in Java that can be adapted to be used in BASILES through a C++ interface.

4.3.2.1 Orbit

OrbitNumInt

The PATRIUS library provides 4 models for orbit calculation (ORBIT. . . , 2018). AEROSIM uses the numeric integration orbit calculation model. The OrbitNumInt model numerically integrates the accelerations on the satellite to obtain its state in the Inertial Reference Frame (IRF). It receives as input the sum of the satellite accelerations.

It implements an interface provided by the IntegratorRK4Isis[2] model to perform this integration.

OrbitConversion

The OrbitConversion model computes orbital parameters, the state of the satellite in the Earth Reference Frame (ERF) and the quaternions corresponding to the rotations between several reference frames. It receives as input the vector containing the state of the satellite in the IRF reference frame.

4.3.2.2 Environment

AlbedoIRIrradiance

The AlbedoIRIrradiance model determines the albedo and infrared radiation coming from the Earth at the satellite position. It receives as input the position of the Sun and the state of the satellite.

AtmosphericDensity

The AtmosphericDensity model determines the atmospheric density at the satellite position as well as the velocity, molar mass and temperature of the air. It receives as input the position of the Sun and the state of the satellite.

EarthGravity

The EarthGravity model determines the acceleration due to the Earth's gravity acting on the satellite using spherical harmonics. It has as input the state of the satellite. It implements an interface provided by the IntegratorRK4Isis model to perform this calculation.

Eclipse

The Eclipse model determines the direction of the Sun in the Spacecraft Frame (SCF) and the percentage of sunlight visible from the satellite. It receives as input the position of the Moon, the position of the Sun and the state of the satellite.

[2] IntegratorRK4Isis is a model that implements numerical integration through the Runge-Kutta fourth-order method.

ThirdBodyGravity

The ThirdBodyGravity model calculates the acceleration due to a third body acting on the satellite. It receives as input the state of the satellite and the position and gravitational constant of the body. AEROSIM uses two instances of it: one for the Sun and another one for the Moon.

MagneticField

The MagneticField model calculates the intensity of the Earth's magnetic field in the IRF at the satellite position. It has as input the state of the satellite.

SolarIrradiance

The SolarIrradiance model determines the solar radiation at the satellite position. It receives as input the state of the satellite, the position of the Sun and the percentage of sunlight visible from the satellite.

ThirdBody

The ThirdBody model provides the positions of the Sun and the Moon in the IRF and the gravitational constants of these bodies.

TidalForces

The TidalForces model determines the gravitational acceleration due to tidal forces acting on the satellite by adding variations to the spherical harmonic approximation of the Earth's gravitational potential. It has as input the state of the satellite. It implements an interface provided by the IntegratorRK4Isis model to perform this calculation.

4.3.2.3 Attitude dynamics

DynamicSat

There are 4 methods for attitude calculation (DYNAMICS. . . , 2018). AEROSIM uses the numerical integration method. The DynamicSat model determines the attitude quaternion of the satellite, the rotation quaternion from the IRF to the SCF and the sum of the accelerations acting on the satellite. It implements an interface provided by the IntegratorRK4Isis model to perform these calculations.

Body

The Body model represents a component of the satellite, a "body", with all its physical properties necessary for dynamics calculations. AEROSAT is modeled as a **single instance** of Body in AEROSIM since it is a relatively simple and small satellite.

ForceTorqueCalc

The ForceTorqueCalc model is **contained** in the **Body** model. It calculates the forces and torques applied to the satellite by the environment. It receives as input the data calculated by the orbit and environment models that are necessary for these calculations.

ExtForceTorque

The ExtForceTorque model is **contained** in the **ForceTorqueCalc** model. It allows to define a force or a torque applied to a Body instance. In the case of AEROSIM, each equipment model that generates torque is connected to an instance of ExtForceTorque. It is possible to connect several instances of ExtForceTorque to an instance of ForceTorqueCalc, so there is no limitation to the number of sources of force or torque applied to a body.

4.3.2.4 Model interfaces

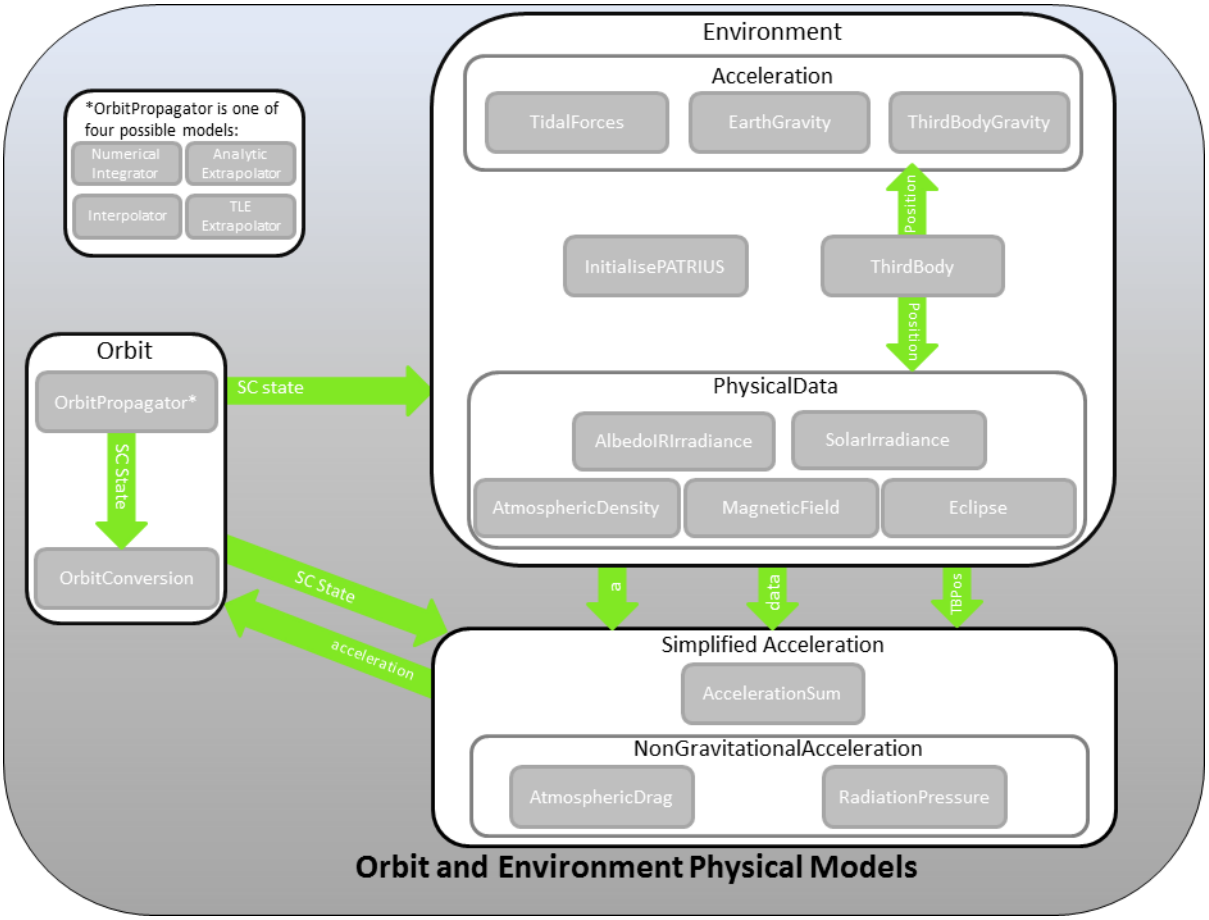
Figure 22 presents a global view of the interface between orbit and environment models. The orbit models compute the state of the satellite and send it to the environment models, which compute all environmental parameters. The "Simplified Acceleration" models are not used in AEROSIM.

Figure 23 shows the interfaces of the dynamics models. The attitude dynamics models receive from the orbit and environment models the state of the satellite and all environmental variables. In addition, they also receive the torque values produced by the satellite's actuators. They can thus calculate the attitude and accelerations on the satellite and send these results to the orbit, environment and equipment models. The dynamics models are interconnected using interface models such as IBody, IForceTorqueCalc and IForce.

4.3.3 AOCS models

AEROSAT's AOCS is composed of three equipment: an eletronic board including three magnetorquers and a magnetometer, henceforth referred to as **magnetoboard**, a **Sun sensor** and a **reaction wheel**. In order to model this system as close to reality as

Figure 22 – Orbit and environment model interfaces.



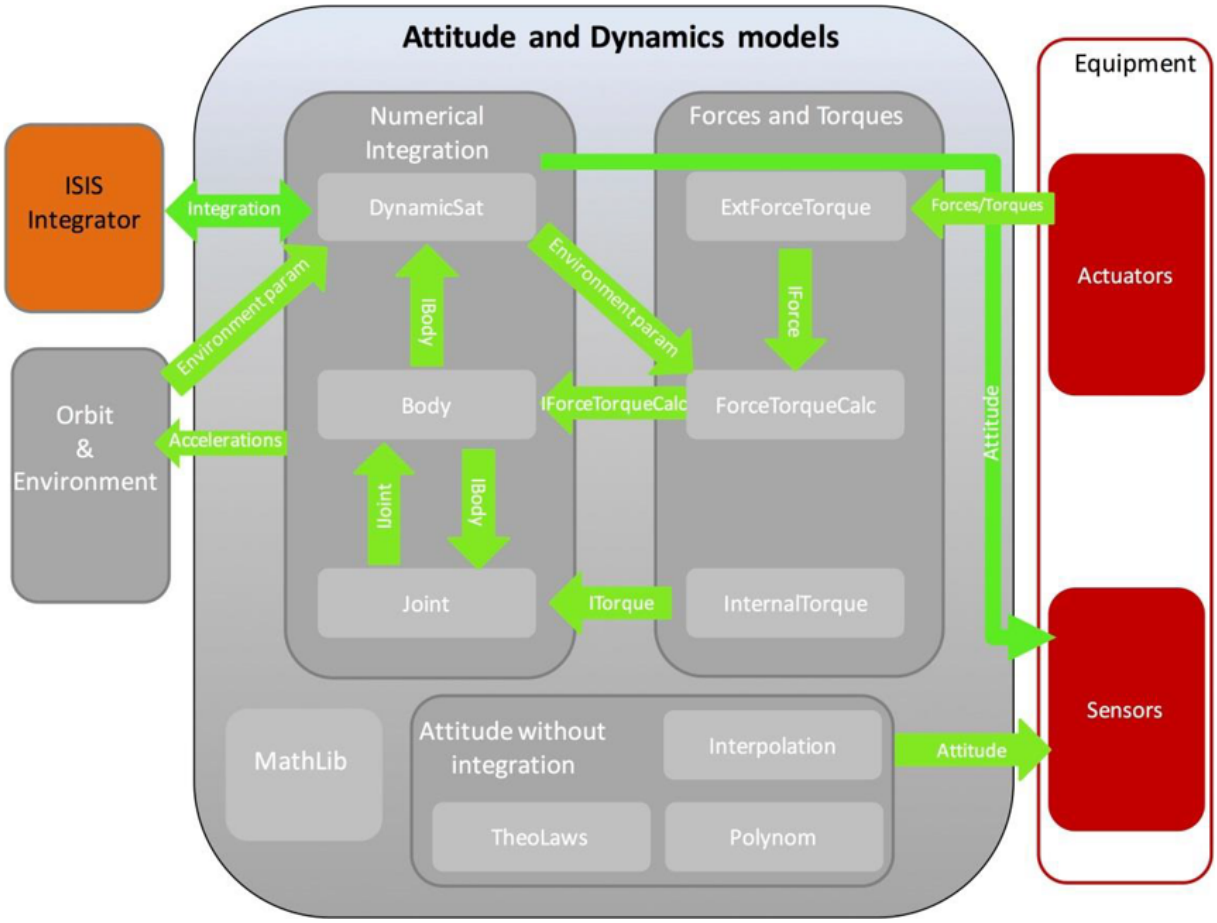
Source: Orbit and Environment - Interface Control Document (ORBIT. . . , 2018).

possible, one model for each one of them was designed. Each equipment was modeled following the description of its behavior given by the manufacturer. Each model was written in C++ within the skeleton of source code generated by SIMSAT.

4.3.3.1 iMTQ Magnetorquer Board

The iMTQ Magnetorquer Board is a magnetoboard developed by ISISPACE. Figure 24 shows the two sides of the board. It includes a three-axis magnetometer and three magnetorquers: two torque rods and one air core coil. iMTQ's **magnetometer** detects the intensity of the magnetic field in three orthogonal directions, and this measurement can be compared to a model of Earth's magnetic field in order to determine the attitude of the satellite. The three **magnetorquers** create a magnetic field that interacts with Earth's magnetic field. This interaction creates a torque that rotates the satellite. The intensity of the magnetic field generated by the magnetorquers can be controlled so that the satellite is rotated to a target orientation.

Figure 23 – Dynamics model interfaces.

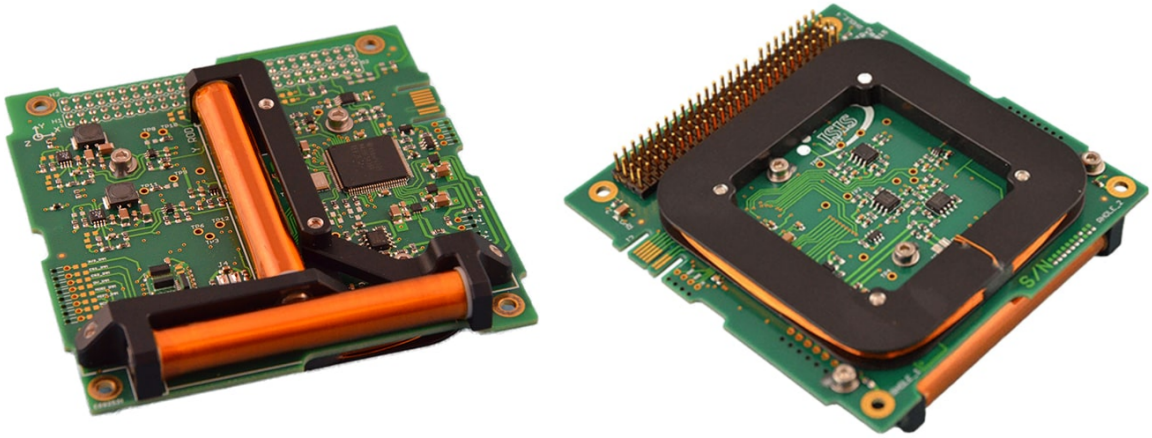


Source: Dynamics of Attitude - Interface Control Document (DYNAMICS. . . , 2018).

Since EyeSat carries the same magnetoboard, the business logic[3] of the magnetoboard model comes from the models made for the EyeSat simulators, but the structure of its design was modified. Three separate models were used for EyeSat: one for the board in a global way, one for the magnetorquers and another one for the magnetometer. The board model only transmitted inputs and outputs to and from the other two models, which did all the functional work. Since in reality there is only one equipment, the board, and it contains the magnetorquers and the magnetometer, this logic was also adopted for the AEROSAT models. For AEROSAT there is a **Board** model, which manages the operations of its two internal models: **Magnetometers** and **Magnetorquers**.

[3] The business logic is the part of a program that "does something meaningful", as opposed to code that deals only with data flow or data storage, for example.

Figure 24 – Top (left) and bottom (right) faces of the iMTQ board.



Source: ISISPACE (IMTQ. . . , 2022).

Board

The Board model represents an electronic board containing three magnetorquers and a magnetometer. This model is based on the ISIS iMTQ board. It manages the magnetic dipole command to be applied to the magnetorquers and the magnetic field measurements of the magnetometer. It receives as input a magnetic dipole command, the duration of this command and the value of the magnetic field in the SCF at the position of the satellite.

Magnetorquers

The Magnetorquers model is contained in the Board model. It represents the system of three magnetorquers of the board. It calculates the torque generated by these magnetorquers. It receives as input a magnetic dipole command and the magnetic field in the SCF at the satellite position.

Magnetometer

The Magnetometer model is contained in the Board model. It represents the magnetometer of the board. It calculates the magnetic field at the satellite position in the magnetometer frame. It receives as input the magnetic field in the SCF at the satellite position.

4.3.3.2 nanoSSOC-D60

nanoSSOC-D60 is a two-axis Sun sensor developed by SOLARMEMS. Figure 25 presents an image of the sensor.

Figure 25 – The nanoSSOC-D60 Sun sensor.

Source: SOLARMEMS (NANOSSOC. . . , 2022).

It works by measuring two angles, called **alpha** and **beta**, related to the position of the Sun projected into the sensor's field of view. This sensor also generates an error code. Its value varies depending on its illumination conditions as follows:

- **0**: no error, the angles have been calculated successfully.
- **10**: no radiation detected, the angle measurements should not be considered.
- **11**: there is radiation passing through the field of view, but it is below 1366 W/m² minus a 20% tolerance of this value, meaning that the sensor sees only the Earth.
- **12**: there is radiation passing through the field of view, but it is above 1366 W/m² plus a 20% tolerance of this value, meaning that the sensor sees both the Earth and the Sun.
- **13**: a light source is detected, but it is outside the field of view.

Since nanoSSOC-D60 is a very simple sensor, it is modeled as a single model, called **SunSensor**.

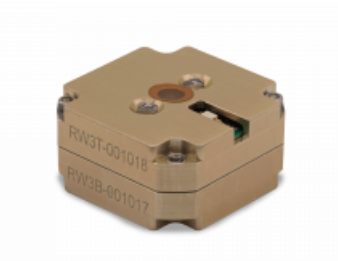
SunSensor

The SunSensor model represents a solar sensor. This model is based on the nanoSSOC-D60 sensor by SOLARMEMS. It calculates two angles measured from the position of the Sun relative to the satellite. Each measurement is accompanied by an error code that also depends on the albedo irradiance flux of the Earth. It receives as inputs the solar irradiance, the albedo irradiance flux from the Earth and the directions of these two fluxes.

4.3.3.3 RW-0.003

RW-0.003 is a small reaction wheel developed by Sinclair Interplanetary. It is presented in Figure 26. When it turns, it creates angular momentum that is used to control the attitude of the satellite. It is also a simple equipment modeled with a single model, called **ReactionWheel**.

Figure 26 – The RW-0.003 reaction wheel.



Source: Sinclair Interplanetary (REACTION. . . , 2022).

ReactionWheel

The ReactionWheel model represents a reaction wheel. This model is based on the RW-0.003 wheel by Sinclair Interplanetary. It calculates the angular velocity, the torque and the angular momentum generated by the reaction wheel. It receives as input an angular velocity command.

5 RESULTS

The results that followed the development of the models for the AEROSIM simulators and the implementation of the simulators themselves are described in this chapter.

5.1 MODEL VALIDATION

This section describes the validation of each one of the AOCS models and its results. The models were validated based on the concept of *recettes*. Recettes are simulators used to test models in a simplified environment in order to validate their behavior. Typically, one creates auxiliary models that generate made-up outputs that are connected to the model being tested. Then, a simulation is run so that the variables and outputs of the model are analyzed in order to be validated.

For the AOCS models, the results generated by recettes were compared to the corresponding expected output. If the results obtained through the recettes were exactly the same as expected, the model was considered validated.

5.1.1 Magnetoboard results

The following test models were created for the validation of the magnetoboard model:

- **BoardController**: model that generates actuation dipole commands and receives magnetic field and housekeeping data.
- **MagneticFieldSource**: model that generates magnetic field values.
- **TorqueSink**: model that receives a torque value.

Figure 27 presents the recette simulator for the magnetoboard model in BASILES.

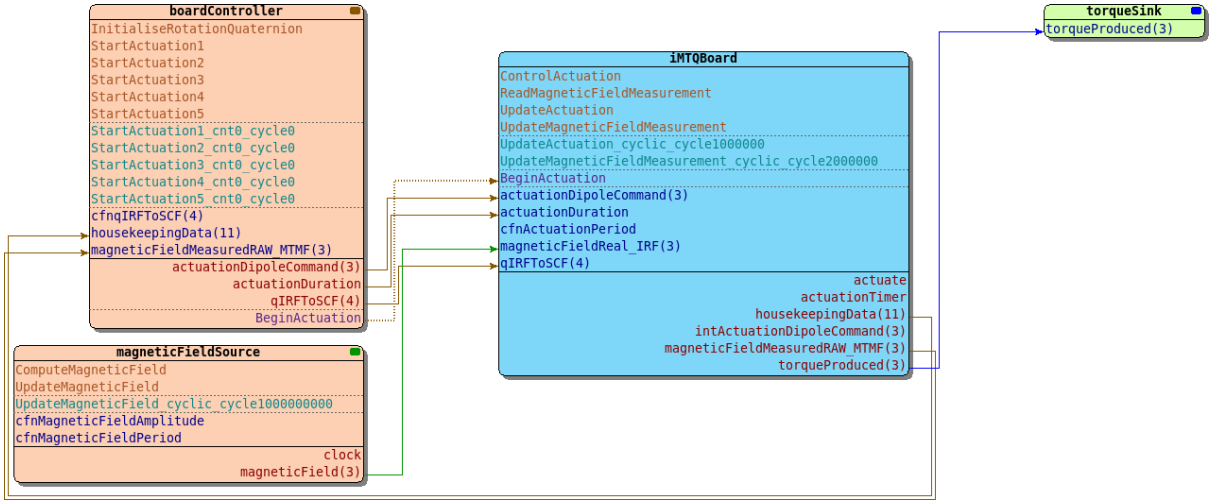
5.1.2 SunSensor results

In order to validate the behavior of the SunSensor model, the following test models were designed:

- **RadiationSource**: model that generates solar irradiance and Earth's albedo flux values.
- **SunPositionSink**: model that receives Sun position angles alpha and beta and the corresponding measurement error code.

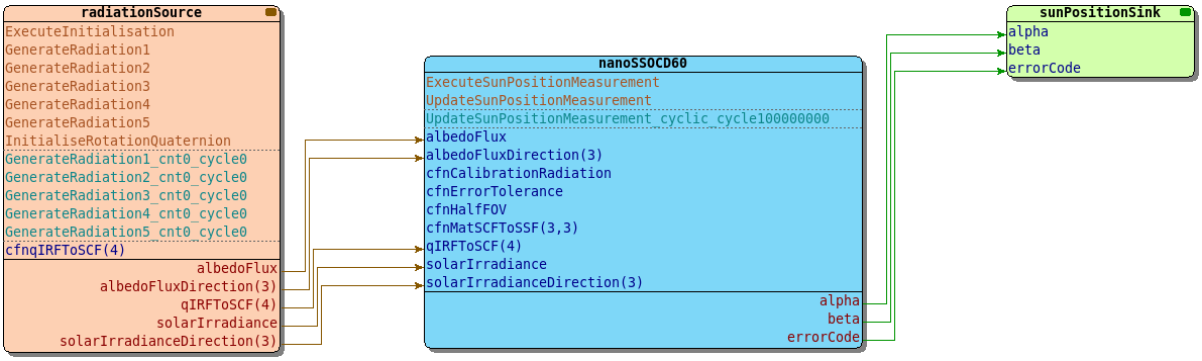
Figure 28 shows the recette simulator for the Sun sensor model in BASILES.

Figure 27 – The recette for the magnetoboard model, BASILES 4.0.2.



Source: Author, 2021.

Figure 28 – The recette for the Sun sensor model, BASILES 4.0.2.



Source: Author, 2021.

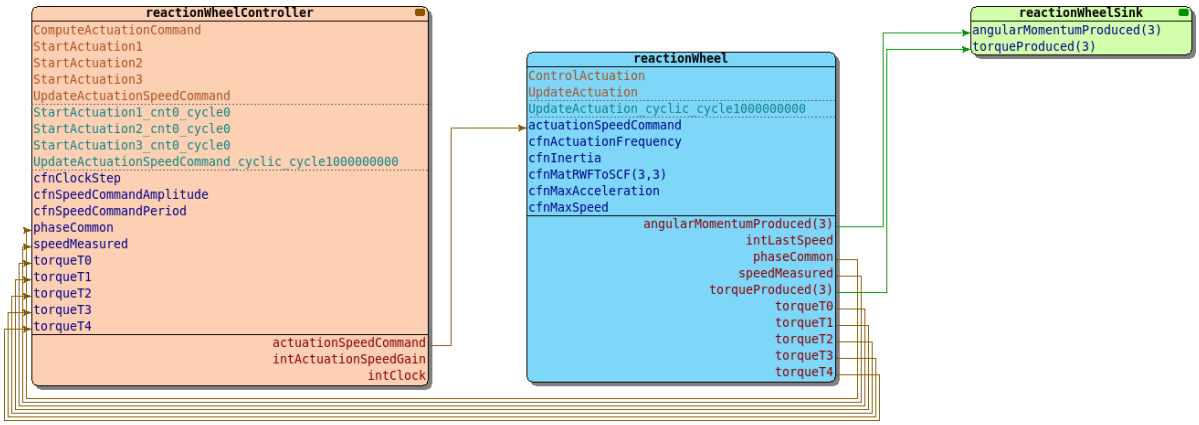
5.1.3 ReactionWheel results

For the validation of the ReactionWheel model, the following test models were developed:

- **ReactionWheelController**: model that generates speed commands and receives measured speed, winding voltage and torque data.
- **ReactionWheelSink**: model that receives the values of angular momentum and torque produced.

Figure 29 presents the recette simulator for the reaction wheel model in BASILES.

Figure 29 – The recette for the reaction wheel model, BASILES 4.0.2.



Source: Author, 2021.

5.2 FINAL MODEL INTERFACES

This section presents the final model interfaces after the validation of each one of the models.

5.2.1 Magnetoboard

Figure 30 shows the interfaces related to the magnetoboard model. The Board model receives from the environment models a magnetic field value. It also has as input from the OBC interface models a dipole command. It sends the generated torque value to the dynamics models and the magnetic field measurement and housekeeping data back to the OBC interface models.

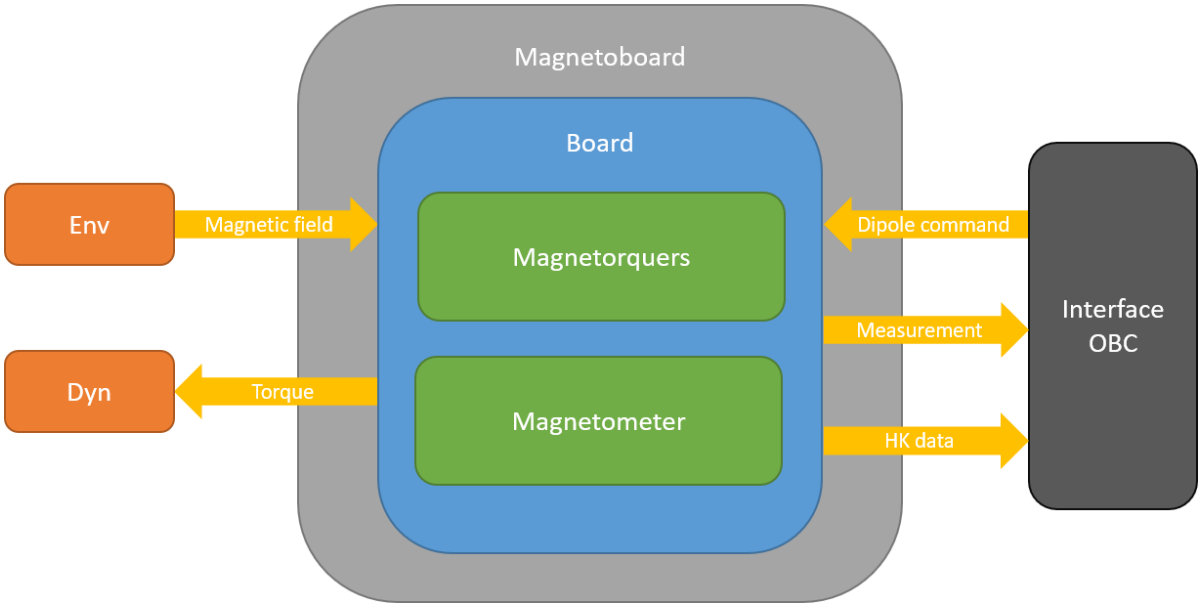
5.2.2 SunSensor

Figure 31 presents the interfaces related to the Sun sensor model. The SunSensor model receives from environment models the fluxes of albedo and solar irradiance (magnitude and direction) as well as the quaternion from the IRF to the SCF. It sends the resulting alpha, beta and error code to the OBC interface models.

5.2.3 ReactionWheel

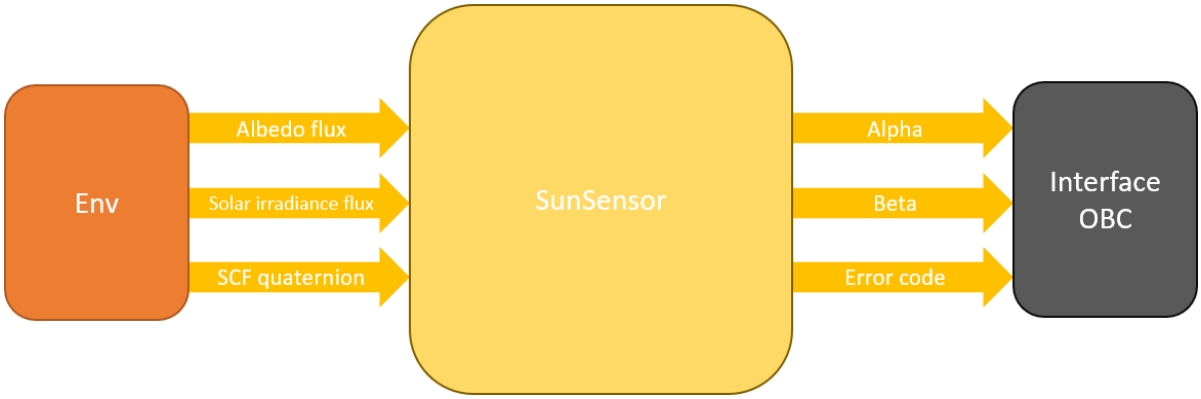
Figure 32 shows the interfaces related to the reaction wheel model. The ReactionWheel model receives from the OBC interface models a value of speed command. As a result of the application of this command, it sends to dynamics models the produced torque and angular momentum. It also sends the measured speed and torque values and the wheel's winding voltage back to the OBC interface models.

Figure 30 – The interfaces of the magnetoboard model.



Source: Author, 2021.

Figure 31 – The interfaces of the Sun sensor model.



Source: Author, 2021.

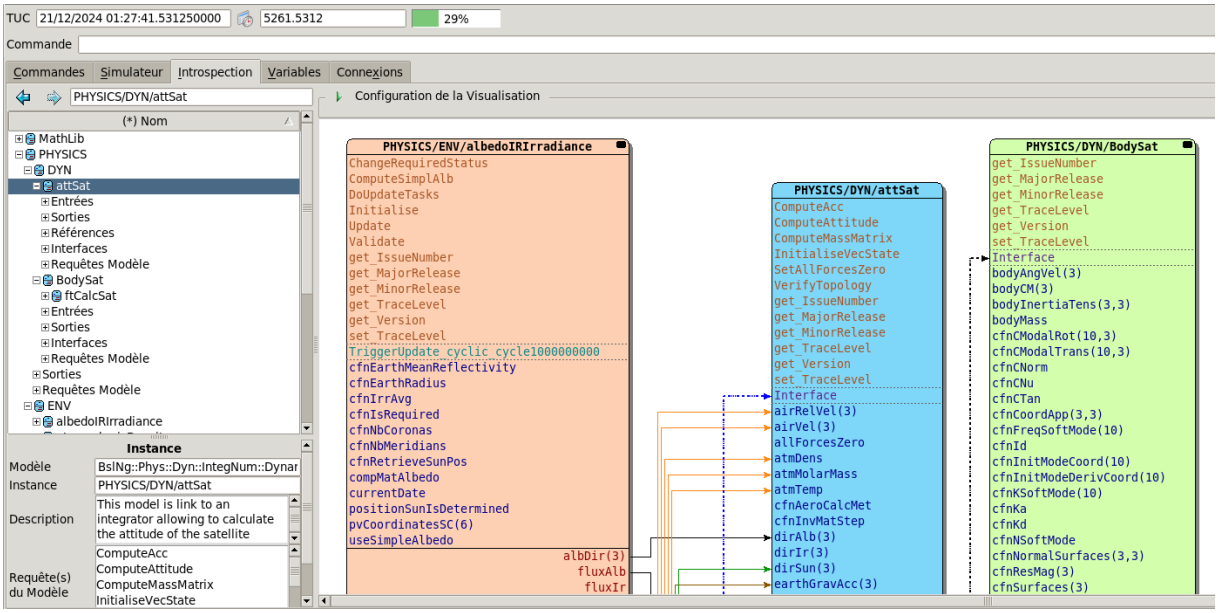
5.3 AEROSIM 0 RESULTS

AEROSIM 0 was created as an orbit extrapolator simulator. It is an adaptation of the orbit extrapolator simulator described in Section 3.3.3. All models used by AEROSIM 0 are the same used by the orbit extrapolator simulator that serves as its basis, with the difference that the orbit and dynamics instances are modified so that their parameters correspond to the ones of the AEROSAT mission. Figure 33 presents a screenshot showing the instances corresponding to the models AlbedoIRIrradiance, DynamicSat and Body during a simulation with AEROSIM 0.

Figure 32 – The interfaces of the reaction wheel model.

Source: Author, 2021.

Figure 33 – AlbedoIRIrradiance, DynamicSat and Body instances in AEROSIM 0, BASILES 4.0.2.

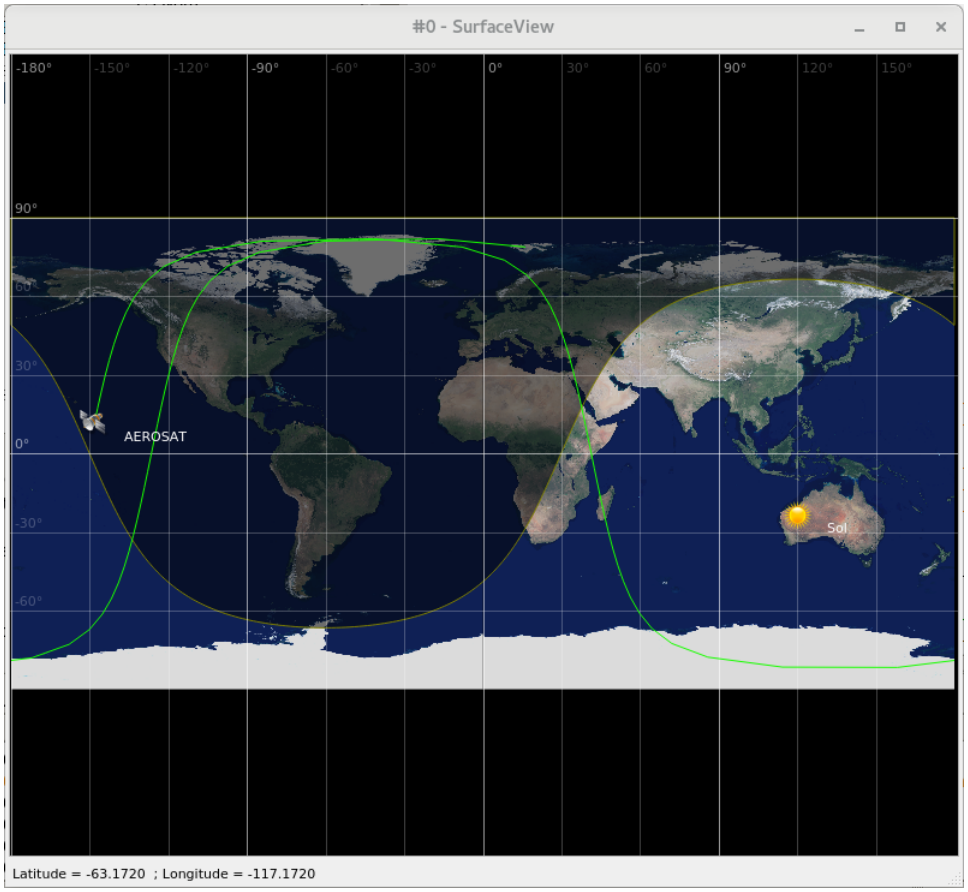


Source: Author, 2021.

5.3.1 AEROSIM 0 validation and results analysis

As an orbit extrapolator simulator, AEROSIM 0 can be used to predict AEROSAT's position, velocity and attitude over time given an initial state. The simulator was validated by comparing the resulting orbital parameters of the simulation to those of AEROSAT, which are described in Section 3.8.4. Since AEROSIM 0 proved to predict successfully the orbit of the satellite based on this comparison, the simulator was considered validated. The resulting data can be represented graphically through VTS and Celestia, as shown by Figure 34 and Figure 35.

Figure 34 – Orbit path of AEROSAT by AEROSIM 0, VTS 3.5.1.



Source: Author, 2021.

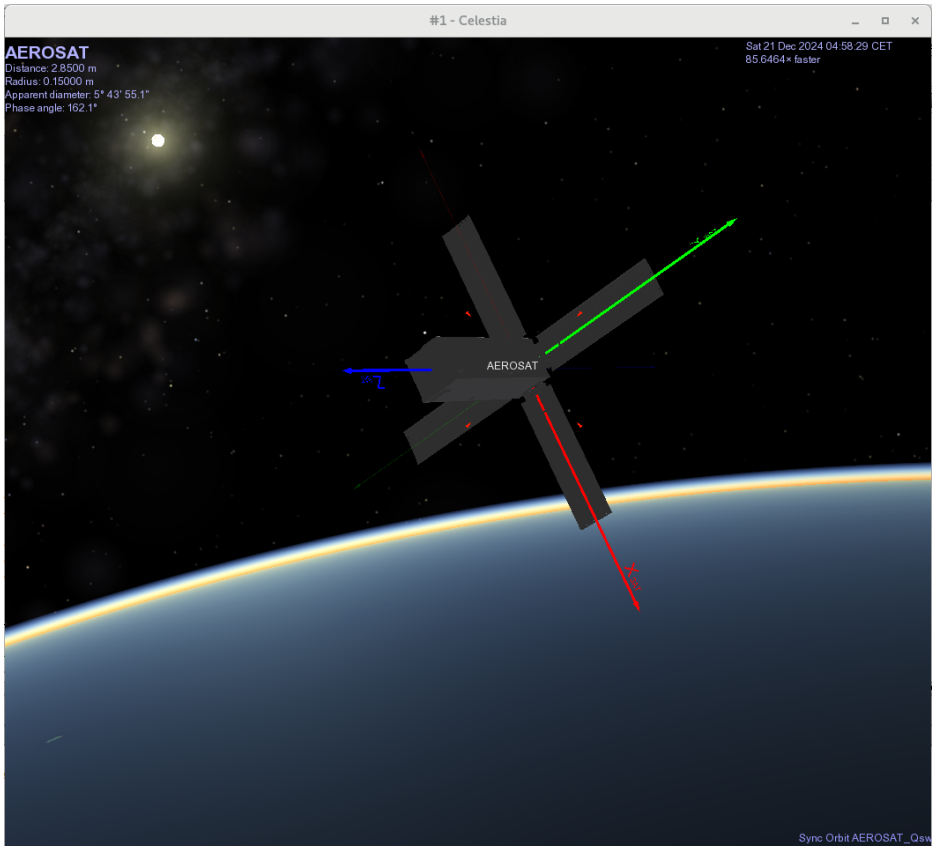
5.4 AEROSIM 1 RESULTS

AEROSIM 1 is an evolution of AEROSIM 0 that includes the AOCS equipment models. Figure 36 presents an example of how these models are included in the simulator by showing the connections of iMTQBoard's Board instance.
The connections to the OBC interface models are not present because these models are still to be developed.

5.4.1 AEROSIM 1 validation and results analysis

Since the individual behavior of each one of the AOCS models was already validated through their recettes, only the transmission of data between model instances within the simulator needed validation. Therefore, AEROSIM 1 was tested by checking whether the inputs and outputs of each one of the AOCS models were really being transferred between model instances. Since all data was being transmitted normally, the simulator was also considered successfully validated. Figure 37 presets the simulation results graphically represented as the orbit path of AEROSAT. Finally, Figure 38 shows

Figure 35 – AEROSIM 0 animation screenshot, VTS 3.5.1 and Celestia.



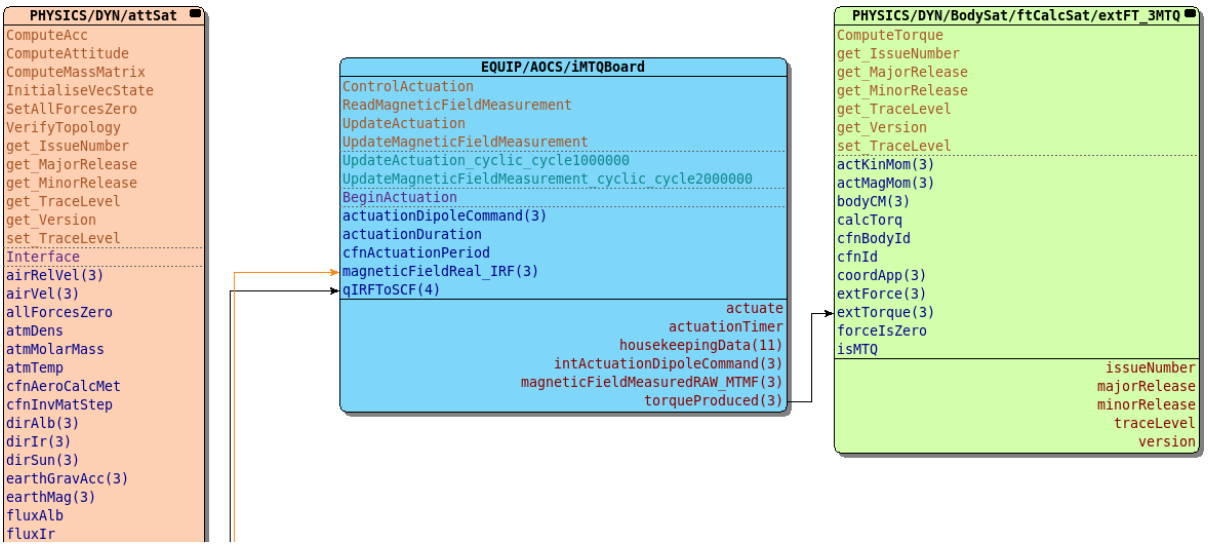
Source: Author, 2021.

a screen capture of an AEROSIM 1 simulation in Celestia.

5.5 AEROSAT TEST BENCH FINAL ANALYSIS

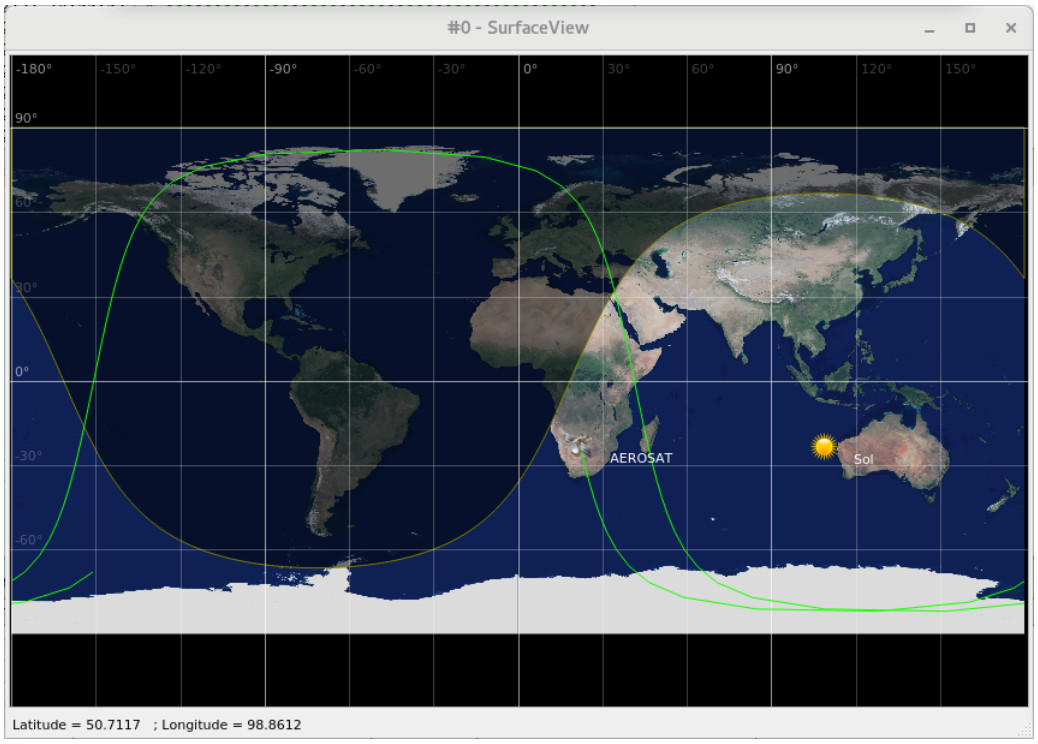
The AOCS models created for AEROSIM follow SMP2, which means that they can be easily reused in other missions and comply with the most modern European space simulation standard. Furthermore, the AOCS is an essential part for any nanosatellite, which means that these models represent an important part of a generic simulator. Moreover, AEROSIM 0 and 1, being able to successfully predict the satellite's orbit, can be used as foundation for any other kind of space simulator whose objective is to test more specific behaviors and systems of a nanosatellite. Thus, this work fulfills its objective of continuing the development of the AEROSAT simulator and, consequently, the development of a generic satellite test bench platform, by adding these important and essential contributions to it. On the other hand, the simulator is still not complete, since other portions of the satellite, such as its communication, thermal and electrical systems are still to be modeled in future work.

Figure 36 – The iMTQBoard instance connected to other models in AEROSIM 1, BASILES 4.0.2.



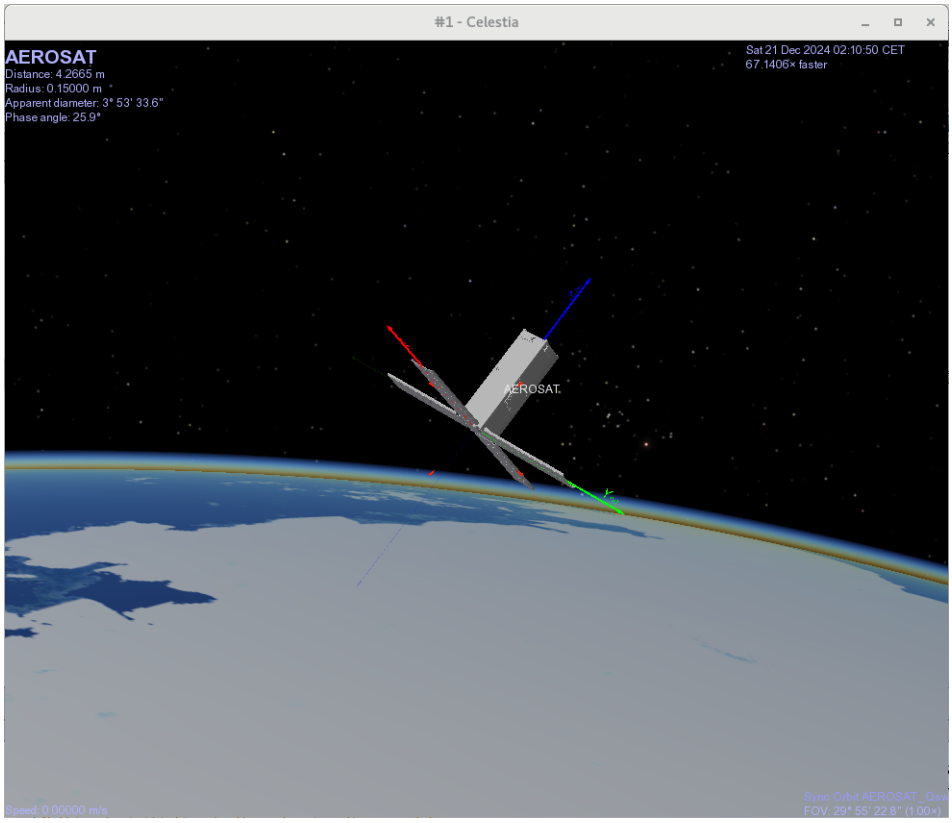
Source: Author, 2021.

Figure 37 – Orbit path of AEROSAT by AEROSIM 1, VTS 3.5.1.



Source: Author, 2021.

Figure 38 – AEROSIM 1 animation screenshot, VTS 3.5.1 and Celestia.



Source: Author, 2021.

CONCLUSION

The objective of this work was to continue the development of a test bench for the AEROSAT mission, which had already been started by the previous intern. The following advancements were made through this work:

- The architecture versions of the AEROSAT test bench were defined.
- The hardware communication interface for the AEROSAT test bench was chosen and its design was refined.
- AEROSIM 0, a first orbit extrapolator simulator for AEROSIM, was created.
- The AOCS models representing AEROSAT's magnetoboard, Sun sensor and reaction wheel were designed.
- A first version of a simulator containing AEROSAT's AOCS, AEROSIM 1, was developed.

As a result, the AOCS models that were designed are all SMP2 compliant, which allows for model reuse and portability. AEROSIM 0 is a functional orbit extrapolator simulator, which can successfully predict the satellite's position and velocity over time, and thus can be used to help in the development and planning of the AEROSAT mission in general. AEROSIM 1 is a working AOCS simulator which later on, with the conclusion of the development of the FS, will play the important role of validating the nanosatellite's attitude control software and algorithms.

Moreover, the outcomes of this work are essential for one of the most important objectives of Nanolab-Academic with this project, which is the creation of a generic nanosatellite simulator test bench for helping students from French CSU to develop their own cubesats. The AOCS models can be easily reused for other missions thanks to being SMP2, and the AEROSIM simulators can also be effortlessly readapted to other nanosatellites by simply changing the simulation parameters and adding or subtracting some model instances.

From a personal point of view, this work has also been an excellent first experience working in the space industry. I had the opportunity to put into practice and develop my programming skills, especially in C++, while being tutored and helped by CNES's and Spacebel's experts in the field of space simulators. I have acquired extremely valuable knowledge on the development of nanosatellites and satellite test benches, and this will certainly be significantly useful for me as I continue following my path through space engineering.

FUTURE WORK

The main aspects of the future work necessary for the development of the AEROSAT test bench are:

- The design of other simulation models for the test bench architecture with the new briquette if necessary (such as the hardware communication and electric models).
- The implementation of a prototype of the new briquette with a model that has already been developed, such as the magnetoboard model.
- The development of models of the payloads and other parts of the satellite's platform, such as the power, thermal control and communication systems. It may be possible to readapt models made for previous missions. Possible sources of models are the ones made for the missions SVOM (thermal control) and ANGELS (power and communication).
- Updates to the parameters of some instances, as several parameters are still to be defined as the satellite is developed. An Excel file meant to be used by the next interns was created in order to keep track of all the parameters in AEROSIM.
- The choice of the RISC-V microprocessor to be flashed in the FPGA for the hybrid test bench.
- The definition of the connection between the FPGA and the COM ports (use of the ISIS 305 interface).
- The complete definition of the enhanced UDP protocol with message acknowledgment.

REFERENCES

AMOUROUX, Manuel. **Testbed Engineer for AEROSAT**. Institut supérieur de l'aéronautique et de l'espace. Apr. 2021a.

AMOUROUX, Manuel. **Tuto portage modèle BASILES natif en SMP2**. Centre national d'études spatiales. Apr. 2021b.

BOIVINET, Xavier. **Lancement réussi pour Angels, premier nano-satellite industriel français**. L'Usine Nouvelle. Dec. 2019. Available from: https://www.usinenouvelle.com/article/lancement-reussi-pour-angels-premier-nano-satellite-industriel-francais.N1815037. Visited on: 22 Nov. 2022.

CAMARGO DA SILVA, Lucas. **Choix de l'interface entre le simulateur BASILES et la carte OBC du banc de test AEROSAT**. Centre national d'études spatiales. Dec. 2021a.

CAMARGO DA SILVA, Lucas. **Compte rendu : réunion de brainstorming sur la nouvelle brique HW du CNES**. Centre national d'études spatiales. Oct. 2021b.

CAMARGO DA SILVA, Lucas. **État final du banc de test EyeSat**. Centre national d'études spatiales. Oct. 2021c.

LES 5 domaines d'intervention du CNES. Centre national d'études spatiales. Oct. 2021. Available from: https://cnes.fr/fr/web/CNES-fr/3358-les-5-themes-dapplications.php. Visited on: 29 Nov. 2021.

NANOLAB-ACADEMY. Centre national d'études spatiales. Oct. 2021. Available from: https://nanolab-academy.cnes.fr/fr. Visited on: 30 Nov. 2021.

SUPERCAM. Centre national d'études spatiales. Jan. 2021. Available from: https://supercam.cnes.fr/fr. Visited on: 29 Nov. 2021.

DEB, Chaymaa. **EyeSat, un nanosatellite et une réussite technologique pour les stagiaires du CNES**. Techniques de l'Ingénieur. Aug. 2020. Available from: https://www.techniques-ingenieur.fr/actualite/articles/eyesat-un-nanosatellite-et-une-reussite-technologique-pour-les-stagiaires-du-cnes-82806/. Visited on: 30 Nov. 2021.

LANCEMENT réussi du satellite de télécommunications militaires Syracuse 4A. Direction générale de l'armement. Nov. 2021. Available from: https://www.defense.gouv.fr/dga/actualite/lancement-reussi-du-satellite-de-telecommunications-militaires-syracuse-4a. Visited on: 29 Nov. 2021.

ECSS-E-ST-40-07C. European Cooperation for Space Standardization. Mar. 2020.

IMTQ Magnetorquer Board. ISISPACE. 2022. Available from: https://www.isispace.nl/product/isis-magnetorquer-board/. Visited on: 27 Nov. 2022.

JOHNSTONE, Alicia. **CubeSat Design Specification**. Cal Poly. Feb. 2022.

WHAT is Meld? Meld. Available from: https://meldmerge.org/help/introduction.html. Visited on: 19 Dec. 2022.

ABOUT RISC-V. RISC-V International. Available from: https://riscv.org/about/. Visited on: 19 Dec. 2022.

ROHAN, Quentin. **Retour d'expérience sur BASILES**. EYESAT-NT-152-170-CNES. Centre national d'études spatiales. Feb. 2016.

REACTION Wheels. Sinclair Interplanetary. 2022. Available from: https://www.rocketlabusa.com/space-systems/satellite-components/reaction-wheels/. Visited on: 27 Nov. 2022.

NANOSSOC Sun Sensor for Nano-Satellites. SOLARMEMS. 2022. Available from: https://www.solar-mems.com/nanossoc/. Visited on: 27 Nov. 2022.

DYNAMICS of Attitude - Interface Control Document. BASIL-PHY_MD-ICD-0002-SPB. Spacebel. June 2018.

DYNAMICS of Attitude - Software Design Document. BASIL-PHY_MD-SDD-0002-SPB. Spacebel. June 2018.

HELP to create a SMP2 simulator for the BASILES infrastructure. BASIL-MU-SG_MO-2458-SPB. Spacebel. Dec. 2014.

ORBIT and Environment - Interface Control Document.
BASIL-PHY_MD-ICD-0001-SPB. Spacebel. Mar. 2018.

ORBIT and Environment - Software Design Document.
BASIL-PHY_MD-SDD-0001-SPB. Spacebel. Mar. 2018.

USER Manual - SMP Service. BASIL-MU-SG_AI_NS-1296-SPB. Spacebel. Nov. 2020.

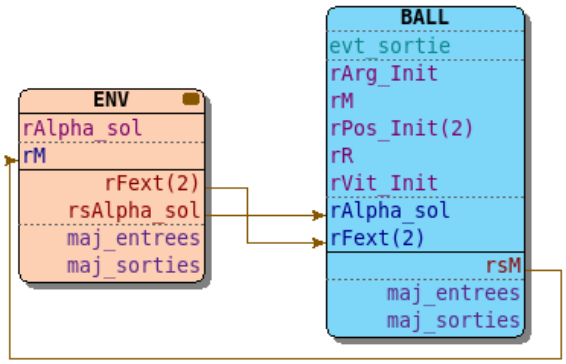
LANGUAGE. Tcl Developer Xchange. Available from:
`http://www.tcl-lang.org/about/language.html`. Visited on: 10 Dec. 2021.

Appendix

APPENDIX A – THE BOUNCING BALL SIMULATOR

This simulator is very simplified since it does not take into account atmospheric drag nor any kind of friction. It contains only to models: "BALL", which simulates the ball itself and its parameters, and "ENV", which is the surface the ball is bouncing on. Figure 39 shows both models running in a BASILES simulation.

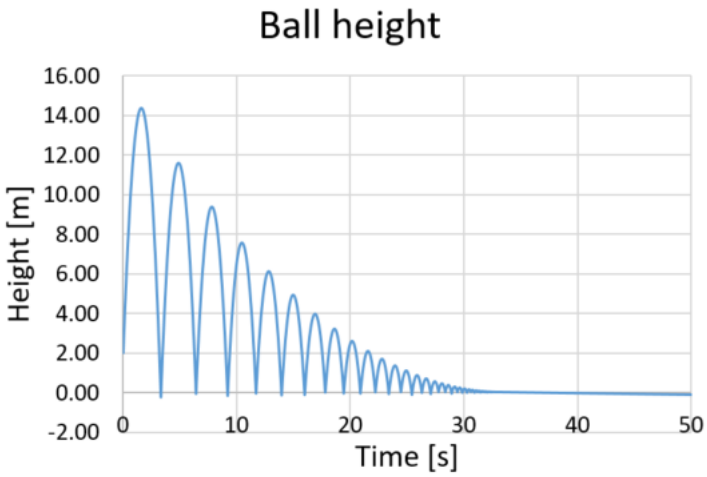
Figure 39 – The models of the "bouncing ball" simulator, BASILES 4.0.2.



Source: Author, 2021.

Figure 40 and Figure 41 illustrate the bouncing ball height and traveled distance results respectively.

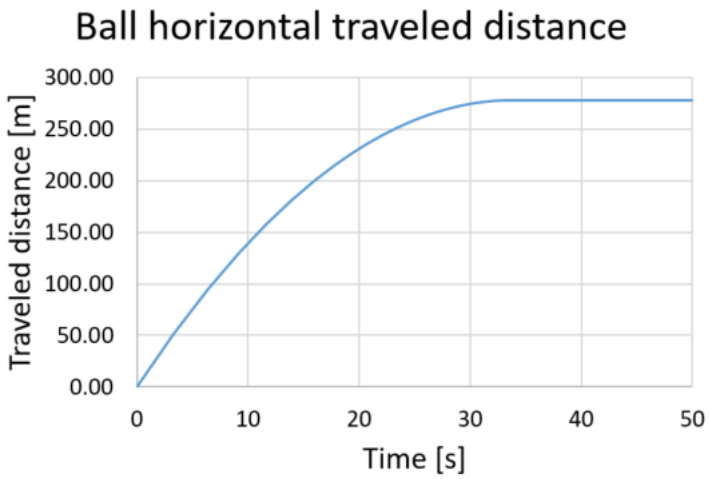
Figure 40 – Bouncing ball height.



Source: Author, 2021.

These results were obtained with the parameters and initial conditions presented in Table 7 and Table 8.

Figure 41 – Bouncing ball traveled distance.



Source: Author, 2021.

Table 7 – Bouncing ball parameters.

Instance	Parameter	Value
BALL	Mass	0.06 kg
	Radius	0.03 m
ENV	Absorption coefficient	0.9

Source: Author.

Table 8 – Bouncing ball initial conditions.

Instance	Variable	Value
BALL	Angle to the ground	0.785398 rad (45°)
	Velocity	22 m/s
	X position	0 m
	Y position	2 m

Source: Author.

Following Rohan's tutorial to create this simulator is a great way to start learning BASILES's main tools, HMI and simulation basics. This simulator makes use of the BASILES model standard, which is different from the SMP standard, but knowing how the BASILES standard works also makes learning SMP easier later on, as they both share some similarities.

APPENDIX B – SEMANTIC VERSIONING FOR AEROSIM

All versions of AEROSIM, as well as all versions of the models developed by the author, have been tracked following a semantic versioning scheme in which new versions receive a tag in the vMAJOR.MINOR.PATCH format. Each simulator and each model starts at v0.1.0.

Simulator versioning

Changes to the MAJOR, MINOR and PATCH numbers are subject to the following criteria:

- **MAJOR**: addition or removal of model instances.
- **MINOR**: change of model parameters, change of simulation starting time and duration.
- **PATCH**: correction of bugs; modification, addition or deletion of observations (such as variables to observe and synoptics).

Model versioning

Since simulators and models are inherently different, changes to the latter follow different criteria:

- **MAJOR**: changes that make a model incompatible with simulators using its previous version.
- **MINOR**: significant changes to a model that do not make it incompatible with simulators using its previous version.
- **PATCH**: small changes to a model, such as the correction of bugs.

Let it be noted that whether or not a modification requires a change to the version number can be a subjective question. It is up to the developer to decide what is best to do, preferably maintaining the same criteria throughout the development of the project. In any case, what is important is to keep track of the changes, so that the project itself is better understood by the future developers who will work on it and any error that appears can be reversed in an easy way.