GLAC-14-S7B,9x20442

CONCURRENT SYSTEMS ENGINEERING OF A CUBESAT SYSTEM.

Antonio Cassiano Júlio Filho
National Institute for Space Research (INPE), Brazil, cassiano.filho@inpe.br

Fernando Putarov Vieira[1], Flávio de Azevedo Corrêa Júnior[2], Luis Carlos Catarino[3], Geilson Loureiro[4], Mário Henrique Trentim[5], Wagner Schalch Mendes[6]

[1]Atech Technology and Business, Brazil, fernando.putarov@gmail.com
[2]Aeronautics and Space Institute (IAE), Brazil, flaviocj@gmail.com
[3] Federal Institute of Education, Science and Technology of São Paulo (IFSP), Brazil, lcatarino_br@yahoo.com,
[4]National Institute for Space Research (INPE), Brazil, geilson@lit.inpe.br
[5]Aeronautics and Space Institute (IAE), Brazil, mario.trentim@gmail.com
[6]Alstom Brazil Energy and Transport, Brazil, wagner.schalch@gmail.com

Abstract: The paper presents a method combining traditional systems engineering with the concepts of the concurrent systems engineering to develop a CubeSat system in a case study. The method consists of Mission Definition, Lifecycle Processes Analysis, Stakeholder Analysis, Stakeholder Requirements, Functional Analysis, Architecture Design and Detailed Design. The approach was exemplified by modeling requirements, functions and implementation elements, simultaneously, for the CubeSat product and for the organizations that implement the CubeSat life cycle processes. With the increase of complexity of systems, traditional systems engineering methods are failing to anticipate all requirements needed to secure a solution that satisfies the stakeholders' needs. The concurrent systems engineering is a multidisciplinary and collaborative approach to derive, evolve and verify a system solution, composed of products and the organization implementing the products' life cycle processes, balanced throughout the system's life cycle to satisfy stakeholders' needs and get public acceptance and the products and the organization composing the system solution for the CubeSat system will be developed concurrently. Picosatellites class, by definition, are extremely small, lightweight satellites, the progenitor of the pico class is the CubeSat, an open source satellite architecture for space research that usually has a volume of exactly one liter and typically uses commercial off-the-shelf electronics components. Those miniaturized satellites have as core components: an antenna, a radio transmitter for up-linking commands and down-linking data, an onboard computer, an electric power system, and payloads. The CubeSat specification accomplishes several high-level goals. Simplification of the satellite's infrastructure makes it possible to design and produce a workable satellite at low cost

I. INTRODUCTION

According to INCOSE (2006) [1], a system is a construct or collection of different elements that together produce results not obtainable by the elements alone. The results include system level qualities, properties, characteristics, functions, behavior and performance. The value added by the system as a whole, beyond that contributed independently by the parts, is primarily created by the relationship among the parts; that is, how they are interconnected. Buede (2009) [2] defines a system as being a collection of hardware, software, people, facilities, and procedures organized to accomplish some common objectives. Stevens *et al.* (1998) [3] defines a system as being the solution that correctly responds to the users' needs, providing a full operational capability (e.g: set of products, support processes, training material) integrated into a working environment.

A system can be seen as a set of elements that interact among each other to achieve a greater purpose, not achievable by the elements individually. Moreover, a system solution can be defined as being the products plus the organization that implements the products' life cycle processes proposed to fulfill the stakeholders' needs.

As the time passes by, the complexity, or in other words the variety and interactions of the system's elements, is getting higher and higher. Systems are not seen anymore as stand-alone entities, but rather as part of a bigger system that includes other systems as well human beings. We could say that a system is at the same time a subsystem of another system and the super system of other systems. If engineers fail to organize the problem statement and the system solution proposed for the problem in a structured and systematic way, the system will certainly fail to satisfy the stakeholders' needs.

In this paper, a methodology based on systems engineering will be exercised taking as a case study the CubeSat commissioning phase. Concurrently, the product and the organization implementing the life cycle processes will be developed.

II. WHAT IS SYSTEMS ENGENEERING?

According to INCOSE (2004) [4], Systems Engineering (SE) is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem: Operations, Performance, Test, Manufacturing, Cost & Schedule, Training & Support and Disposal. SE integrates all the disciplines and specialty groups into a team effort forming a structured development process that proceeds from concept to production to operation. SE considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the stakeholders' needs.

Nevertheless, SE is not a specific discipline. As stated by Friedenthal *et al*. (2012) [5] as being a multidisciplinary approach to develop balanced system solutions in response to diverse stakeholder needs. SE includes the application of both management and technical processes to achieve this balance and mitigate risks that can impact the success of the project.

Regarding the committed life-cycle cost against time, Weilkiens (2007) [6] states that SE concentrate on the definition and documentation of system requirements in the early development phase, on the preparation of a system design, and on the verification of the system as to compliance with the requirements, taking the overall problem into account: operation, time, test, creation, cost and planning, training and support, and disposal. According this definition, we can conclude that the hardest decisions to questions are those made in the early stages of a project, so the most important work happens there in a phase where decision changes bring not so severe impacts into the overall project.

Taking the perspective of the solution, Stevens *et al.* (1998) [3] define SE as the creation of effective solutions to problems, and managing the technical complexity of the resulting developments. The emphasis begins with requirements definition and the product to be built and goes through to the integration and verification before delivering the system to the customer.

Systems engineering is, therefore, a multidisciplinary and collaborative approach to derive, evolve (or develop) and verify a system solution balanced throughout the system's life cycle to satisfy stakeholders' needs and get public acceptance.

Unfortunately, just doing SE is not enough. According to Holt and Perry (2008) [7], projects fail and disasters happen due to different reasons. However, there are three underlying reasons why things go wrong, which are known as the 'three evils' of systems engineering: complexity, lack of understanding and communication issues. That means that if we don't have a systematic approach to do Systems Engineering, the probability of those evils becoming a reality increases.

III. CONCURRENT SYSTEMS ENGINEEERING PROCESS, METHOD AND FRAMEWORK

One can find several processes for SE, all trying to address the problem in an efficient way and make sure nothing is left behind. One important characteristic of a SE process is being parallel and iterative, but for didactic and expository purposes, they are usually presented in a sequential manner. The SE process at any company should be documented, measurable, stable, of low variability, used the same way by all, adaptive, and tailorable. There cannot be a unique process suitable for every single application, but all of them will certainly have similarities at the end.

It is a common sense among systems engineers that the system solution for a given problem is not only made of products, but also the organization implementing the life cycle processes of the products themselves. Meaning, we cannot simply develop the products and take the system as being completed for granted. We need also to develop the responsible product organization.

In order to study the complexity in product development Hitchins (2000) [8] proposed a framework including three dimensions factors: variety, connectivity and disorder. Stated on this very proposal, Loureiro (1999) [9] proposed to this framework the analysis dimension with the four analysis sub-processes (stakeholders, requirements, functional and implementation) addresses the variety complexity factor. The integration dimension with the product and organizational elements to be integrated addresses the connectivity complexity factor. The structure dimension with the system breakdown structure hierarchy addresses the disorder complexity factor.

The concurrent systems engineering approach must be applied at each layer of the system breakdown structure. Loureiro (1999) [9] proposed a method called Concurrent Structured Analysis that is used in this approach. The figure 1 presents the steps by which the analysis sub-processes are performed concurrently for product and organization (Loureiro, 2010) [10].

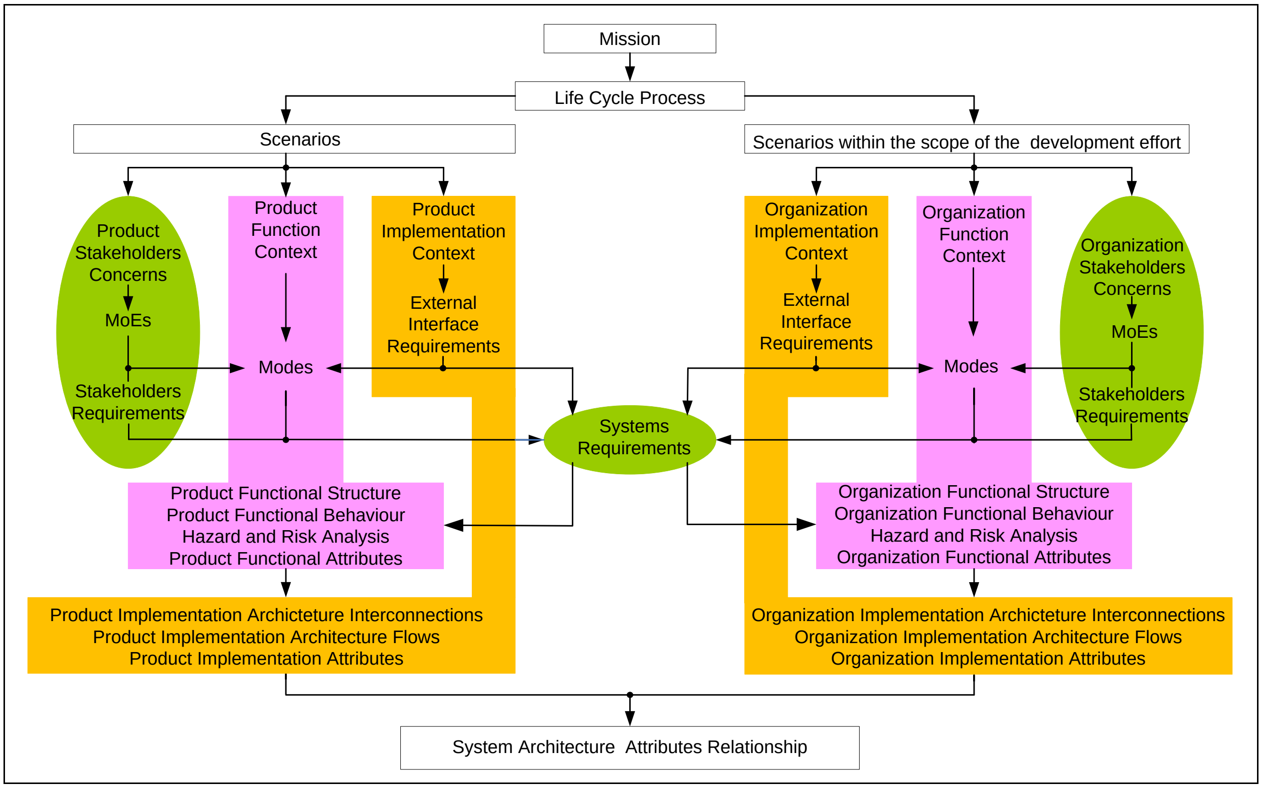


Figure 1: Steps of the concurrent systems engineering process.

IV. CASE STUDY: CUBESAT SYSTEM

Picosatellites, by definition, are extremely small, lightweight satellites. The progenitor of the pico class is the CubeSat, an open source satellite architecture for space research that usually has a volume of exactly one liter (10 cm cube) – stated as one unit (1U) –, has a mass of no more than 1.33 kilograms, and typically uses commercial off-the-shelf electronics components. Those miniaturized satellites have as core components: an antenna, a radio transmitter for up-linking commands and down-linking data, an onboard computer, an electric power system, and sensors.

The CubeSat specification accomplishes several high-level goals. Simplification of the satellite's infrastructure makes it possible to design and produce a workable satellite at low cost. Encapsulation of the launcher-payload interface takes away the prohibitive amount of managerial work that would previously be required for mating a piggyback satellite with its launcher. Unification among payloads and launchers enables quick exchanges of payloads and utilization of launch opportunities on short notice.

V. MISSION DEFINITION

This part of the method tries to understand the real problem to be solved, in terms of what the stakeholders are expecting from the system, and how they will give acceptance of the system developed. Hereafter, we will go through each step of the process within this part of the method, using the CubeSat system as a base.

V.I. Statement of the Need

The statement of need have to be defined in a short phrase: *“The system shall be able to measure ionospheric plasma properties”.* Some constraints have already been defined by stakeholders which sponsors the mission. Among them, it is assumed that the mission will be short lived, have low cost, and will involve high risks. These preliminary constraints define the boundary of feasible solutions for the system, becoming a mandatory requirement from the stakeholder that provides funds to the project.

V.II. Initial Stakeholders

Some initial stakeholders involved have been listed below: Scientists interested in ionospheric plasma phenomena; Telecom equipments and services providers as end users; Brazilian Space Agency (AEB) as mission sponsor;

As interactive and non linear process, the systems engineering could identify more preliminary stakeholders, after performing the first cycle of analysis. The interactive process repeats for each level of analysis. The next steps of mission definition have focused on stakeholders "Telecom equipments and services providers" only.

V.III. Measures of Effectiveness and Qualification Strategy

The table 1 presents a Measures of Effectiveness (MoE) and Qualification Strategy.

Concern	MoE	Qualification Strategy
Persistence	Frequency of data collecting on the coverage area	Data analysis of Mission
Prediction based on models	Time period of data collecting (minimum 1 year)	Data analysis of Mission
Coverage	Coverage area taking Brazilian territory	Data analysis of Mission
HF Frequencies	Register the affected frequencies	Data analysis of Mission

Table 1: MoE and Qualification Strategy

V.IV. Operational Scenarios

In this step, were identified the current operational scenario and, after decision-making alternatives, the future mission´s operational scenario:

Figure 2 presents a "As-Is" and "To Be" operational scenarios".

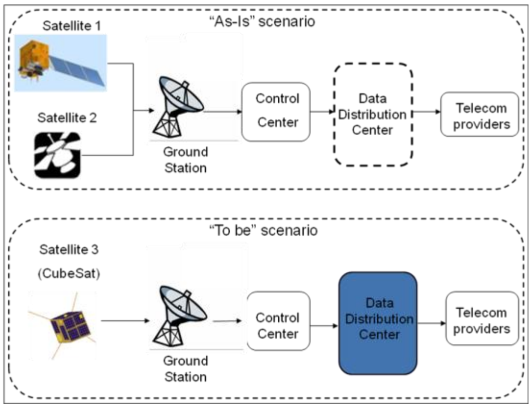


Figure 2: As-Is" and "To Be" operational scenarios.

A new Data Distribution Center, not currently available, identified in As-is/To-be scenarios, will perform the operational steps shown in the figure 3 below.

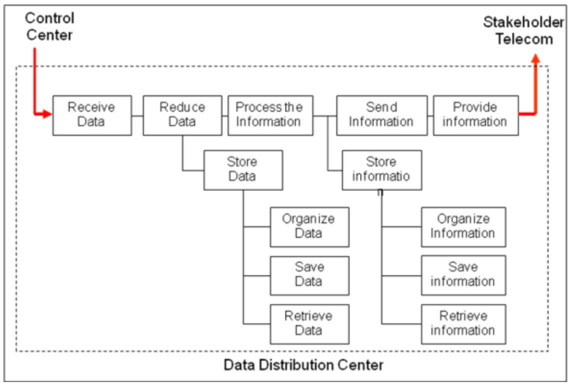


Figure 3: Steps of operational scenarios.

V.V. Stakeholder´s Capabilities and Constraints.

The table 2 presents a list of stakeholder capabilities and table 3 presents a list the stakeholder constraints.

Stakeholder Capabilities:
Access the data directly by internet
Receive a diary report by e-mail
Access historical data

Table 2: Stakeholder´s Capabilities

Stakeholder Constraints:
Cover the equatorial area over Brazilian territory
Minimum operation time: 1 year
Disposal process comply the space standards

Table 3: Stakeholder´s Constraints

V.VI. Concepts of Operations and System Operational Architecture.

This step in the system Engineering process describes the way the system works from the operator's perspective. A CubeSat was defined as the space segment of the mission and the new Data Distribution Center was included in the Concepts of Operations (CONOPS) landscape, shows in figure 4.

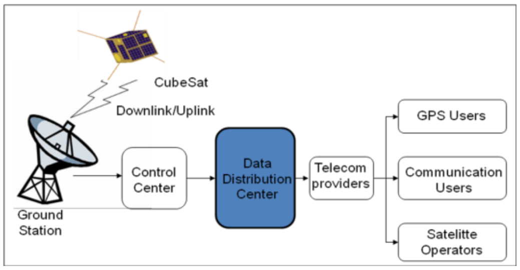


Figure 4: Concept of operations.

According to Larson et al (2009) [11], the system's operational architecture consists of:

- Its major elements,
- Their positioning within the reference universe, and,
- Tracings of key operational scenarios between the system, its elements, and the context.

The table 4 presents the System Operational Architecture.

System Segment	Functions
Space segment: CubeSat	Operates payload to collect data; Generates and send telemetry to Control Center (downlink); Receives and processes telecommand from Control Center (uplink)
Ground segment: Ground station and Control Center	Receives , analyses and storages telemetry from CubeSat; Generates and sends telecommand to CubeSat; Provides payload data to Data Distribution Center
User segment-1: Data Distribution Center	Receives payload data from Control Center; Stores payload data and information; Processes payload data; Provides information to different users;
User segment-2: End user	Accesses or receives information of interest

Table 4: System Operational Architecture

VI. LIFE CYCLE PROCESSES ANALYSIS

The purpose of this part is to identify completely the environment where the products and the organization composing the system will be inserted to. Hereafter, we will go through each step of the process within this part of the method, using the CubeSat system as a base.

VI.I System-of-Interest's Life Cycle Processes

The system-of-interest is divided in a timeline with different life cycle processes, as shown below:

- Development;
- Launching campaign;
- Operation;
- Disposal;

Each lifecycle process takes additional stakeholders with new concerns and new requirements, performing the system engineering process from a higher level to a lower level.

VI.II Life Cycle Process Scenarios

To simplify, the authors has approached two scenarios of life cycle processes, selected among previous life cycle processes, one focusing on product and the other focusing on organization. According to the system engineering method presented in this paper, all lifecycle processes for the product and all its scenarios of life cycle process must be taken in account along the analysis. From the organization perspective, not all of them will be included because some of them are not within of the scope of development effort of the system.

The table 5 presents a list of Lifecycle processes for organization and table 6 presents a list of the Lifecycle processes for product.

Organization Perspective
Life cycle process: Development
Lifecycle process scenario: Functional tests
Description: The Development Organization tests an engineering model of the CubeSat

Table 5: Lifecycle processes for organization

Product Perspective
Life cycle process: Operation
Life cycle process scenario: Operational tests
Description: The CubeSat is tested for commissioning

Table 6: Lifecycle processes for product

VII. STAKEHOLDERS ANALYSIS

The objective is to identify additional stakeholders, together with their concerns, and how they will give acceptance of the final system. Hereafter, we will go through each step of the process within this part of the method, using the CubeSat system as a base. The stakeholder analysis has been performed by using the Integrated Definition Language (IDEF0) tool (Buede, 2009) [2]. The figure 5 shows Stakeholders analysis for organization and figure 6 shows Stakeholders analysis for product.

The table 7 presents Stakeholders concerns and Moe´s from the Development Organization perspective.

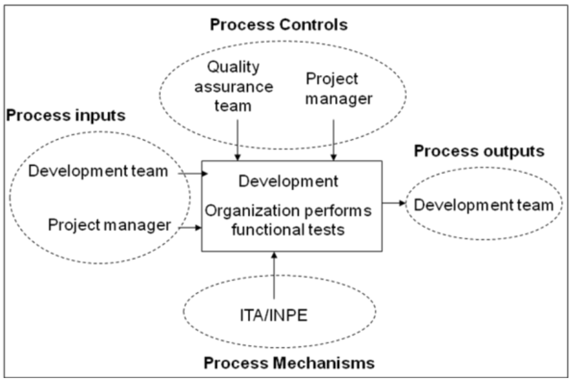


Figure 5: Stakeholders analysis for organization

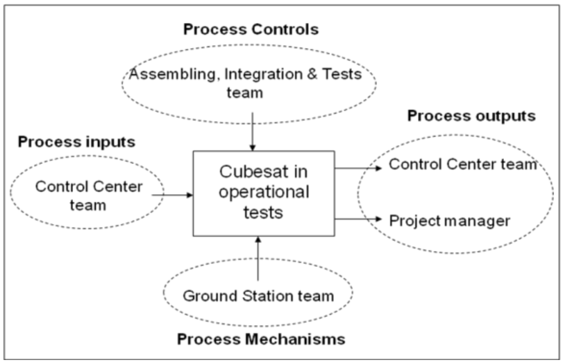


Figure 6: Stakeholders analysis for product.

Concern	Stakeholder	MoE
Engineering model verification	Development team	Compliance to the specifications
	Quality assurance team	Compliance to the procedures and standards
	Project Manager	Costs and schedule according to project baseline
	ITA/INPE	Suitable facilities and personnel to perform the tests

Table 7: Stakeholders concerns from the Organization

The Engineering model of the CubeSat must be tested, complying all standards, procedures and specifications. The Development Organization must perform all functional tests accordingly. The Tests team and facilities must be ready to perform them. The table 8 presents the list of the Stakeholders concerns and Moe`s from the CubeSat perspective.

Concern	Stakeholder	MoE
Operational verification	Project manager	One CubeSat payload is operating at least
	Control center team and Ground Station Team	Execute correctly all telemetry and telecommand instructions
	AIT team	The CubeSat bus shall be totally operational

Table 8: Stakeholders concerns and Moe`s from the CubeSat perspective

The mission is characterized by low costs and high risks. Thus, if one of CubeSat´s payloads is running well at least, the mission is considered successful. The same thinking is applied to the CubeSat´s bus, but in this case, all bus subsystems are critical and they must be run accordingly.

VII.I. STAKEHOLDERS REQUIREMENTS

In this part, the system specification begins. Starting from the stakeholder requirements we derive the system requirements with all the formalism intrinsic to them. Hereafter, we will go through each step of the process within this part of the method, using the CubeSat system as a base. The steps of SE processes capture stakeholder requirements. Based on previous stakeholder analysis, some requirements defined from them are shown in the table 9.

Req. ID	Description	Concern	Stakeholder	1	2	3
REQ.01	The Control Center team shall be able to perform operational tests in a period no longer than 10 minutes.	Operational verification	Control Center team	O	M	D
REQ.02	The Development Team shall be able to check a particular module of engineering model independently of any other module	Engineering model verification	Development team	P	M	T

Table 9: Stakeholders Requirements
Legend:
(1) P: product, O: Organization, (2) MoSCoW It is an acronym that means: Must, Should, Could and Wish
(3) T-Test, I-Inspection, D-Demonstration, A-Analysis)

VII.II Stakeholder Requirements Documents

The first version of detailed document, containing all information related to stakeholder requirements defines the baseline to transform stakeholder requirements (problem domain) in system requirements (solution domain). This document has additional attributes to maintain, manage and track the stakeholders requirements. The stakeholder requirements document has been structured according the topics shown below:

1. General Description
Introduces the system and its context
    1.1 General capabilities
    (Describes what capabilities are required and why they are needed).
    1.2 General constraints
    (Describes what constraints apply and why they exist).
2. Stakeholders
(Describes the stakeholders and their interests).
    2.1 Stakeholders description
    2.2 Stakeholders concerns
3. Operational environment
(Cooperating or competing systems and their interfaces with the product).
    3.1 External systems
    3.2 Interfaces with external systems
4. Specific requirements
    4.1 Scenarios capabilities
    (The scenario)
    4.2 Scenarios constraints
    (The quality demanded by users)

VIII. FUNCTIONAL ANALYSIS

The idea behind this part of the method is to create a functional model that will answer to the stakeholder requirements. Hereafter, we will go through each step of the process within this part of the method, using the CubeSat system as a base.

VIII.I Product and Organization Functional Contexts

For each scenario of each life cycle process identified previously, we need to identify the environment where the product and the organization are inserted into. We call that context and it means seeing the product and organization as a black box with the external entities that interact with them. Having that, we are able to identify the flows of material, energy and/or information exchanged between the product and organization and its environment (i.e. external entities). The figure 7 shows the CubeSat functional context within the 'operational tests' scenario identified for the 'operation' life cycle process. We will use the method developed by Hatley and Pirbhai (1988) [12] to model the functional context.

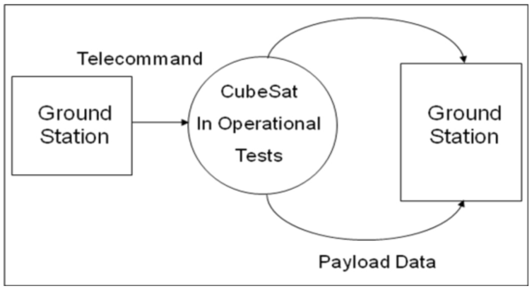


Figure 7: Functional context of the CubeSat in operational tests.

VIII.II Circumstances in Contexts

As stated before, the context is the environment where the product or organization is inserted into. The external entities may present different states during the scenario of the context being studied (e.g. the ground station may be able to communicate with the satellite or not, an external organization may be available or not). A circumstance is the set of states the external entities are at a given moment in the scenario. For each circumstance identified, the product and the organization may need different modes of operation to answer to that. The table 10 shows the Circumstances and functional tests organization modes.

Circumstance	Mode
Development team available	Active
Development team unavailable	Inactive

Table 10: Circumstances and functional tests organization modes

VIII.III Mode Analysis

After identifying the modes, we need to determine the dynamics behind them. It means, we need to perform a mode transition analysis in order to model when the product and/or organization will change from one mode to the other. The figure 8 shows the mode transition diagram for the functional tests organization.

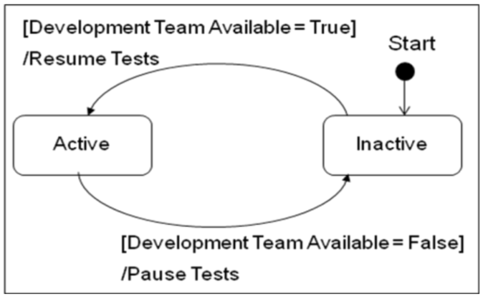


Figure 8: Mode transition diagram for the functional tests organization.

VIII.IV Essential Functional Architecture.

For each mode identified previously, we identify the essential functions provided by the product and organization when in that mode. When we say 'essential functions' we mean functions that are not related to any specific technology or implementation. They are the very essence of the system. Independently from the technology or implementation that will be used to develop it, those functions must be present. In this step, we are only concerned with the structural aspect of the functions, meaning, the functions themselves and the flows of material, energy and/or information exchanged among them. No behavioral information will be given. The figure 9 shows the Data Flow Diagram 0 (DFD0) for the CubeSat in active mode and the figure 10 shows the Data Flow Control 0 (DFC0) for the CubeSat in active mode.

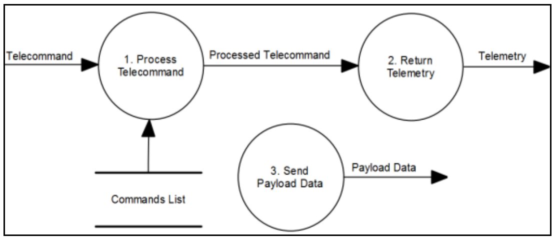


Figure 9: DFD0 for the CubeSat in active mode.

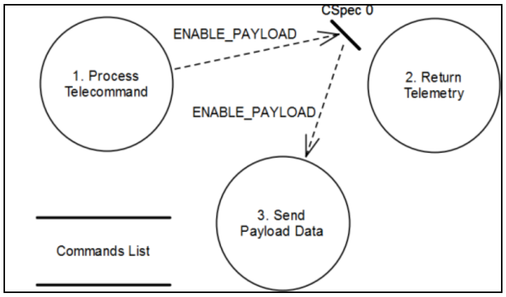


Figure 10: DFC0 for the CubeSat in active mode.

VIII.V Functional Decomposition.

For each essential function identified previously and still using the DFDs and DFCs diagrams, we begin to decompose it into lower level functions, but never forgetting to keep aside technological and implementation aspects. We keep going into lower functional levels until we feel comfortable to write down what the function must do in a simple piece of structured text (Hatley and Pirbhai, 1988) [12].

VIII.VI Physical Context.

After defining the ideal world (i.e. essential functional architecture), we must starting entering into the real world, meaning, the world constrained by tangible elements. In the same way we did for the functional contexts (both for product and organization), we should identify the physical contexts of the product and organization. The idea behind that is very similar to the first one, but instead of identify the flows of material, energy and/or information, we must identify the physical interface that will be used to exchange the flows already identified in the functional contexts. The figure 11 shows the Physical context of the CubeSat in operational tests.

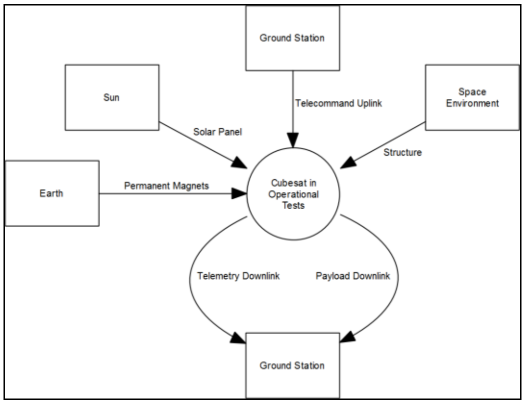


Figure 11: Physical context of the CubeSat in operational tests.

VIII.VII Enhanced Functional Architecture

Having the physical context in hands, we can now identify non-essential functions that must be present to support the physical external interfaces identified. Taking as a base the Hatley and Pirbhai (1988) [12] method, shows in figure 12, those functions can be classified in: input processing functions, output processing functions, user interface processing functions and enabling functions. The first two types denote functions that will transform the physical flows into the logical flows (input processing functions) and the other way around (output processing functions). The third type means functions responsible to convert the user information into the logical flows identified in the essential model. The fourth type represents functions interfacing with other products or organizations that make possible our product and organization to work properly in the mode of the scenario of the life cycle process we are studying (e.g. external test device, maintenance organization).

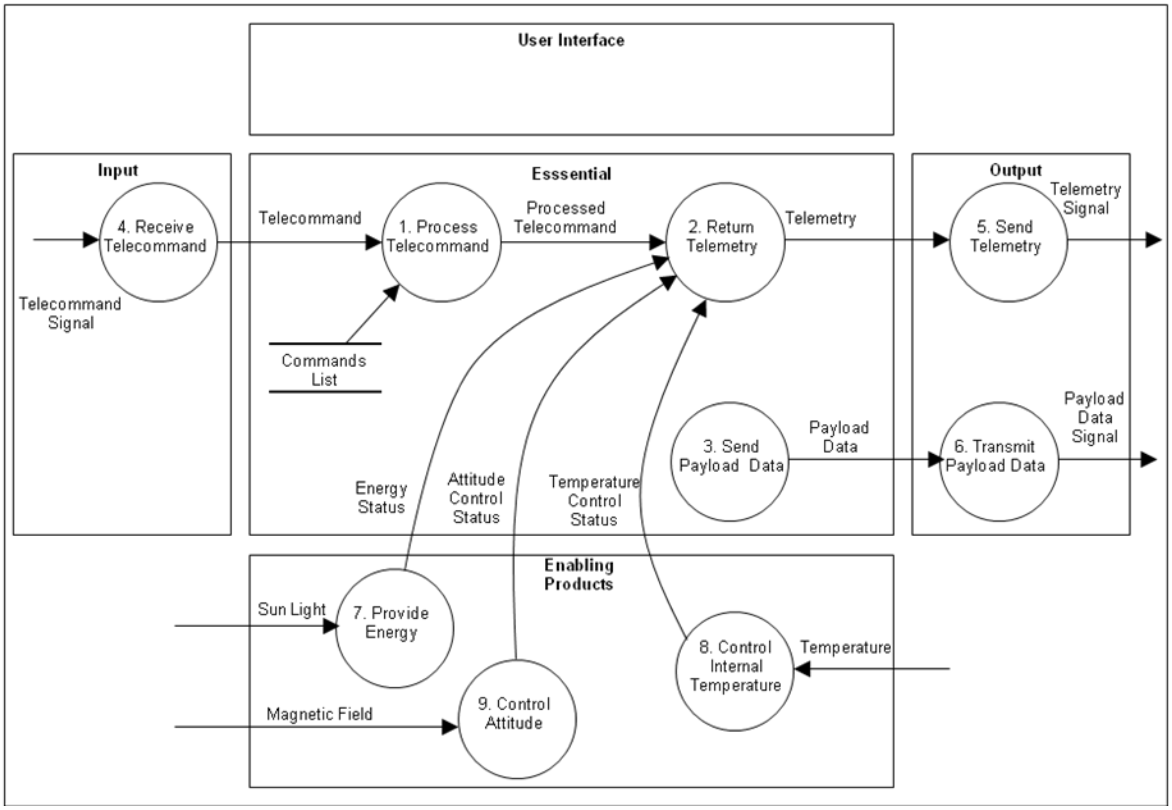


Figure 12: Enhanced functional architecture for the CubeSat in active mode

VIII.VIII FMECA

In order to perform a systematic approach, we usually start analyzing the system in the normal cases, and when that is finished we need to enter in the exceptional cases (failures). In this step, we perform the Failure Mode, Effects and Criticality Analysis (FMECA). Here, we try to identify possible causes for failures and provide a way to mitigate them by adding functions to our system. The failure may have three different sources for causes: circumstance (i.e. originating due to the state of an external entity), passport (i.e. originating from an interface) or non-function (i.e. originating from a system function not executing). The functions to be added to the system may be of the following types: preventive (avoid the failure), corrective (put the system back into the state previous to the failure), protective (protect the system against the failure) and/or detection (detect the failure before it happens).

VIII.IX Behavioral Analysis.

Until now, we have defined the structural aspects of the system functions. We need also to define their behavioral aspects. By behavioral aspects we mean when each function will be activated and/or deactivated, as well in which sequence the functions will be executed. According to the Hatley and Pirbhai (1988) [12] method, we create a state transition diagram showing the states and how the product/organization goes from one state to the other. The events that trigger the state transitions are the ones identified in the DFCs. The actions associated with the transitions are used in a process-activation-table to indicate which processes from the DFDs are active or inactive and in which state.

The figure 13 shows the CubeSat in operational tests state transition diagram. We can also use this table 11 to show the sequence in which the process are activated or deactivated.

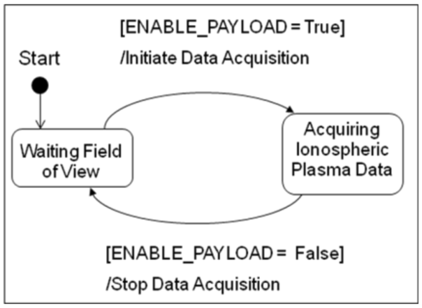


Figure 13: CubeSat in operational tests state transition diagram.

Process → Action ↓	1. Process Telecommand	2. Return Telemetry	3. Send Payload Data
Initiate Data Acquisition	X	X	X
Stop Data Acquisition	X	X	

Table 11: Process-activation-table for the CubeSat in operational test

IX. ARCHITECTURE DESIGN

At this stage, the physical elements of the system will be defined and mapped to the functions already defined. Interfaces between physical elements will be identified and specified. Hereafter, we will go through each step of the process within this part of the method, using the CubeSat system as a base.

IX.I Generic Physical Architecture.

At this point, we start defining the physical elements that will compose the system (product and organization) and how they will interface to achieve the system's requirements defined so far. Those elements are still not the real elements, but generic entities, such as, "on-board computer", "heater", "battery", "documentation", "planning". Hatley and Pirbhai (1988) [12] propose the use of two main diagrams to depict this architecture: Architecture Flow Diagram and Architecture Interconnect Diagram. The first one is very similar to a DFD used for functional analysis, but instead of having bubbles representing functions (or processes) we use rounded rectangles representing the generic physical elements and the arrows represents the physical flows between them. The second diagram we reuse the same physical elements and draw solid lines representing the physical interface between them. Those are the interfaces that will be detailed in the interface control document (see below).

The figure 14 shows the Product breakdown structure, the figure 15 shows the Product architecture flow diagram and figure 16 shows the Product architecture interconnect diagram.

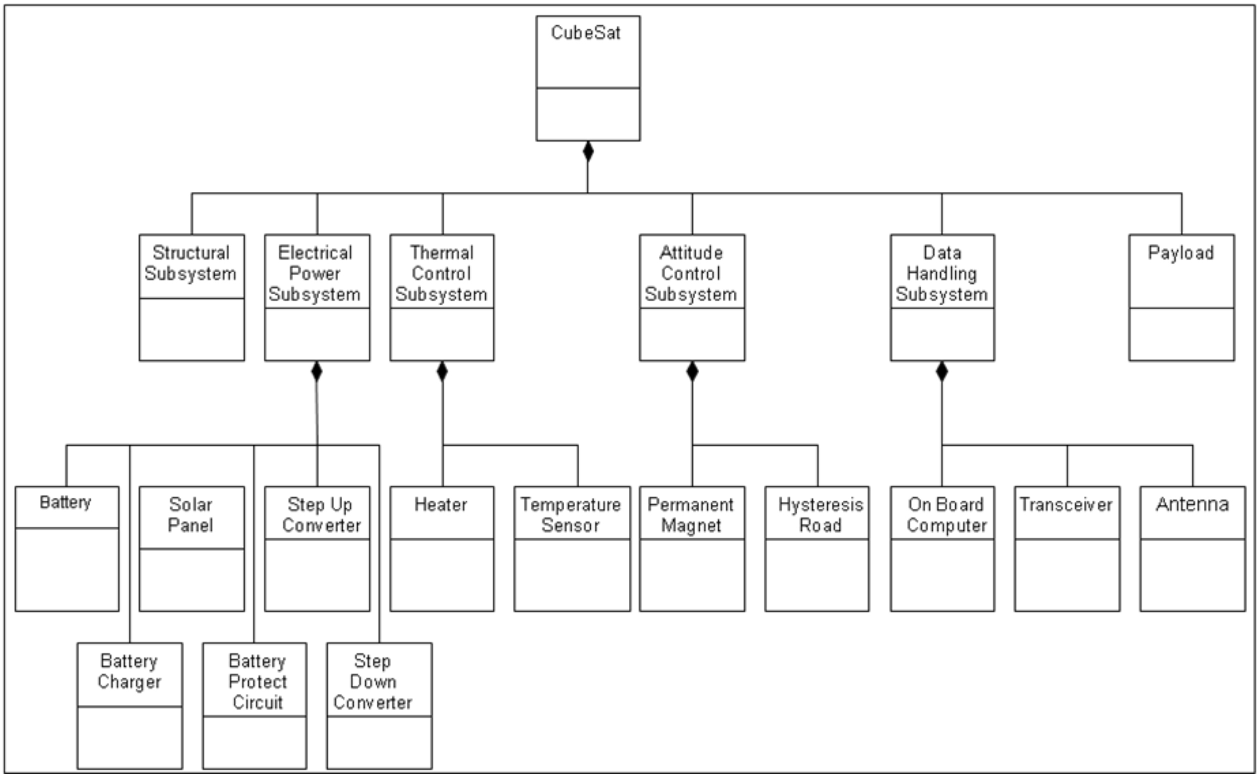


Figure 14: Product breakdown structure

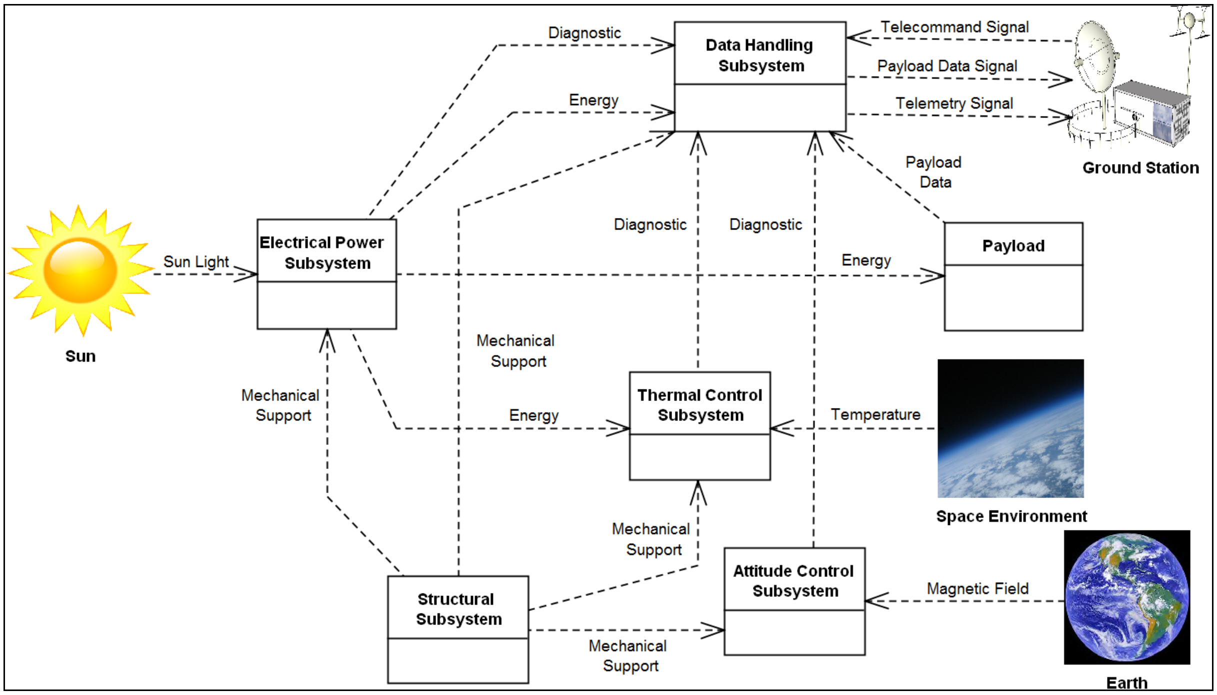


Figure 15: Product architecture flow diagram.

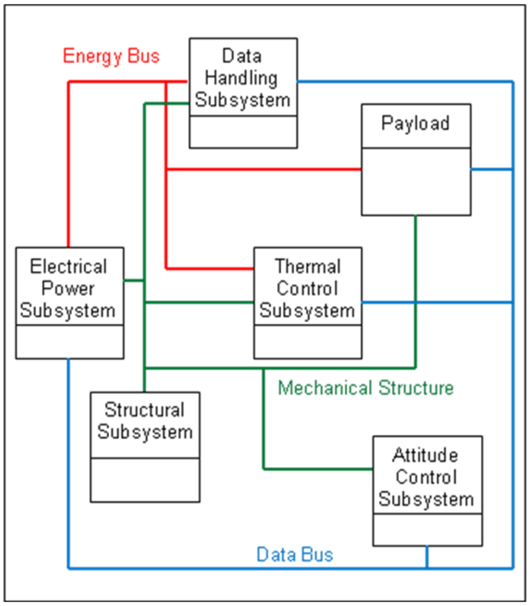


Figure 16: Product architecture interconnect diagram.

IX.II Allocatable Functional Architecture.

Physical world constrains functional world. At the end, the functional model we have developed so far may not be compatible with the physical architecture we came up with. Sometimes, it is necessary to iterate between functional world and physical world until we have a perfect match.

IX.III Requirements Allocation

Stakeholder needs have been translated into stakeholder requirements that have been derived to system requirements. The solution must, therefore, satisfy the system requirements identified. We must, then, allocate all system requirements to either product or organization elements.

IX.IV Morphological Chart

We have several alternatives for the generic elements identified in the generic physical architecture. The pertinent alternatives shall be raised and organized in a morphological chart.

IX.V Decision Analysis

Based on the MoEs raised previously and/or on the stakeholder requirements identified, we need to decide on which alternative of each physical element to use. A table can be elaborated to justify the decision.

IX.VI Functional Allocation

Each function identified in the allocatable functional architecture must be allocated to one single physical element and one physical element must always allocate functions with high level of cohesion. This allocation can be done with the help of an allocation matrix.

IX.VII Interface Control Document

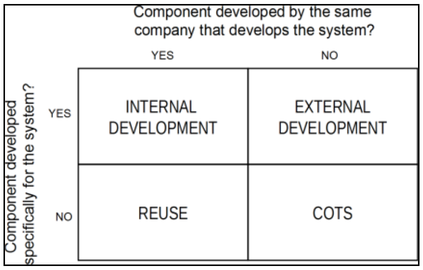
Interfaces are the most critical part in any system development and must be, therefore, thoroughly specified. For each interconnection identified in the physical architecture, we must specify what are the requirements (or functions) involved and what flows on it. With the help of an Interface Control Document (ICD) we can provide the information necessary for the correct development of the interface.

X. DETAILED DESIGN

Each physical element of the system must be carefully specified and decisions should be made either to buy or develop those elements. Hereafter, we will go through each step of the process within this part of the method, using the CubeSat system as a base.

X.I Make or Buy Decision

One decision we have to make about physical components is if they will be internally developed, externally developed, reused or bought from an external supplier. To help this decision we can use the Make or Buy matrix shown in figure 17.



		Component developed by the same company that develops the system?	
		YES	NO
Component developed specifically for the system?	YES	INTERNAL DEVELOPMENT	EXTERNAL DEVELOPMENT
	NO	REUSE	COTS

Figure 17:Make or Buy decision Matrix.

X.II Component Specification.

Independently if the component will be a new development or bought from an external supplier, we need to go deep into its details, but never forgetting the black-box aspect. For each component identified, a specification must be created with the necessary information to support development or outsourcing.

XI. CONCLUSIONS

The paper presented a method for the concurrent systems engineering approach applied to the development of a CubeSat system. The approach was exemplified by modeling requirements, functions and implementation elements, simultaneously, for the CubeSat product and for the organizations that implement the CubeSat life cycle processes (only those within the scope of the CubeSat development effort).

XII. ACKNOWLEDGMENTS

We thank the Brazilian National Institute for Space Research (INPE, acronym in Portuguese) by their support and the Brazilian National Council for Scientific and Technological Development (CNPq, acronym in Portuguese) by their financial support.

XIII. REFERENCES

[1] INCOSE. 2006. “A Consensus of the INCOSE Fellows”. Last Updated: 02 Oct 2006. http://www.incose.org/practice/fellowsconsensus.aspx.

[2] Buede, D.M. 2009. *The Engineering Design of Systems: Models and Methods*. 2nd Ed. John Wiley & Sons Inc.

[3] Stevens, R., Brook, P., Jackson, K., and Arnold, S. 1998. *System Engineering: Coping with Complexity*. Prentice Hall Europe.

[4] INCOSE. 2004. “What is Systems Engineering?” Last Updated: 14 Jun 2004. http://www.incose.org/practice/whatissystemseng.aspx.

[5] Friedenthal, S., Moore, A., and Steiner, R. 2012. *A Practical Guide to SysML: The Systems Modeling Language*. 2nd Ed. Elsevier/Morgan Kaufmann.

[6] Weilkiens, T. 2007. *Systems engineering with SysML/UML: Modeling, Analysis, Design*. Morgan Kaufmann.

[7] Holt, J., and Perry, S. 2008. *SysML for Systems Engineering*. The Institution of Engineering and Technology.

[8] Hitchins D. 2000. Getting to Grips with Complexity. Last Updated: Feb 2005. http://www.hitchins.net/ASE_Book.html.

[9] Loureiro, G. 1999. “*A Systems Engineering and Concurrent Engineering Framework for the Integrated Development of Complex Products”*. PhD thesis, Loughborough University (Leicestershire, UK).

[10] Loureiro, G. 2010. “Lessons learned from 12 years of space systems concurrent engineering”. In Proceedings of the 61st International Astronautical Congress, Prague, Czech Republic, 27 Sept to 1 Oct 2010, IAC-10-D1.5.2., IAF, Paris.

[11] Larson, W., D. Kirkpatrick, J. J. Sellers, D. Thomas, and D. Verma. 2009. *Applied Space Systems Engineering*. Boston, MA (US): McGraw Hill.

[12] Hatley, Derek J. and Pirbhai, Imtiaz A. 1988. Strategies for Real-Time System Specification, Dorset House Publishing.

XII. BIOGRAPHY

Antonio Cassiano Júlio Filho is graduated in Analysis and Development of Systems at FAETEC, Brazil. Has 29 years of the work at Brazilian National Institute for Space Research (INPE), in the Space Engineering and Technology in the area Systems Development of Ground Division in the design, development, test and integration of the ground segment necessary for the tracking and control of satellite and processing to permit the dissemination of environmental data collected by satellites. He is taking a MSc course on Space Engineering and Technology at INPE in São José dos Campos, Brazil.

Fernando Putarov Vieira is gratuated in Computer Sciences at UMESP (Universidade Metodista de São Paulo). He has been working in the last 10 years on the development of computational systems. Fernando works since 2004 at Atech with software and systems for Air Defense, Air Traffic Control, Simulation and Systems Integration. He is taking a MSc course on Space Systems Engineering at INPE (Instituto Nacional de Pesquisas Espaciais) in São José dos Campos, Brazil.

Flávio de Azevedo Corrêa Junior was graduated in Mechanical Engineering (1984) and Engineering Production (1985) at *Faculdade de Engenharia Industrial*, Brazil. From 1986 to 1999, he worked as a design engineer for launching vehicles and sounding rockets at Institute of Aeronautics and Space (IAE), Brazil. Since 2000, he was taking part of the IAE's Systems Engineering team. He worked as the IAE's liaison for the Microgravity Programme of Brazilian Space Agency, being a member of the technical team for the first Brazilian astronaut mission to the International Space Station (2006). Nowadays, he is attending a master course with emphasis on Space Systems Management at National Institute for Space Research, Brazil.

Geilson Loureiro is the Founder and first president of INCOSE Brasil since March 2012. Professor of Systems Engineering at ITA (Aeronautical Technological Institute) and INPE (Brazilian Institute for Space Research) at graduate and undergraduate levels, since 2006. Post doctorate internship at Wurzburg University, Germany (2011) and at MIT (Massachusetts Institute of Technology), USA (2004 - 2005) taking parts in university satellite projects and space exploration system of systems, respectively. PhD on systems engineering from Loughborough University (1999). Technologist at INPE since 1988, working at the Laboratory of Integration and Testing (LIT), having dealt with many satellite life cycle processes. Geilson has around 100 publications including 2 books and has 3 PhD completions and around 15 Master completions to his name from 2006 up to 2012.

Luis Carlos Catarino was graduated in Electrical Engineering at USP (Universidade de São Paulo) in Brazil and took his MBA at FGV (Fundação Getúlio Vargas) in Brazil. In the past 15 years, he has been working with implementation of ERP systems in Pharmaceutical, Telecom, IT and Energy companies. Since 2009, he has been working as independent consultant in the space area for a specialist company in designing and developing attitude control systems. He is taking a MSc course at INPE (Instituto Nacional de Pesquisas Espaciais) in São José dos Campos, Brazil.

Mário Henrique Trentim is graduated in Electronics Engineering at ITA (Instituto Tecnológico de Aeronáutica), MBA in Finance at CBS (CEDEPE Business School) and Executive Certificate in Project Management at University of La Verne. He has been working for the last 10 years managing projects and programs in Aeronautics, Space, Telecom and IT. Mario works since 2011 as PMO manager at the Instituto de Aeronáutica e Espaço. He is taking a MSc course in Engineering at Instituto Tecnológico de Aeronáutica.

Wagner Schalch Mendes was graduated in Electrical Engineering at FEI (Faculdade de Engenharia Industrial) in Brazil and took his MSc course at Unicamp (Universidade Estadual de Campinas) also in Brazil with emphasis in Modeling and Control of Discrete Event Systems. Nowadays he is a specialist engineer at Alstom Transport and has been working with software and system modeling using UML and SysML for the past ten years. In 2009 he became a global Alstom trainer for the internal systems engineering program. In 2011, he was nominated senior expert at Alstom Transport in development and engineering processes. He is now taking a PhD course on Systems Engineering at INPE (Instituto Nacional de Pesquisas Espaciais) in São José dos Campos, Brazil.