A Decision-Making Methodology for Lean Satellite Assembly-Integration-Testing Programs Management

By

Pauline Faure

Kyushu Institute of Technology
Graduate School of Engineering
Department of Mechanical and Control Engineering

A dissertation submitted to Kyushu Institute of Technology in accordance with the requirements of the degree of Doctor of Engineering in the Faculty of Engineering

August 2017

ABSTRACT

Ideas of lean satellite programs are still at an early stage of development and data on their actual management, development, testing strategy, and/or achieved outcomes are scarce if not non-existent.

For a part of the space community, lean satellite programs represent the possibility of building a satellite despite lack of resources; and hope, through successful achievement, to be recognized as a positive contributor to space science, technology, engineering, and applications. For another part, lean satellite programs represent the black sheep of satellites engineering in the sense that they have the highest rate of infant mortality and mission failure, and a non-functional satellite is another space debris.

High rate of infant mortality and mission failure of lean satellites tend to point out that ground based testing is not carried out optimally or sufficiently. Currently, work to define an adapted testing strategy for lean satellites is undergoing through the establishment of ISO19683 and HORYU-IV testing strategy was established based on it.

Experimental data show that most failures are discovered during the early stage of the assembly, integration, and testing (AIT) processes. The results also show the importance of interfaces verifications. HORYU-IV failures taxonomy is also presented.

Simulations were also carried out. Simulations results show that drastically increasing the initial AIT processes time does not drastically increase system's reliability, but does drastically increase cost by more than 80% for HORYU-IV case.

The research outcomes are intended to be used as guidance by lean satellite program developers and managers for better planning of AIT processes and resources allocation in order to develop more reliable, and therefore more sustainable, lean satellite systems.

ACKNOWLEDGMENTS

First and foremost, I would like to express my sincerest gratitude to Professor AKAHOSHI and Professor CHO, who, through their impeccable sense of leadership, patiently and kindly supported me in my research and helped me broaden my knowledge and skills in hypervelocity impact engineering, systems engineering, space environment, management, diplomacy, and more. Then, I would like to thank Professor YONEMOTO and Dr. KITAZAWA for their precious time and interest in this research.

I would also like to express my heartfelt appreciation to the members of the Computational Mechanics Laboratory and to the members of the Laboratory of Spacecraft Environment Interaction Engineering, whose insightful questions and comments during research seminars helped improving the work hereafter presented. Special thank you also goes to Ms. Kumiko SHIRAKAWA, Ms. Sayo TSUKINARI, Mr. Seichi KAWANO, and Mr. George MAEDA, whose smiles, joyful mood, and kindness always tremendously help during harder times. My heartfelt appreciation also goes to all HORYU-IV members with a special attention to Atomu TANAKA and Nori AIT-MOHAMMED, who helped me collect experimental data.

Finally, my greatest indebtedness goes to my mother, Chantal BONNET, my father, Jean-Luc FAURE, my brother, Maxime FAURE, my grand-mother, Lucienne GIRARDI, and my great grand-mother, Emilienne FAURE, for supporting me even 10,000km away, and for helping and showing me the path to grow as a strong, independent, and capable woman.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

NOMENCLATURE

Abbreviations

A	
AD	Analog-to-digital
AIT	Assembly, integration, and testing
AODS	Attitude and orbit determination sub-system
AVC	Arc vision camera
B	
BBM	Bread board model
Big apple	DLP, PEC, and VAT integrated on the same PCB
C	
CAD	Computer assisted design
CAM	Earth photography camera
COM	Communication
CW	Continuous waves
D	
DCDC	Direct current to direct current
DLP	Double Langmuir probe
E	
EM	Engineering model
EPS	Electrical power supply
F	
FBC	Faster, better, cheaper
FM	Flight model
G	
GS	Ground station

H	
HK	Housekeeping
HVSA	High voltage solar array
I	
IAA	International Academy of Astronautics
ISO	International Organization for Standardization
L	
LEO	Low earth orbit
M	
Mother board	OBC, EPS, and COM integrated on the same PCB
MPU	Micro-processor unit
N	
N/A	Non applicable
NASA	National Aeronautics and Space Administration
O	
OBC	On-board computer
OBO	On-board oscilloscope
P	
PCB	Printed circuit board
PEC	Photoelectrons current measurement
PI^2	Prevent infant mortality, identify culprit sub-systems, improve testing strategy
R	
RBD	Reliability block diagram
S	
SNG	Digi-singer

<u>T</u>	
TBF	Time before failures
TX	Transmitter
<u>V</u>	
VAT	Vacuum arc thruster

Symbols

α	Growth rate
β	Shape parameter
ΔTr	Time delay for random failure mode
ΔTs	Time delay for single-AIT failure mode
λ	Scale parameter
θr	Random failure
C(t)	Cumulative failure rate at time t
N(t)	Cumulative number of failures at time t
r(t)	Failure rate at time t
R(T)	Reliability at time T

Subscripts

0 (zero)	For parameters associated to cross-AIT failure mode
i	For parameter associated to single-AIT failure mode
A	For level A failure criticality
B1	For level B1 failure criticality
B2	For level B2 failure criticality
B3	For level B3 failure criticality
B4	For level B4 failure criticality

CHAPTER 1 – Introduction

From a search on ScienceDirect, which gathers over 14 million publications from more than 3,800 journals and 35,000 books [1], it appears that only 67 results are found when searching for “small satellite and reliability” key words. From these, only 9 results are actually relevant in the sense they deal with the overall satellite project reliability and not just one of the sub-systems reliability, for instance. Similarly, when searching for “small satellite and cost” key words, 112 results can be found, but only 5 are actually relevant. Finally, when searching for “small satellite and schedule” key words, 49 results can be found from which only 8 are actually relevant.

Paradoxically, “small satellites” are being developed since the beginning of the space era with the launch of the “small satellite” Sputnik I in 1957, and Wertz [2] estimated that *“historically, about 20% of all satellites launched are less than 400kg”*. Yet, information on “small satellites” programs including cost, schedule, and reliability is extremely scarce.

Since nearly a decade, the demand for “small satellites” is growing [3]; however, it was shown in [4] that their infant mortality rate is the highest among all satellite categories considered. This means that they are most likely to become non-operational soon after their insertion in orbit and consequently become space debris. This is not desirable for a sustainable use of space. Moreover, a high rate of infant mortality points out that testing prior to launch is insufficient. The reader shall note that in this research, “testing” is equally used to refer to “assembly, integration, and testing (AIT) processes”. The lack of testing of “small satellites” prior to launch can either be linked to lack of funds, time, knowledge, or development philosophy, and it all impacts the reliability of the satellite after launch.

Another problem to be addressed is regarding the terminology for the definition of non-traditional satellites. Throughout the world, a large portion of the space community uses “small satellites” to refer to non-traditional satellites. This terminology therefore limits non-traditional satellites to a certain mass, size, and/or volume. Yet, depending on the entities, countries, or even decades considered, different definitions of the values for the mass, size, and/or volume can be found. There is therefore no standard definition on what the mass, size, and/or volume of these so-called “small satellites” should be. Moreover, defining non-traditional satellites by their mass, size, and/or volume is reductive of their capabilities. The terminology “lean satellite” is thus introduced to refer to non-traditional satellites, i.e., satellites developed in a fast and low cost manner regardless of their mass, size, and/or volume.

From the problems addressed above, it is clear that there is a need for more studies addressing non-traditional satellite programs, namely “lean satellite programs”, taking into account how these programs’ schedule, cost, and reliability can be affected depending on the selected AIT strategy prior to launch. This dissertation is the first study on this topic and it aims at guiding current and future “lean satellite” developers to help them optimizing their decisions for the improvement of their programs’ reliability and/or cost and/or schedule depending on their resources, requirements, and desired objectives.

It should be noted that throughout this dissertation, the terminology “lean satellite” is always used to refer to non-traditional satellite programs except when referring to other studies, in which case the terminology as adopted in the considered study is used.

For the achievement of the aforementioned aim, the research was separated into a two-step action plan. The first step consisted in using HORYU-IV project to gather, during AIT phases, actual data on the number of failures, occurrence of failures, and type of failures. In this research, failure is defined as any mistake, malfunction, anomaly, or glitch in the hardware or software, such as misplaced hole, power line inversion, component short-circuit, and others. The data were collected for about one year and from them, the relation between the cumulative number of failures and the cumulative AIT processes time were studied as well as the time between failures, system reliability evolution, and failures taxonomy. The second step consisted in carrying out simulations based on reliability engineering as applied to satellite systems. From the simulations, failures mode and failures criticality could be simulated and their effect on total AIT schedule, project overall schedule, and satellite reliability after launch were studied. Finally, based on HORYU-IV financial data and simulations results, the evolution of cost over the different AIT phases and over the overall project schedule was studied.

From this two-step action plan, there are three main objectives the research attempts to achieve.

1) Demonstrate lean satellites reliability improves through AIT processes, even under harsh program limitations.
2) Study the relations between AIT processes time, reliability, project schedule, and cost.
3) Serve as a guidance to future lean satellite programs developers and managers.

For the report of the research, the dissertation was separated into six chapters. In this first chapter, the problems to be addressed, the overall research goal, and the research methodology were introduced. In the second chapter, background and literature relevant to the research are detailed. In the third chapter, the methodology for the experimental data collection and for the simulations is described. In the fourth chapter, results and discussion on collected experimental data and simulations are presented. In the fifth chapter, the research outlook, including the current research limitations and countermeasures as well as a practical application of the outcomes of this research, is outlined. In the sixth and final chapter, the research conclusions and recommendations are stated.

CHAPTER 2 – Background and Literature

In this chapter, the “lean satellite” approach as a paradigm shift from the “small satellite” approach is explained. Moreover, the different notions of cost, schedule, and reliability as applied to satellite systems engineering is discussed and the importance of their consideration for lean satellite programs is described.

2.1. From “small satellite” to “lean satellite” - The necessity of a standardized terminology

Though “small satellites” are present since the beginning of the space era in 1957, a standardized definition was never established of what the mass, size, and/or volume a “small satellite” should be [5]. With the development of the “small satellites” market and the desire to miniaturize satellites as much as possible, many different sub-categories of “small satellites” have emerged since the last 20 years. Among them are “micro-, nano-, pico-, femto-satellites and CubeSats”, whose number is exponentially increasing since 2012 [3]. Yet, from this wide range of “small satellites” designation, only CubeSats’ design is standardized [6], whereas other categories’ definition depends on the country, developing entity, or even decade considered [3, 7-9].

At first sight, it does not seem that having different mass definitions for the same satellite category are troublesome. Let us take a step back and consider two case scenarios. In the first scenario, consider an international collaboration on the development of at least one “nano-satellite” for remote sensing applications. During the conceptual design phase of the project, entity A and entity B thought and designed separately a “nano-satellite” to present to each other after a few months. Mid-point to the conceptual design phase, entities A and B meet and present their own vision of the "nano-satellite". For entity A, "nano-satellite" meant a limitation of 10kg and accordingly to this mass constraint, chose a low resolution and narrow spectrum imager easily adaptable for a 10kg satellite. For entity B, however, "nano-satellite" meant a limitation of 50kg and accordingly chose high resolution and wide spectrum imager, and even add a propulsion system for de-orbit purpose. From this misunderstanding of what a "nano-satellite" mass is, entity A might stop its collaboration with entity B due to lack of funding for a 50kg-class "nano-satellite". Even if the international collaboration is pursued, time, money, and energy was lost during the few months the conceptual design phase lasted.

One might argue that this scenario is unlikely or even absurd, but let us remember that when one thinks a certain criteria is obvious and thus implicitly understood by everyone, one tends to forget to verify that this criteria is actually obvious and similar for everyone. Two famous, but unfortunate, examples of very promising systems that failed due to that "obvious criteria" factor are Mars Climate Orbiter in which it was obvious the international unit system should prevail [10], and more recently ASTRO-H for which it was obvious to input the right sign to limit the satellite rotational speed [11]. These examples resulted in a loss of approximately USD125 million [12] for Mars Climate Orbiter and USD360 million [13] for ASTRO-H. This does not include the cost of the scientific discoveries and advances that could have been made if the systems had been executed properly. Though for the purpose of illustration the Mars Climate Orbiter and ASTRO-H failures were simplified, I understand that one "simple" mistake is unlikely to result in the complete failure of such complex space systems. An accumulation of "simple" failures and processes mistakes, however, can; and preventing “simple” failures starts by clearly defining the obvious.

For the second scenario, consider a private entity that wants to develop a “small satellite” and is looking for investments from either the government or private investors with little or no knowledge on the technicalities of the space sector. There is usually not much time to convince someone your project is valuable and thus, worth being funded. Unfortunately the terminology “small satellite” tends to connote “small capabilities”, “small objectives”, and/or “small results” when considering limitations on power generation, data rates, and instruments resolution [14]. This infamous image makes it even more difficult to convince investors that “small satellites” can actually help and benefit the society and is worth the investment.

From these scenarios, it clearly appears that the “small satellite” terminology is not obvious. It is a confusing and negatively connoted terminology. It could therefore lead to great misunderstanding with possible consequences of having troubles finding funding or troubles concretizing international and/or domestic collaborations and partnerships.

The question remains: if a terminology based on satellite’s mass, size and/or volume is not appropriate, what is the terminology that could better describe and encompass the goals to be achieved by non-traditional satellites?

For answering this question, it is necessary to understand the essence of non-traditional satellites. Non-traditional satellites objective is not to be “small” either in size or capabilities. Their essence is to adopt a development strategy different from traditional satellites to allow satellites’ development to be fast and at low cost as compared to traditional satellites' schedule and cost.

A definition of non-traditional satellites based on the development philosophy rather than on a mass, size and/or volume is not a novel idea. In 1994, Fleeter [15] already pointed out that issue. Yet, it is only in 2014 that a group of international experts proposed to adopt the "lean satellite" terminology to designate non-traditional satellites developed in a fast and low cost manner [16]. As of October 2016, the work is still being undergoing and the group is actively working through the International Organization for Standardization (ISO) and the International Academy of Astronautics (IAA) to establish official report and possibly standards defining the requirements for a satellite to be considered "lean".

"Lean" is not a new word, but its utilization in combination with satellite engineering is novel. The "lean" concept was first introduced through the Toyota production system and the associated book from Ohno [17]. Though in this book the Toyota production system is not qualified using the actual "lean" terminology, Ohno's work at Toyota was later described as "lean manufacturing" by Womack *et al.* [18]. The "lean" concept was born. Later, in addition to "lean manufacturing", concepts of "lean enterprise", "lean systems engineering", "lean thinking" and even "lean aerospace" emerged [19-22].

The general concept of leanness is to deliver to the customer the desired product in the fastest manner and at the lowest cost possible. To achieve this, there are four core values to adopt.

1) **Minimize waste** – Waste has many forms. It could be waiting time, overproduction, defective products, and more. As countermeasures, just-in-time, no inventory, production flow, and quality control approaches should be adopted.
2) **Produce valuable product** – This mostly means to deliver to the customer the

desired product. In other word, the production of a product should be driven by the customer’s demand not by the company desire to push a certain product on the market.

3) **Respect employees –** Employees’ talent should not be wasted at watching a machine to make sure it is working properly. Instead, employees should be trained to be flexible and multi-tasks capable workers. This way, their work actually adds value to the product by limiting defective pieces and wasting time for examples. In the meantime, the machine can monitor itself for malfunctions. As main users of the production plant, they should also be asked their opinion on how the production plant can be improved.
4) **Improve continuously –** Even though waste, delivery time, and costs have been reduced by 95% from 2015 to 2016, for example, it is not a reason to stop the improvement and maintain the same way of producing from 2016 to 2017. There is always a process that can be improved, made faster and less costly. It is important not to rest on the accomplishments made, but looking forward to continuously improving them from one point in time to another.

The “lean” concept emerged in the automobile industry and therefore some parts are difficultly transferrable to the satellite industry. An example of limitation is the number of units to be produced in a year. In the automobile industry, a same car model can be produced at thousands of thousands copies. In the current satellite industry, however, it is unlikely that a same satellite will be produced a thousand times even for mega-constellation such as One Web [23] or Planet Labs [24]. Another limitation is the environment the two products should undergo. Automobiles should be resistant to rain, snow, dirt, and be safe; but satellites should survive in space, in a vacuum, without the possibility to be maintained or repaired during their on-orbit lifetime, while exposed to harsh radiations that can damage electronics, as well as extreme and fast temperature variations. These imply thorough and costly on-ground testing, and even more costly launch to put the satellite on-orbit. From these limitations, it is clear that "lean" can currently be applied to the satellite industry only until a certain extent.

One might argue that due to all these differences and consequent limitations, "lean" philosophy cannot be applied to satellites engineering. Let us remember two critical points. The first point is that "lean" is about delivering to a customer the desired product in the fastest manner and at the lowest costs possible. This is applicable to any industries, from the electronic chip maker to the cranes maker going through automobiles and satellites industries. Second, though the "lean" concept has emerged from the automobiles industry, this is not about comparing automobiles industry with satellites industry. This is about developing satellites in a new and forward philosophy to, again, ensure low cost and fast development as compared to traditional satellites.

From this perspective, the terminology "lean satellites" is particularly adapted and tackle appropriately the development philosophy adopted by the non-traditional satellites makers.

2.2. "Fast development" and "low cost" approach – The foundation for lean satellite programs

The precedent section highlighted the essence of lean satellite programs, i.e. fast development and low cost, and in this section the importance, feasibility, and limitations of such development philosophy is described.

Until the 1990s, stakeholders for satellites' design, manufacturing, testing, and operations were mainly governments through their respective space agencies. Satellites and spacecraft in general, were getting more complex integrating advanced payload for remote sensing, communication, or planetary exploration applications. However, 30 years after the success of the Apollo missions, in an instable political climate between the Cold War, the war in Vietnam, and the war in Iraq, the budget for space development was limited [25, 26] and it was no longer suitable to have space assets taking decades to develop with consequent cost overruns. As an innovative way to continue developing innovative systems with a constrained budget, the National Aeronautics and Space Administration (NASA), established the Faster, Better, Cheaper (FBC) policies in 1992 to *“decrease the amount of time and cost for each mission and to increase the number of missions and overall scientific results obtained on each mission”* [27].

NASA was not the only one to implement the FBC policies, its partners did too. This resulted in successful missions such as the planetary exploration mission NEAR in collaboration with the John Hopkins University Applied Physics Laboratory (JHU/APL) [28] or the ocean and climate forecasting mission JASON-1 in collaboration with the Centre National d’Etudes Spatiales (CNES) [29]. In total, 31 missions were developed by NASA and its partners over 16 years from 1992 to 2008 [30].

Thanks to the FBC philosophy initiated by NASA, the market of “small satellites” was opened to new comers, such as universities, that previously had little or no experience in practical satellite engineering. In the 2000s, these universities started to design, manufacture, test, and operate their own satellite as practical hands-on projects to educate tomorrow’s engineers to space engineering. This lean satellite revolution was further pushed forward by the establishment of the CubeSats standard by Twiggs and Puig-Suari [2]. Through the expansion of the lean satellite development philosophy, it became possible for one entity with restricted resources to fully develop several satellites, namely constellation, to achieve significant and meaningful scientific objectives. Successful examples from a mission and educational standpoints are the QB50 mission that aims at taking measurements in the lower thermosphere [31], UNISAT program that aims at testing terrestrial technology on-orbit [32], HORYU series that aims at studying on-orbit the principles of discharge generation on solar panels [33], and HODOYOSHI program that aims at observing the Earth with different types of instruments [34].

To sustain the enthusiasm of the space community toward lean satellite programs, different programs emerged to encourage and promote space utilization through lean satellite applications. Some examples are the Fly Your Satellite! program by the European Space Agency (ESA) that offers full support from development to test and launch of CubeSat [35], the Satellite Contest that aims at *"activating basic and applied research related to the space science and technology"* for high school to university students [36], and also the Mission Idea Contest that offers an international platform to promote outstanding ideas for space utilization from individuals or groups of space enthusiasts [37].

Since the 2010s, the market for lean satellites is booming and new private companies are entering into the lean satellite commerce [3, 38]. These space ventures aim at building satellites constellations to provide different services such as Earth observation, weather forecast, or telecommunications, and also spacecraft to achieve goals emerging from outside of the box thinking. One of the most interesting points of these ventures is that they succeeded in tackling the interest of investors with no prior involvement in the space sector opening the door to new intrepid space utilization that could have not been considered before due to their excessive “fantasy”, such as asteroid mining.

With nearly 25 years of experience and growth in adopting a lean philosophy in satellite engineering, not only the importance and feasibility of the approach was demonstrated, but the growing interest in such approach was also shown from parts with originally no involvement in the space sector. Moreover, a lean approach is not only significant for space agencies or private companies to develop satellites in a fast and low cost manner in order to increase the number of missions and resulting number of data. It is also significant to new comers such as universities, developing countries, and emerging space ventures to provide a new approach to the teaching of engineering through hands-on activities and to offer services to a larger part of the population in a new and disruptive manner.

There is however a main drawback to a philosophy solely based on fast development and low cost approach: the lack of quality of the end product. Though this problem was somewhat tackled in the NASA’s FBC policies under the “better” part, it has failed to be unequivocally demonstrated. The skeptical to NASA’s FBC policies even say to choose only two parameters out of the three. The skepticism mostly comes from two points: 1) under the FBC policies, NASA lost nearly 25% of all the missions developed [27]; 2) when NASA decided in adopting the FBC policies, no definition, implementation plan, or guidance using quantitative parameters were established, which lead to an absence of common understanding of the FBC policies [27, 39, 40].

In this research, reliability is used as a quantitative measure of how “better” a system can be compared to another and in the following section, the importance of reliability consideration along with cost and schedule for the success and sustainability of lean satellite programs is described.

2.3. “Reliability” and the “PI^2” action plan – The cement to sustainable lean satellite programs

As described in the previous section, fast development and low cost are the foundations of lean satellite programs. However, not taking into account reliability, i.e. the quantification of how well the system will perform on-orbit, would be a great mistake. In this section, the importance of reliability consideration along with schedule and cost is thus investigated.

Reliability is a mathematical function defined as the probability of non-failure. Specifically for space systems, it can be seen as the quantification of the system performances in space at a certain given time. A space system with a low reliability at a certain given time means that the system is most likely unable to perform as planned, i.e. is most likely to fail. On the other hand, a space system with a high reliability at a certain given time means that the system is most likely able to perform as planned, i.e. is most likely to succeed. Therefore, according to each lean satellite programs objectives, there is a sufficient reliability to be achieved at a certain given time on-orbit. How much this “sufficient reliability” should be is not to be defined by me. It should be defined by the stakeholders of a considered lean satellite program. Some might consider that a 60% reliability after 6 months on-orbit for a 1 year mission is “sufficient”, whereas others might consider that a 95% reliability is “sufficient”. Though it is not my intention to put a number on “sufficient reliability”, we can broadly define it as a reliability that will ensure the program minimum success criteria.

Without taking into consideration the reliability, lean satellites are launched for the solely purpose of launching a satellite. This was an important first step at the beginning of the space era or even at the beginning of the lean satellites revolution to demonstrate feasibility and capability, but nearly 60 and 25 year after the respective beginning of these eras, this is not true anymore. Launched lean satellites should have an added value, such as providing remote sensing data, telecommunications capabilities, and more for the benefit of society. Without this added value, lean satellites will turn quickly into useless man made space objects, namely space debris; and with the ever growing space debris problem [41] and consequent concerns rising in the space community [42, 43] this is not acceptable. More importantly, it should not be accepted in order to ensure a sustainable space environment and a sustainable lean approach to satellites engineering.

Unfortunately, from on-orbit surveys, Dubos [44] showed that “small satellites” present the highest rate of infant mortality among all satellites categories considered. In other words, “small satellites” have the highest likelihood to fail soon after their insertion on-orbit. More strikingly, when considering only the CubeSats category, the infant mortality rate rises by 1/3 as compared to the “small satellites” category [45]. This high rate of infant mortality points out that AIT processes of lean satellites prior to launch is insufficient. There is therefore a need to identify the critical AIT processes to be performed prior to launch to improve lean satellites reliability after launch.

In [46], it was shown that less than half of pico- and nano-satellites successfully inserted on-orbit actually achieves full mission success. From different on-orbit failure surveys [47-49], it appears that the two main drivers of satellites failure regardless of the category are: 1) solar array deployment/operations sub-systems account for about 25% of the failures and 2) communication sub-system accounts for about 20% of the failures. If we take a closer look at the CubeSats category, there are three main drivers of satellites failure [45]: 1) electrical power supply sub-system accounts for nearly 45% of the failures, 2) on-board computer system accounts for 20% of the failures, and 3) communication sub-system accounts for nearly 20% of the failures.

Despite the fact that data on lean satellites failures prior to launch is inexistent or unavailable and though the number of data on on-orbit satellites failures is scarce, it allows us to draft an action plan based on three pillars to ensure lean satellites sufficient reliability after launch: 1) prevent satellite infant mortality, 2) identify culprit sub-system(s) driving failures, and 3) improve AIT strategy prior to launch. This is what I decided to call the PI^2 action plan for Prevent, Identify, and Improve. The strong base of the PI^2 action plan relies on preventing infant mortality to ensure that the lean satellite will be reliable enough for the program minimum success criteria to be achieved. Based on that, culprit sub-systems driving lean satellites failures must be identified and the corresponding efficient and optimal AIT strategy must be improved accordingly to the lean program development. Thanks to this action plan, lean satellite programs will succeed by being driven not only by a low cost and fast development approach, but also by sufficient reliability insurance (Figure 2-1).

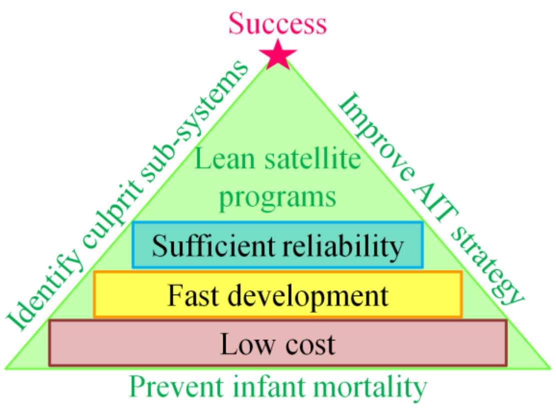


Figure 2-1. PI^2 action plan for successful and sustainable lean satellite programs

CHAPTER 3 – Methodology

The methodologies for the experimental data collection and for the simulations were described in [50, 51]. However, for the readers to be able to fully understand this research, the methodologies are also detailed in this chapter.

3.1. HORYU-IV experimental data

3.1.1. A brief overview of HORYU-IV project

HORYU-IV, also designated as the Arc Event Generator and Investigation Satellite, is a lean satellite developed within approximately two years at Kyushu Institute of Technology in Japan. HORYU-IV is a cubic satellite, whose envelope is about 45cm in all dimensions (Figure 3-1) for a total mass of about 10kg. The satellite mission statement is to acquire on-orbit data of discharge phenomena occurring on a high voltage solar array to deepen understanding of satellite charging, to contribute to the reliability improvement of current space systems, and to positively contribute to the realization of future high power space systems. To achieve these objectives, an oscilloscope and a camera system were developed in house as described in [52] and [53]. In total, HORYU-IV carries nine payloads, which along with the bus characteristics are described in [54]. The preliminary execution of its main mission and sub-missions were a success and the satellite is still operational as of August 2017.

Over its two years development, HORYU-IV underwent various assembly and integration phases as well as environmental, mechanical, and functional testing. During the AIT phases, the number of failures, the time at which failures occurred, and the sub-system(s) involved were recorded. Based on the acquired data, plots of the cumulative number of failures versus time could be obtained and the cumulative failure rate as well as the time between failures could be estimated. Data on failures occurrence prior to a satellite launch are inexistent or not available. Thanks to the data compiled and analyzed during HORYU-IV project, main culprit systems leading to failures depending on the AIT phase and malfunctions occurrence over the different project phases could be identified prior to launch. The details of the experimental data collection are presented in section 3.1.2.

With HORYU-IV data collection, it is intended to help future satellite developers and managers to identify, prior to launch, while the satellite is a repairable system, critical sub-systems, and project phases in order to decide where to concentrate resources (workmanship, financial, etc…) to develop a lean satellite, i.e. adopting a philosophy of fast delivery and low cost while achieving sufficient reliability.

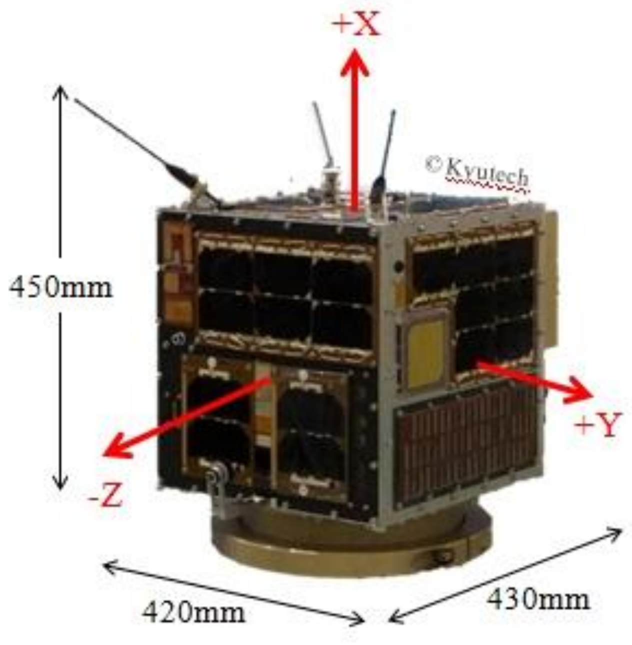


Figure 3-1. HORYU-IV external appearance, dimensions, and orientation

3.1.2. HORYU-IV experimental data collection methodology

HORYU-IV development was separated into three main phases: 1) bread board model (BBM), 2) engineering model (EM) itself separated into two sub-phases, EM1 and EM2, and 3) flight model (FM). These are respectively equivalent to phases B, C, and D as defined for NASA projects life cycle [2].

The number of failures, the time at which failures were discovered, and failures types were only recorded during the EM and FM phases. This corresponds to an approximate period of one year of experimental data collection.

The BBM lasted from October 2013 until the end of 2014. During this phase, satellite bus sub-systems were designed depending on payload requirements and the satellite structural and thermal models were established. During BBM, the payload requirements were constantly improved to ensure the payload functions are executed as planned. Hence, the sub-systems (payload or bus) design during BBM is constantly evolving and it was judged irrelevant to record data on failures that occurred during this phase. After sub-system requirements were decided and verified, the project moved to the EM.

During the EM, the different sub-systems' printed circuit boards (PCB) are manufactured. Then, testing of sub-systems as standalone and integrated sub-systems were performed. In addition to electrical performance verifications, the satellite as a whole also underwent thermal vacuum, vibrations, and shock testing. EM phase was separated in two sub-phases: EM1 and EM2, whose characteristics are explained in the last two paragraphs of this section. After the satellite soundness was verified, the project moved to its final phase, FM.

During FM, HORYU-IV was extensively tested as a whole to ensure its soundness after its launch. As for EM, electrical performances testing and environmental tests were carried out. For the latter, in addition to the aforementioned environmental tests, FM also underwent plasma testing. HORYU-IV‘s main mission is to study arcing phenomenon occurring on dedicated triple junction solar cells in low earth orbit (LEO). To achieve this objective, the satellite must be biased to a high negative voltage in comparison with the surrounding plasma environment and this is performed by attracting and collecting electrons contained in space plasma. Therefore, for HORYU-IV case, testing in simulated LEO environment, i.e. plasma test, was critical to ensure its main mission minimum success. The details of testing performed during EM and FM are provided in Table 3-1.

Table 3-1. Details of testing carried out during HORYU-IV EM and FM phases

		EM1	EM2	FM
Electrical testing	Electrical performance verifications test – standalone sub-system	○	○	○
	Electrical performance verifications test – integrated sub-systems	○	○	○
	End-to-end test	N/A	N/A	○
Launch environment testing	Vibrations test	N/A	○	○
	Shock test	N/A	○	○
Space environment testing	Thermal vacuum test	N/A	○	○
	Thermal cycle test	N/A	○	○
	Plasma test (LEO environment simulation)	N/A	○	○

HORYU-IV is constituted of seven bus sub-systems and nine payload sub-systems. For EM1, there were three main AIT phases: 1) the on-board computer (OBC), electrical power supply (EPS), and communication (COM) sub-systems were tested as standalone sub-systems; 2) OBC, EPS, and COM were integrated and tested.

The integrated board including OBC, EPS, and COM is hereafter designated as the mother board; 3) the S-band/Digi-singer (Sband-SNG), high voltage solar array (HVSA), double Langmuir probe/photoelectrons current measurement/vacuum arc thruster (DLP-PEC-VAT, hereafter designated as Big apple), and attitude and orbit determination/Earth photography camera (AODS-CAM) sub-systems were integrated one by one with the mother board and tested. During EM1, many modifications were made to the different sub-systems and new PCBs integrating the modifications were manufactured. The newly manufactured PCBs were tested as EM2. During EM2, AIT processes similar to EM1 were performed. Moreover, the on-board oscilloscope (OBO) and arc vision camera (AVC) sub-systems were integrated with the mother board and tested. Finally, launch and space environment testing were performed after all the sub-systems were integrated with the structure. The main verifications performed during electrical tests are described in Table 3-2.

Table 3-2. Main verifications performed during electrical performances testing

Sub-system		Tests as standalone sub-system		Tests as integrated sub-systems
Mother board	OBC	・MPU reset ・Satellite reset	・AD converters ・Share flash memory	・H8 reset ・Satellite reset ・AD converters ・HK saved to flash memory ・HK transmission ・Data decoding ・CW transmission interruption ・Command reception
	EPS	・DCDC converters ・S-band TX switch ・Satellite reset ・Separation switches	・Kill switches ・Voltage dividers ・Current sensors	
	COM	・CW transmission ・Command reception	・Data transmission	

Mother board + Sband-SNG, HVSA, Big apple, AODS-CAM, OBO, or AVC	-	-	・Reading of missions flash memory ・Writing to missions flash memory

During EM1, AIT processes were performed by the same personnel in series and the time at which failures occurred, as well as AIT processes start time, breaks, AIT processes resume time, and AIT processes end time were precisely recorded. However, for EM2 and FM, AIT processes were performed in parallel and various personnel recorded failures. Therefore, though failures occurrence and type was well documented, details on AIT processes break time, failures occurrence time, and AIT processes end time were overlooked due to their tedious character. As a result, for a better representation of the results over the different AIT phases, it was assumed for all AIT phases, i.e. EM1, EM2, and FM, that for a given AIT period, AIT processes were carried out from Monday through Friday, except if stated otherwise, for a period of 8 hours for all AIT processes but thermal vacuum test and safety verifications at Tsukuba Space Center. During thermal vacuum testing period, the testing period was 24 hours and during safety verifications at Tsukuba Space Center, the testing period was 2 hours per day. The details of the AIT processes period for all phases are presented in Table 3-3.

Table 3-3. HORYU-IV's AIT processes period description

Phase	AIT processes		AIT dates [YYYY/MM/DD]	AIT time per day [h]
EM1	Electrical performance verifications test – standalone sub-systems	OBC	2015/02/24-28	8
		EPS	2015/02/24-28	8
		COM	・2015/02/24-28 ・2015/04/13 ・201504/21	8
	Electrical performance verifications test – integrated sub-systems	Mother board	・2015/03/01-07, 09-13, 16-20, 23-27, 30-31 ・2015/04/01-03, 06-08	8
		Mother board + other sub-systems	・2015/04/09-10, 13-17, 19-24, 26-30 ・2015/05/01, 04-06	8
EM2	Electrical performance verifications test – integrated sub-systems		・2015/06/11, 29-30 ・2015/07/01-03, 06-10 ・2015/08/10-14, 17-20	8
	Sub-systems/structure interface verification		・2015/06/11-19, 21-26	8
	Vibrations		・2015/02-04	8
	Shock		・2015/07/05-06	8
	Thermal vacuum		・2015/07/11-18 ・2015/08/03-09	24
FM	Electrical performance verifications test – integrated sub-systems		・2015/08/27-28, 31 ・2015/09/01-04, 07-12, 14-18, 21-25, 28-30 ・2015/10/01-02, 05-09, 11-17, 19-23, 26-28 ・2015/11/01, 05-06, 13, 16-20, 23-28, 30 ・2015/12/01-04, 07-09, 13-17	8

FM	Sub-systems/structure interface verification	・2015/09/28-30 ・2015/10/01-02, 05-09, 11-17, 19-23, 26-28	8
	Thermal vacuum	・2015/10/29-30 ・2015/12/10-11	24
	Vibrations	・2015/11/02-03 ・2015/12/18	8
	Shock	・2015/11/04, 10	8
		・2015/11/09, 11-12	2*[a]
	Plasma	・2015/12/21-25, 28-31 ・2016/01/01-02, 04	8
	Batteries charging	・2016/01/05-08	8
	Safety verifications prior to delivery	・2016/01/11-15	2*[b]

*This corresponds to safety related verifications performed at Tsukuba Space Center: a) before/after shock test, b) before satellite delivery.

For all phases, the failure occurrence date, the type of failures, and the total AIT processes time were recorded and the results are presented in Chapter 4, sections 4.1.1 to 4.1.4.

3.2. Simulations methodology

All simulations were performed using MATLAB® version R2013b and the methodology is based on [55]. Moreover, it was demonstrated in several studies [47, 56, 57] that satellites reliability is well-fitted by a two-parameter Weibull distribution with the scale parameter characterizing the failure rate and the shape parameter characterizing the infant mortality. This also applies in this dissertation. The program scripts can be consulted from Appendix A and Appendix B.

The failure rate, r(t), and the reliability are respectively given in Equation 3-1 and Equation 3-2 [58].

$$r(t) = \frac{d[N(t)]}{d(t)} = \frac{d[\lambda t^{1-\alpha}]}{d(t)} = \lambda(1\text{-}\alpha)t^{-\alpha} = \lambda\beta t^{\beta\text{-}1}$$

Equation 3-1. Failure rate definition

With N(t), the cumulative number of failures; t, the time in hours; λ, the scale parameter; α, the growth parameter; and β=1-α, the shape parameter.

$$R(T) = \exp\left(-\int_0^T r(t)d(t)\right) = \exp\left(-\lambda\left(t_T{}^{\beta} - t_0{}^{\beta}\right) + A\right)$$

Equation 3-2. Reliability definition

With t_T=T, the total cumulative testing time in hours; t_0, the time at which the test started; and A, constant from the integration. Here, $t_0=0$, therefore the reliability is as described in Equation 3-3.

$$R(T) = \exp\left(-\lambda T^{\beta} + A\right)$$

Equation 3-3. Simplified reliability equation $t_0=0$

Based on HORYU-IV experience, it is assumed that the satellite goes through seven main different AIT processes: 1) electrical performance verifications (flat satellite), 2) structure assembly, 3) electrical performance verifications (as assembled satellite), 4) thermal vacuum test, 5) plasma test, 6) software performance verifications, and 7) end-to-end missions simulation test. Though in practice some AIT processes overlap with each other, the simulation algorithm assumes that the different AIT processes are carried out in series.

Here, vibrations and shock tests are not taken into account as AIT processes in the simulations. The rationale is that vibrations and shock tests are performed for ensuring safety from the launcher standpoint and they should be performed in agreement with the launcher schedule. Therefore, they are not a main driver per se of the lean satellite program AIT processes schedule, they are a necessity. From experience, it is very unlikely that vibrations and shock tests will take more than 30 days each for a lean satellite program.

From the seven AIT processes, it results that there are in total nine failure modes. First, there is one failure mode that manifests itself only during a specific process among the seven AIT processes (single-AIT failure mode). For example during the structure assembly, it was found out that a piece cannot be assembled because another part of the structure hinders its harness. In this case, the failure is specific to the structure assembly and other AIT processes, such as electrical performance verifications test, can be performed in parallel if needed until the failure is addressed.

Then, there is one failure mode that manifests itself in any of the seven AIT processes (cross-AIT failure mode). For example during electrical performance verifications, it was found out that OBC can read the memory of other sub-systems, but the read data are not correct due to an unknown reason. Due to time constraint, despite the fact that the reason of the failure is still not understood at the end of the electrical performance verifications test, testing must continue and thermal vacuum test is performed. During thermal vacuum test, it can be verified that OBC can read other sub-systems memory, yet the read data are not correct as previously found out.

Finally, there is one failure mode for random failures (random failure mode). For example, though no failure associated with reset function was detected during EM thermal vacuum test, failure occurs during FM thermal vacuum test.

Following these, the failure rate can be expressed as in Equation 3-4.

$$r_i(t) = \lambda_i \beta_i t^{\beta_i - 1} + \lambda_{0i} \beta_{0i} t^{\beta_{0i} - 1} + \theta r_i$$

Equation 3-4. Failure rate expression taking into account the different failure modes

With i, the index for the considered AIT process (i = 1, 2, …, 7) and where the first term accounts for the single-AIT failure mode, the second term accounts for the cross-AIT failure mode, and the third term accounts for the random failure mode.

In the first AIT process, the failure rate is given by Equation 3-4 with i=1. Then, the reliability at time T in the first AIT process is obtained as described in Equation 3-5.

$$R_1(T) = \exp\left(-\int_0^T r_1(t)dt\right) = \exp\left(-\lambda_1 T^{\beta_1} - \lambda_{01} T^{\beta_{01}} - \theta r_1 T\right)$$

Equation 3-5. Reliability at time T for the first AIT process

Where the time T is counted from the start of the first AIT process.

In the second AIT process, the failure rate is written as described in Equation 3-6.

$$r_2(t) = \lambda_2 \beta_2 t^{\beta_2 - 1} + \lambda_{02} \beta_{02} (t + TAIT_1)^{\beta_{02} - 1} + \theta r_2$$

Equation 3-6. Failure rate of the second AIT process

With $TAIT_1$, the total testing time of the first AIT process that should be taken into account for cross-AIT failure mode.

Then, the reliability at time T in the second AIT process is obtained as described in Equation 3-7.

$$R_2(T) = \exp\left(-\int_0^T r_2(t)dt\right) = \exp\begin{pmatrix} -\lambda_2 T^{\beta_2} \\ -\lambda_{02}(T+TAIT_1)^{\beta_{02}} + \lambda_{02}(TAIT_1)^{\beta_{02}} \\ -\theta r_2 T \end{pmatrix}$$

Equation 3-7. Reliability at time T for the second AIT process

Where the time T is counted from the start of the second AIT process and the constant A, from the integration in Equation 3-3, is equal to $\lambda_{02}*(TAIT_1)^{\beta 02}$ for the cross-AIT failure mode.

Likewise, the reliability at time T from the start of the i-th AIT process is obtained as described in Equation 3-8.

$$R_i(T) = \exp\begin{pmatrix} -\lambda_i T^{\beta_i} \\ -\lambda_{0i}(T+TotAIT_i)^{\beta_{0i}} + \lambda_{0i}(TotAIT_i)^{\beta_{0i}} \\ -\theta r_i T \end{pmatrix}$$

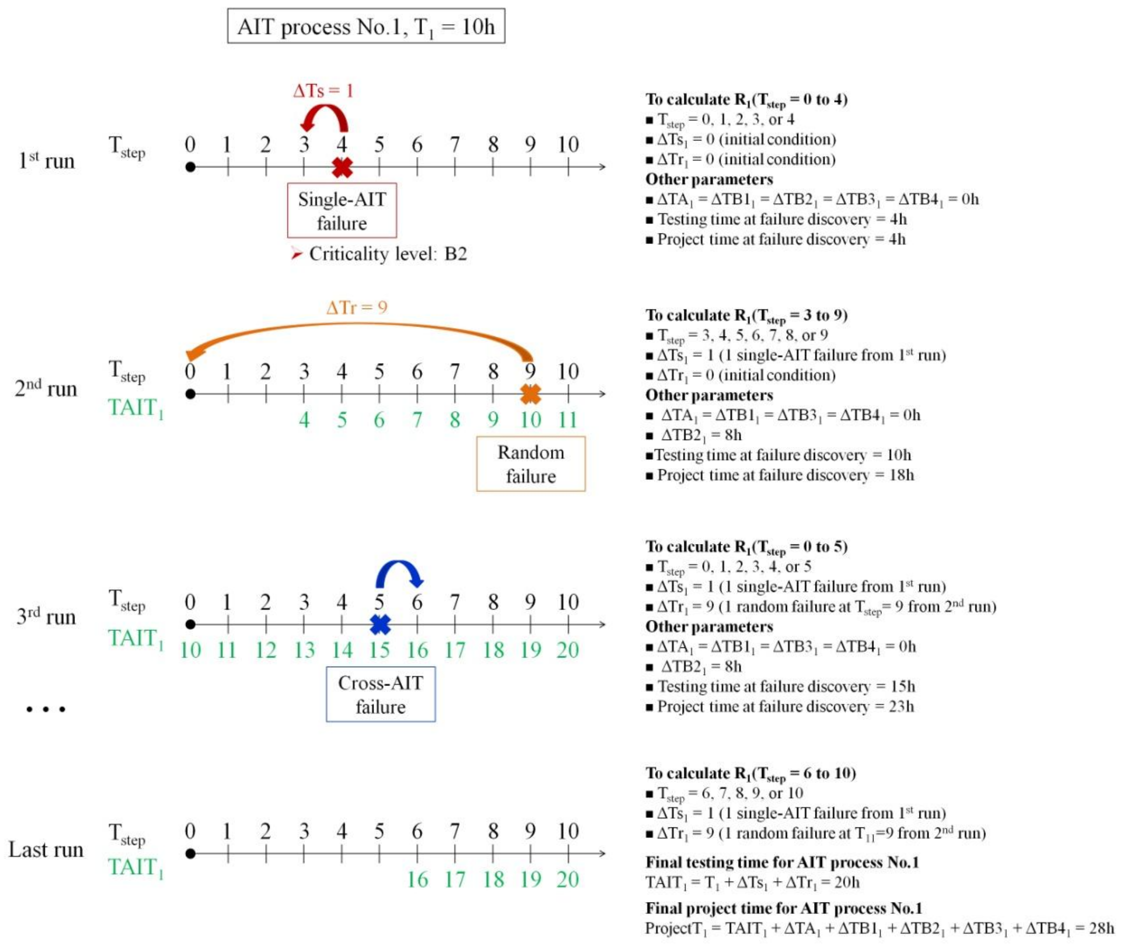
Equation 3-8. Reliability at time T for the i-th AIT process

Where $TotAIT_i$ is the total testing time until the i-th-1 AIT process for cross-AIT failure mode and it is defined by Equation 3-9.

$$TotAIT_i = \sum_{j=1}^{j=i-1} TAIT_j$$

Equation 3-9. Definition of the total testing time. $TotAIT_i$

Each AIT process time is originally defined to be Ti=20, 50, 100, or 200 hours. In the simulation, the testing time is considered in the unit of one hour. Within the testing time Ti, the judgment of whether a failure occurs or not is performed every hour. The time index is designated by Tstep, which is from zero to Ti hours. An illustration of the process is presented in Figure 3-2 for the case of Ti=10h. In this case, Tstep is from zero to 10.



To calculate $R_1(T_{step} = 0 \text{ to } 4)$
- $T_{step} = 0, 1, 2, 3,$ or 4
- $\Delta Ts_1 = 0$ (initial condition)
- $\Delta Tr_1 = 0$ (initial condition)

Other parameters
- $\Delta TA_1 = \Delta TB1_1 = \Delta TB2_1 = \Delta TB3_1 = \Delta TB4_1 = 0h$
- Testing time at failure discovery = 4h
- Project time at failure discovery = 4h

To calculate $R_1(T_{step} = 3 \text{ to } 9)$
- $T_{step} = 3, 4, 5, 6, 7, 8,$ or 9
- $\Delta Ts_1 = 1$ (1 single-AIT failure from 1st run)
- $\Delta Tr_1 = 0$ (initial condition)

Other parameters
- $\Delta TA_1 = \Delta TB1_1 = \Delta TB3_1 = \Delta TB4_1 = 0h$
- $\Delta TB2_1 = 8h$
- Testing time at failure discovery = 10h
- Project time at failure discovery = 18h

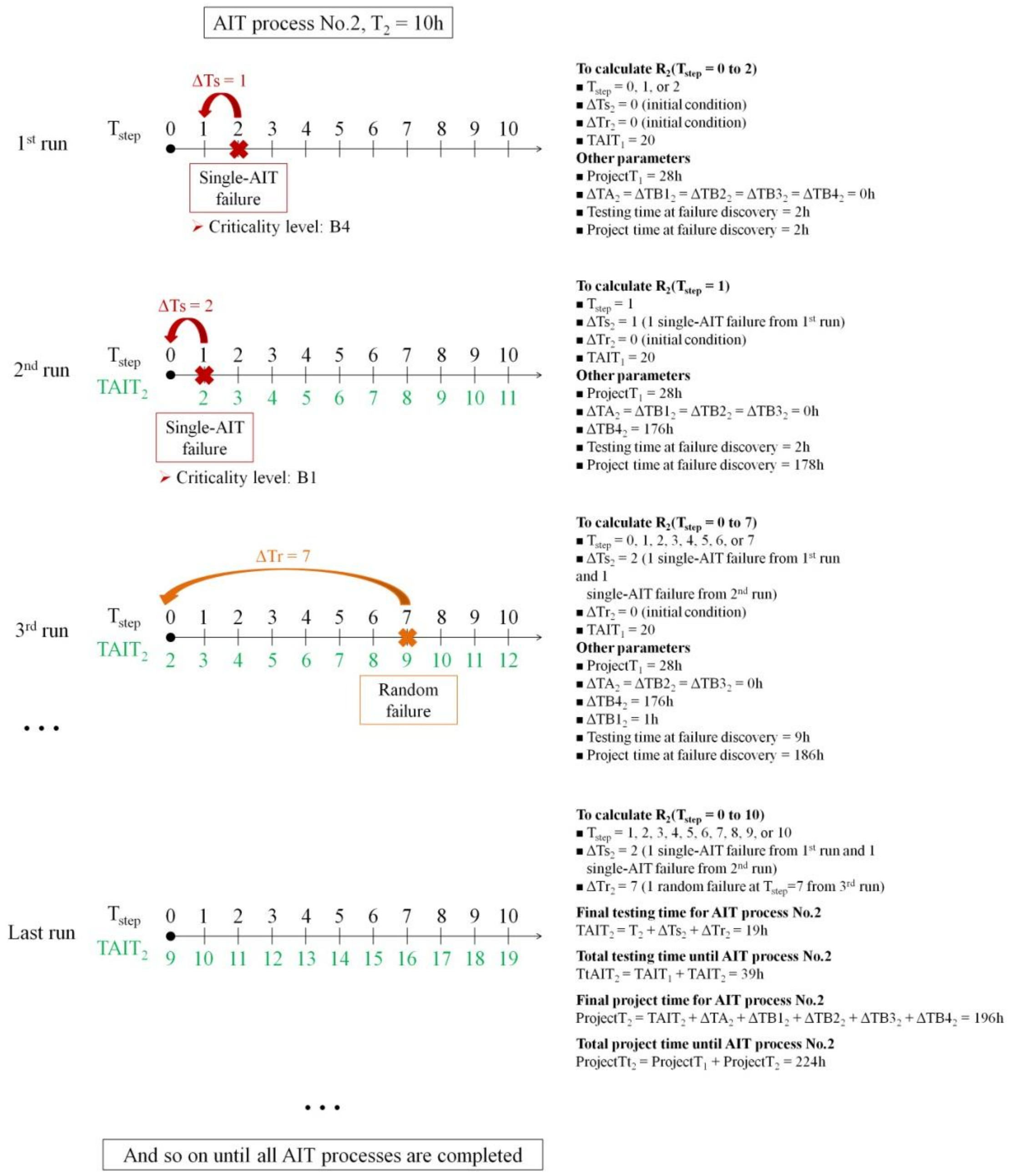
To calculate $R_1(T_{step} = 0 \text{ to } 5)$
- $T_{step} = 0, 1, 2, 3, 4,$ or 5
- $\Delta Ts_1 = 1$ (1 single-AIT failure from 1st run)
- $\Delta Tr_1 = 9$ (1 random failure at T_{step}= 9 from 2nd run)

Other parameters
- $\Delta TA_1 = \Delta TB1_1 = \Delta TB3_1 = \Delta TB4_1 = 0h$
- $\Delta TB2_1 = 8h$
- Testing time at failure discovery = 15h
- Project time at failure discovery = 23h

To calculate $R_1(T_{step} = 6 \text{ to } 10)$
- $T_{step} = 6, 7, 8, 9,$ or 10
- $\Delta Ts_1 = 1$ (1 single-AIT failure from 1st run)
- $\Delta Tr_1 = 9$ (1 random failure at T_{11}=9 from 2nd run)

Final testing time for AIT process No.1
$TAIT_1 = T_1 + \Delta Ts_1 + \Delta Tr_1 = 20h$

Final project time for AIT process No.1
$ProjectT_1 = TAIT_1 + \Delta TA_1 + \Delta TB1_1 + \Delta TB2_1 + \Delta TB3_1 + \Delta TB4_1 = 28h$



To calculate $R_2(T_{step} = 0 \text{ to } 2)$
- T_{step} = 0, 1, or 2
- $\Delta Ts_2 = 0$ (initial condition)
- $\Delta Tr_2 = 0$ (initial condition)
- $TAIT_1 = 20$

Other parameters
- $ProjectT_1$ = 28h
- $\Delta TA_2 = \Delta TB1_2 = \Delta TB2_2 = \Delta TB3_2 = \Delta TB4_2$ = 0h
- Testing time at failure discovery = 2h
- Project time at failure discovery = 2h

To calculate $R_2(T_{step} = 1)$
- T_{step} = 1
- $\Delta Ts_2 = 1$ (1 single-AIT failure from 1st run)
- $\Delta Tr_2 = 0$ (initial condition)
- $TAIT_1 = 20$

Other parameters
- $ProjectT_1$ = 28h
- $\Delta TA_2 = \Delta TB1_2 = \Delta TB2_2 = \Delta TB3_2$ = 0h
- $\Delta TB4_2$ = 176h
- Testing time at failure discovery = 2h
- Project time at failure discovery = 178h

To calculate $R_2(T_{step} = 0 \text{ to } 7)$
- T_{step} = 0, 1, 2, 3, 4, 5, 6, or 7
- $\Delta Ts_2 = 2$ (1 single-AIT failure from 1st run and 1 single-AIT failure from 2nd run)
- $\Delta Tr_2 = 0$ (initial condition)
- $TAIT_1 = 20$

Other parameters
- $ProjectT_1$ = 28h
- $\Delta TA_2 = \Delta TB2_2 = \Delta TB3_2$ = 0h
- $\Delta TB4_2$ = 176h
- $\Delta TB1_2$ = 1h
- Testing time at failure discovery = 9h
- Project time at failure discovery = 186h

To calculate $R_2(T_{step} = 0 \text{ to } 10)$
- T_{step} = 1, 2, 3, 4, 5, 6, 7, 8, 9, or 10
- $\Delta Ts_2 = 2$ (1 single-AIT failure from 1st run and 1 single-AIT failure from 2nd run)
- $\Delta Tr_2 = 7$ (1 random failure at T_{step}=7 from 3rd run)

Final testing time for AIT process No.2
$TAIT_2 = T_2 + \Delta Ts_2 + \Delta Tr_2$ = 19h

Total testing time until AIT process No.2
$TtAIT_2 = TAIT_1 + TAIT_2$ = 39h

Final project time for AIT process No.2
$ProjectT_2 = TAIT_2 + \Delta TA_2 + \Delta TB1_2 + \Delta TB2_2 + \Delta TB3_2 + \Delta TB4_2$ = 196h

Total project time until AIT process No.2
$ProjectTt_2 = ProjectT_1 + ProjectT_2$ = 224h

Figure 3-2. Illustration of the simulation algorithm

For the first simulation run, it is decided whether a failure occurred or not by using a random number η_{i1}. A failure occurs if η_{i1} satisfies Equation 3-10.

$$\eta_{i1} > R_i(T)$$

Equation 3-10. Criteria to determine whether a failure occurred or not

With $T = Tstep + \Delta Tr_i + \Delta Ts_i$, the total testing time at the time index, Tstep, for the considered AIT process. The meaning of ΔTr_i and ΔTs_i is defined in the subsequent paragraphs.

Upon discovery of a failure, the failure mode to which it is associated is determined by using a second random number, η_{i2}. If η_{i2} satisfies Equation 3-11, the failure is associated to single-AIT failure mode. In this case, the number of failures associated to the single-AIT failure mode is incremented, the scale parameter, λ_i, is multiplied by a factor ε_i to take into account the failure rate improvement upon failure discovery, at the next run the AIT process resumes one hour prior to the failure discovery (Tstep=Tstep–1) to take into account the countermeasure to be implemented for the failure resolution as shown in Figure 3-2, and the time delay ΔTs_i associated to the one hour penalty is incremented by one. Here, one hour is assumed sufficient to resolve a failure because the failure is specific to the considered AIT and therefore its origin can be easily assessed. For example, if during the satellite assembly it is noticed that an electronic board cannot be mounted to the structure, the mounting failure origin can easily be assessed within less than an hour to be an interface issue, such as holes misalignment or wrong electronic board dimensions.. It should be noted that failure resolution does not mean that the failure is corrected, but the failure origin is assessed and proper actions can be taken to solve the problem, while the test continues. From the previous example, after noticing the interface issue between structure and the electronic board, other parts of the structure can be assembled while in parallel the electronic board is corrected, which could take more than one hour, but the assembly phase itself does not suffer from this reparation time. Another example is if a failure specific to thermal vacuum occurs, one hour for its correction is most likely unreasonable since it takes several hours just for the chamber to return to atmospheric pressure upon test termination. However, engineers can decide that the step or part of the step that lead to the failure is skipped for the remaining testing, while other functionalities can be tested. In this case, failure resolution corresponds to the decision process, which will unlikely take more than one hour when a failure specific to an AIT process occurs.

$$\eta_{i2} < \frac{\lambda_i \beta_i (T)^{\beta i-1}}{\lambda_i \beta_i (T)^{\beta i-1} + \lambda_{0i}\beta_{0i}(T + TotAIT_i)^{\beta 0i-1} + \theta r_i}$$

Equation 3-11. Criteria to determine single-AIT failure mode

If η_{i2} satisfies Equation 3-12, the failure is associated to cross-AIT failure mode. In this case, the number of failures associated to the cross-AIT failure mode is incremented, the scale parameter, λ_{0i}, is multiplied by a factor ε_{0i} to take into account the failure rate improvement upon failure discovery, and at the next run the AIT process continues, i.e. the time index is incremented by one hour (Tstep=Tstep+1). Cross-AIT failures are failures common to any of the AIT processes because they could not be resolved at the end of a specific AIT process, however, due to time constraint, testing should pursue. Therefore in this case no time penalty is taken into account.

$$\frac{\lambda_i \beta_i (T)^{\beta i-1}}{\lambda_i \beta_i (T)^{\beta i-1} + \lambda_{0i} \beta_{0i} (T + TotAIT_i)^{\beta 0i-1} + \theta r_i} \leq \eta_{i2}$$

and

$$\eta_{i2} < \frac{\lambda_i \beta_i (T)^{\beta i-1} + \lambda_{0i} \beta_{0i} (T + TotAIT_i)^{\beta 0i-1}}{\lambda_i \beta_i (T)^{\beta i-1} + \lambda_{0i} \beta_{0i} (T + TotAIT_i)^{\beta 0i-1} + \theta r_i}$$

Equation 3-12. Criteria to determine cross-AIT failure mode

If η_{i2} satisfies Equation 3-13, the failure is associated to random failure mode. In this case, the number of failures associated to the random failure mode is incremented, the random parameter, θr_i, is multiplied by a factor εr_i to take into account the failure rate improvement upon failure discovery, at the next run the AIT process is restarted from the beginning (Tstep=0), and the time delay ΔTr_i associated to the time penalty of restarting the AIT process is incremented by Tstep. For an example, see Figure 3-2. In this example, the random failure occurs at Tstep=9. By this time, the total testing time accumulated is 10h as one hour was already added due to the previous single-AIT type failure. Since it is a random failure, the test goes back to the beginning, but 10h testing time was already accumulated. Random failures are unpredictable failures that can occur due to parts mishandling or that occur during a test phase that was performed in a similar manner at an earlier stage of development, yet no failure was discovered during this previous stage of development. Therefore in this case, the failure origin cannot be determined easily or in some cases additional tests should be performed and to take into account the long time associated with failure origin investigation and/or additional testing, it is assumed in the simulation that the test restart from the beginning.

$$\eta_{i2} \geq \frac{\lambda_i \beta_i (T)^{\beta i-1} + \lambda_{0i} \beta_{0i} (T + TotAIT_i)^{\beta 0i-1}}{\lambda_i \beta_i (T)^{\beta i-1} + \lambda_{0i} \beta_{0i} (T + TotAIT_i)^{\beta 0i-1} + \theta r_i}$$

Equation 3-13. Criteria to determine random failure mode

In case a failure is defined to belong to the single-AIT failure mode, the failure criticality is determined to evaluate the influence of the discovered failure on the overall lean satellite project development. In the case of failures belonging to cross-AIT or random failure modes, the criticality of the failure is not evaluated. For random failure, since it is difficult to determine the origin of the failure, the undergoing AIT process is assumed to restart. In this case, the failure is the most critical to the project AIT phase and associated AIT process cost. Therefore, its impact on the project, outside the considered AIT process, can be assumed negligible. For a failure belonging to cross-AIT failure mode, the failure is common to several AIT processes. In other word, it can be assumed that the failure was discovered and investigated at a previous AIT, subsequent time was thus dedicated to its resolution since at the previous AIT it could have been considered either as a single-AIT failure or random failure. Therefore, in this case, the failure criticality was already evaluated and it can be assumed it will be redundant to reevaluate it.

For the evaluation of failure criticality, five levels of criticality were identified as described below.

- **Level A:** no time for reparation is necessary. For example, a circuit line should be cut, which can be executed within seconds.
- **Level B1:** low repairable time, less than 1 hour. For examples, easy components soldering or removal, a few in-house design changes, etc.
- **Level B2:** medium repairable time, up to 1 day (8h). For examples, electrical design modifications, difficult components soldering or removal, several in-house design changes, etc.
- **Level B3:** high repairable time, up to 1 business week (40h). For examples, new

parts need to be ordered (components in stock), minor design changes need to be outsourced, etc.

- **Level B4:** very high repairable time, up to 1 month (176h). For examples, new parts need to be ordered (components out of stock), major design changes need to be outsourced, etc.

Upon discovery of a single-AIT failure mode, a third random number, η_{i3}, is generated to determine the failure criticality level. If η_{i3} satisfies Equation 3-14, the failure criticality corresponds to level A and no additional time is taken into account for its repair.

$$\eta_{i3} < \frac{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}}{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1} + \lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1} + \lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1} + \lambda_{iB3}\beta_{iB3}(T)^{\beta iB3-1} + \lambda_{iB4}\beta_{iB4}(T)^{\beta iB4-1}}$$

Equation 3-14. Criteria to determine failure criticality, level A

With λ_{iA}, λ_{iB1}, λ_{iB2}, λ_{iB3}, and λ_{iB4}, the scale parameters for each critical level for a considered i-th AIT process; and β_{iA}, β_{iB1}, β_{iB2}, β_{iB3}, and β_{iB4}, the shape parameter for each critical level for a considered i-th AIT process.

The scale parameters, λ_{iA}, λ_{iB1}, λ_{iB2}, λ_{iB3}, and λ_{iB4}, are independent from each other and are independent from the scale parameters, λ_i and e_{0i}, defined in the failure mode algorithm. Similarly, the shape parameters, β_{iA}, β_{iB1}, β_{iB2}, β_{iB3}, and β_{iB4}, are independent from each other and are independent from the shape parameters, β_i and β_{0i}, defined in the failure mode algorithm.

If η_{i3} satisfies Equation 3-15, the failure criticality corresponds to level B1 and one hour is added on the overall project development time span to take into account the repair time.

$$\frac{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}}{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}+\lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1}+\lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1}+\lambda_{iB3}\beta_{iB3}(T)^{\beta iB3-1}+\lambda_{iB4}\beta_{iB4}(T)^{\beta iB4-1}}\leq\eta_{i3}$$

and

$$\eta_{i3}<\frac{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}+\lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1}}{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}+\lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1}+\lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1}+\lambda_{iB3}\beta_{iB3}(T)^{\beta iB3-1}+\lambda_{iB4}\beta_{iB4}(T)^{\beta iB4-1}}$$

Equation 3-15. Criteria to determine failure criticality, level B1

If η_{i3} satisfies Equation 3-16, the failure criticality corresponds to level B2 and 8 hours (1 business day) are added on the overall project development time span to take into account the repair time.

$$\frac{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}+\lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1}}{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}+\lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1}+\lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1}+\lambda_{iB3}\beta_{iB3}(T)^{\beta iB3-1}+\lambda_{iB4}\beta_{iB4}(T)^{\beta iB4-1}}\leq\eta_{i3}$$

and

$$\eta_{i3}<\frac{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}+\lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1}+\lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1}}{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}+\lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1}+\lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1}+\lambda_{iB3}\beta_{iB3}(T)^{\beta iB3-1}+\lambda_{iB4}\beta_{iB4}(T)^{\beta iB4-1}}$$

Equation 3-16. Criteria to determine failure criticality, level B2

If η_{i3} satisfies Equation 3-17, the failure criticality corresponds to level B3 and 40 hours (1 business week) are added on the overall project development time span to take into account the repair time.

$$\frac{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}+\lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1}+\lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1}}{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}+\lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1}+\lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1}+\lambda_{iB3}\beta_{iB3}(T)^{\beta iB3-1}+\lambda_{iB4}\beta_{iB4}(T)^{\beta iB4-1}}\leq\eta_{i3}$$

and

$$\eta_{i3}<\frac{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}+\lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1}+\lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1}+\lambda_{iB3}\beta_{iB3}(T)^{\beta iB3-1}}{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1}+\lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1}+\lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1}+\lambda_{iB3}\beta_{iB3}(T)^{\beta iB3-1}+\lambda_{iB4}\beta_{iB4}(T)^{\beta iB4-1}}$$

Equation 3-17. Criteria to determine failure criticality, level B3

If η_{i3} satisfies Equation 3-18, the failure criticality corresponds to level B4 and 176 hours (1 business month) are added on the overall project development time span to take into account the repair time.

$$\eta_{i3} \geq \frac{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1} + \lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1} + \lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1} + \lambda_{iB3}\beta_{iB3}(T)^{\beta iB3-1}}{\lambda_{iA}\beta_{iA}(T)^{\beta iA-1} + \lambda_{iB1}\beta_{iB1}(T)^{\beta iB1-1} + \lambda_{iB2}\beta_{iB2}(T)^{\beta iB2-1} + \lambda_{iB3}\beta_{iB3}(T)^{\beta iB3-1} + \lambda_{iB4}\beta_{iB4}(T)^{\beta iB4-1}}$$

Equation 3-18. Criteria to determine failure criticality, level B4

Moreover, in each case, the number of failures associated to the level A, B1, B2, B3, or B4 failure criticality is incremented and the corresponding scale parameter, λ_A, λ_{B1}, λ_{B2}, λ_{B3}, or λ_{B4}, is multiplied by the corresponding factor ε to take into account the failure rate improvement.

For the simulation of the reliability after launch (Equation 3-19), each AIT process was run 1000 times and the resulting average value of λ_0, λ_i, β_0, β_i, θr, the total testing time of each AIT, $TAIT_i$, and the average total testing time of all AIT processes combined, TtAIT (Equation 3-20), were used for the calculations.

$$R(T_{after}) = \exp\begin{pmatrix} -\sum_{i=1}^{i=7}\lambda_{ilast}(T_{after} + TAIT_i)^{\beta ilast} + \sum_{i=1}^{i=7}\lambda_{ilast}(TAIT_i)^{\beta ilast} \\ -\lambda_{0last}(T_{after} + TtAIT)^{\beta 0last} + \lambda_{0last}(TtAIT)^{\beta 0last} \\ -\theta r_{last} T_{after} \end{pmatrix}$$

Equation 3-19. Reliability after launch based on 1000 simulation runs

$$TtAIT = \sum_{i=1}^{i=7} TAIT_i$$

Equation 3-20. Definition of the average total testing time for all AIT processes combined, TtAIT

The flowchart summarizing the simulation algorithm is presented in Figure 3-3.

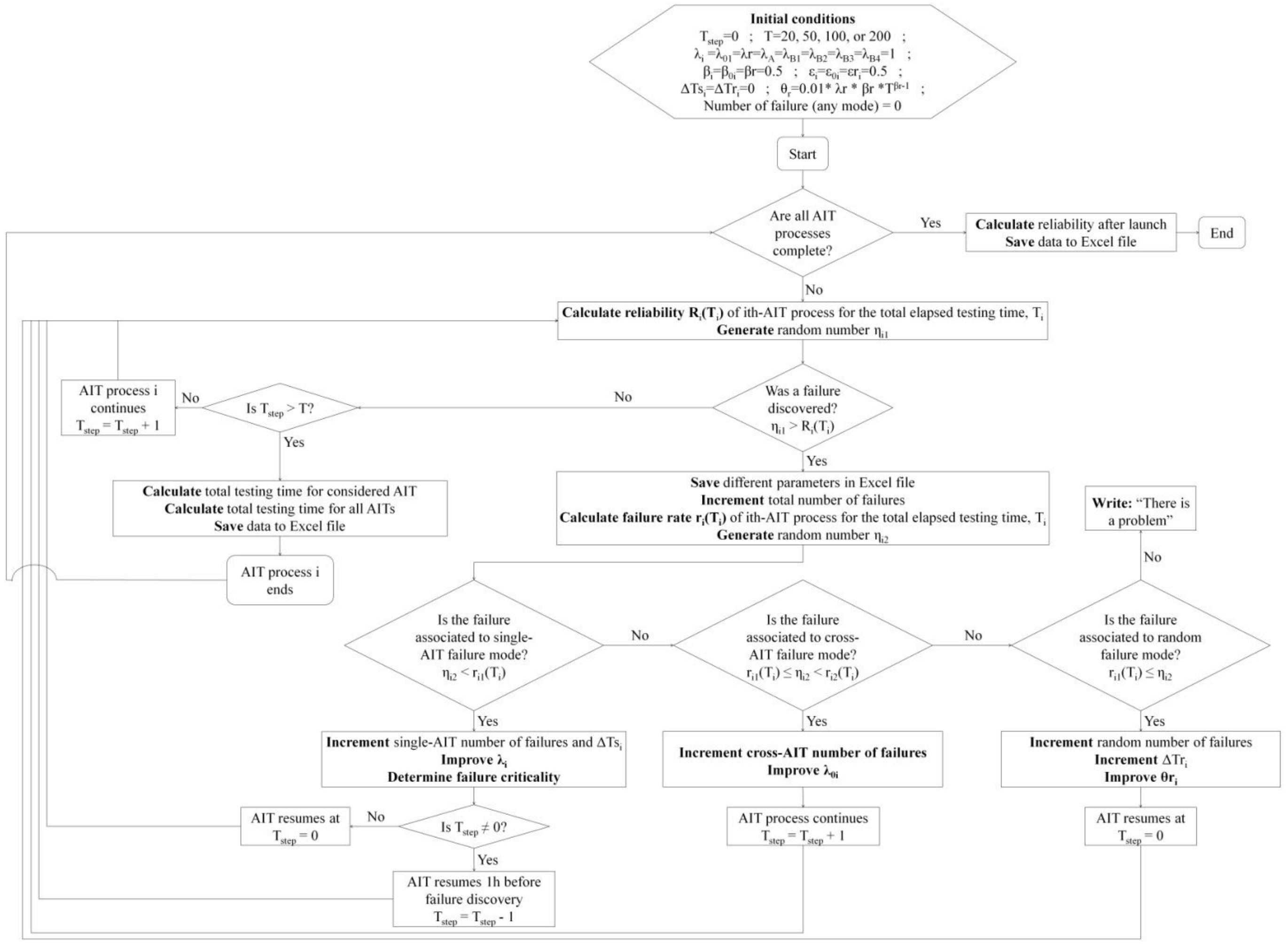

Initial conditions
$T_{step}=0$; $T=20, 50, 100$, or 200 ;
$\lambda_i=\lambda_{0i}=\lambda r=\lambda_A=\lambda_{B1}=\lambda_{B2}=\lambda_{B3}=\lambda_{B4}=1$;
$\beta_i=\beta_{0i}=\beta r=0.5$; $\varepsilon_i=\varepsilon_{0i}=\varepsilon r_i=0.5$;
$\Delta Ts_i=\Delta Tr_i=0$; $\theta_r=0.01* \lambda r * \beta r * T^{\beta r-1}$;
Number of failure (any mode) = 0
Start
Are all AIT processes complete?
Yes
Calculate reliability after launch
Save data to Excel file
End
No
Calculate reliability $R_i(T_i)$ of ith-AIT process for the total elapsed testing time, T_i
Generate random number η_{i1}
AIT process i continues
$T_{step} = T_{step} + 1$
No
Is $T_{step} > T$?
No
Was a failure discovered?
$\eta_{i1} > R_i(T_i)$
Yes
Yes
Calculate total testing time for considered AIT
Calculate total testing time for all AITs
Save data to Excel file
AIT process i ends
Save different parameters in Excel file
Increment total number of failures
Calculate failure rate $r_i(T_i)$ of ith-AIT process for the total elapsed testing time, T_i
Generate random number η_{i2}
Write: "There is a problem"
No
Is the failure associated to single-AIT failure mode?
$\eta_{i2} < r_{i1}(T_i)$
No
Is the failure associated to cross-AIT failure mode?
$r_{i1}(T_i) \leq \eta_{i2} < r_{i2}(T_i)$
No
Is the failure associated to random failure mode?
$r_{i1}(T_i) \leq \eta_{i2}$
Yes
Yes
Yes
Increment single-AIT number of failures and ΔTs_i
Improve λ_i
Determine failure criticality
Increment cross-AIT number of failures
Improve λ_{0i}
Increment random number of failures
Increment ΔTr_i
Improve θr_i
AIT resumes at $T_{step} = 0$
No
Is $T_{step} \neq 0$?
AIT process continues
$T_{step} = T_{step} + 1$
AIT resumes at $T_{step} = 0$
Yes
AIT resumes 1h before failure discovery
$T_{step} = T_{step} - 1$

Figure 3-3. Simulation algorithm flowchart

CHAPTER 4 – Results and Discussion

Results and corresponding discussions were presented in [50, 51]. However, for the readers to be able to fully capture the recommendations established in this research, the results and their discussion are also detailed in this chapter.

4.1. HORYU-IV experimental results and discussion

4.1.1. Investigation of the number of failures vs. testing time

Plots of the cumulative number of failures against cumulative testing time for the overall project development and each development phase are shown in Figure 4-1. It can be observed that at the end of each AIT phase, there is a plateau, which indicates that even if the AIT processes time is increased, only a few new failures are discovered. This is in good agreement with [59] that pointed out that most of failures are discovered during the early stage of testing.

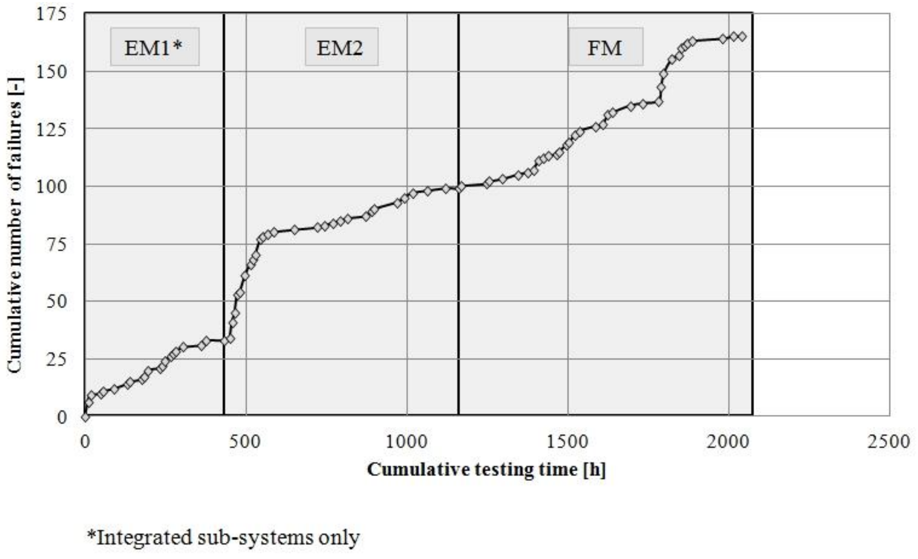


Figure 4-1. HORYU-IV cumulative number of failures vs. cumulative testing time for all the AIT phases

For EM1 (Figure 4-2), when OBC and EPS were tested as standalone sub-systems, the plateau can also be observed, whereas for COM it is absent. This is explained by the fact that all OBC and EPS functionalities, as presented in Figure 3-2, could be thoroughly tested within the 40h testing, whereas for COM, data transmission and command reception could not be verified, due to the discovery of critical failures with unknown origin. Moreover, due to schedule constraints, COM testing as a standalone sub-system had to be interrupted after about 55h of testing to move to integration testing. Therefore, the failure rate did not reach a constant value, i.e. the failure mode did not switch to random type failure, and no plateau was observed.

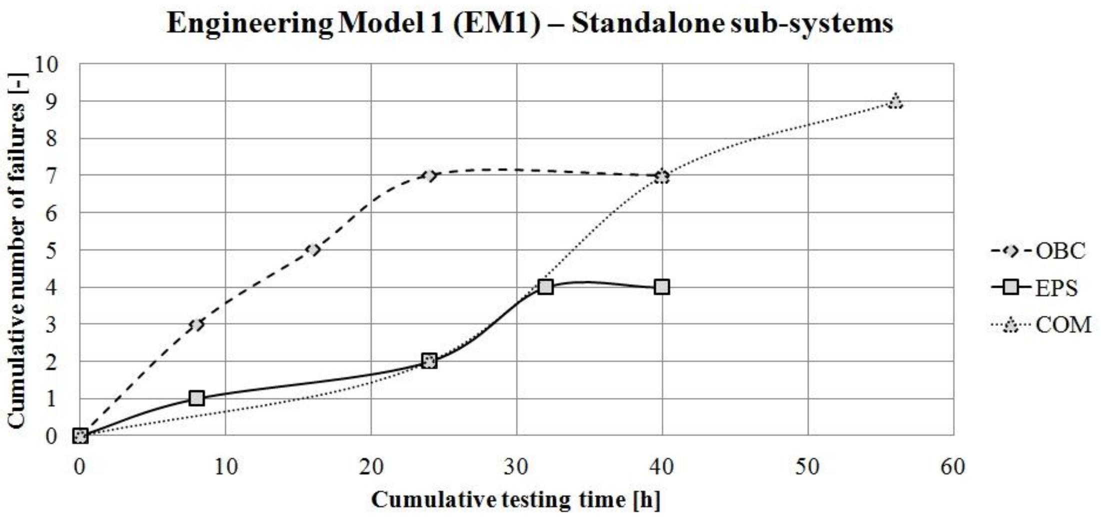


Figure 4-2. HORYU-IV cumulative number of failures vs. cumulative testing time during EM1 phase for standalone sub-systems testing

Upon integration of OBC, EPS, and COM, hereafter referred as mother board, it can be observed from Figure 4-3 that most failures were discovered within the first 25h of testing and after 50h of testing, 45% of the total number of failures were recorded for that integration phase. Moreover, the discovery of the remaining 55% of failures took 5 times more time than the discovery of the 45% of failures. This is explained by the fact that many failures emerged upon integration of the sub-systems, but after 50h of testing, the functionalities to be verified as described in Figure 3-2, were verified except for the remaining COM failures that emerged during COM standalone testing. To resolve the COM failures, different assumptions on their origin were established and corresponding countermeasures were implemented, but they were unsuccessful for the complete resolution of the COM failures. This period of trials-and-errors resulted in a long period of testing mostly focusing on COM with only a few new failures discovered from 50h to 250h of mother board testing.

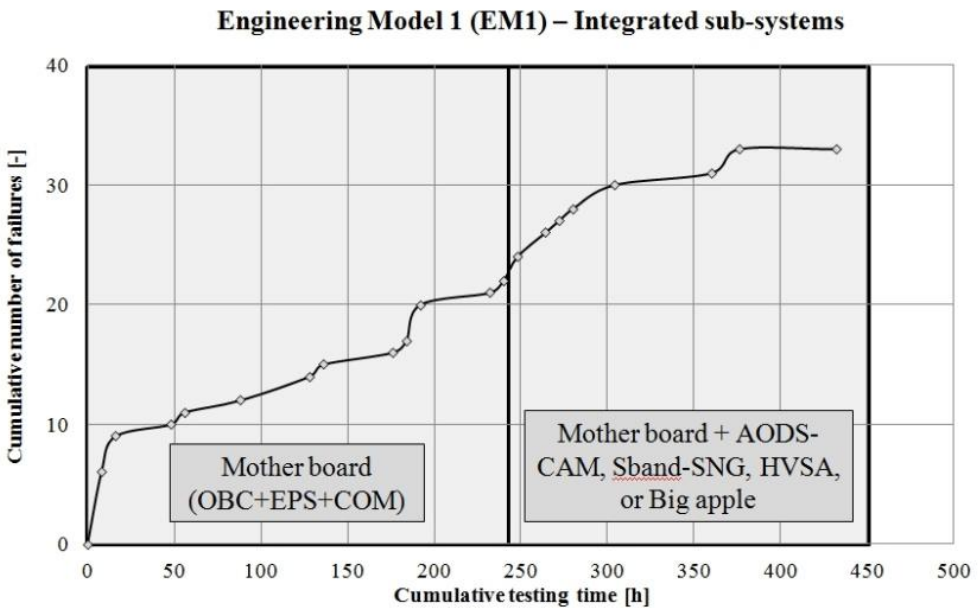


Figure 4-3. HORYU-IV cumulative number of failures vs. cumulative testing time during EM1 phase for integrated sub-systems testing

Upon integration of the mother board with the different sub-systems, COM failures remained and a similar testing pattern as for mother board testing can be observed. Within about 50h of testing, 67% of the failures resulting from the different sub-systems integration were discovered, whereas it took 4 times more time to discover the remaining 33%. There are two main reasons in this case: 1) as for the mother board, COM remaining failures and the associated repairs trials-and-errors increased the AIT period and 2) another critical failure emerged with unknown origin for which sub-systems memory could not be accurately read when integrated with the mother board. During integration testing of mother board with other sub-systems, these two failures were the main drivers for long time testing after the initial 50h.

At the end of nearly 450h of EM1 testing, COM failures were repaired and hence, were not a driver in the EM2 testing, whereas memory reading failure remained during EM2 testing.

From HORYU-IV EM1 testing experience, it appeared that thorough standalone sub-system testing prior to integration helps in reducing the integration testing time. Moreover, due to the increase of the system complexity upon integration, origin and cause of failures are more difficult to trace, which results in longer testing time. Therefore, it seems that from a time management standpoint, it will be worthwhile for lean satellite developers to allocate sufficient testing time to prove the soundness of a sub-system individually prior to its integration with other sub-systems.

For EM2 (Figure 4-4), as for EM1, most of the failures, 71%, are detected during the early stage of AIT phase, within the first 150h. This early stage high failure discovery rate is driven by the mechanical interface between the structural and electrical elements. The failure types vary from quick modifications such as correcting hole positions to more complex and costly modifications such as redesigning panels or PCBs to accommodate unforeseen fitting disturbances.

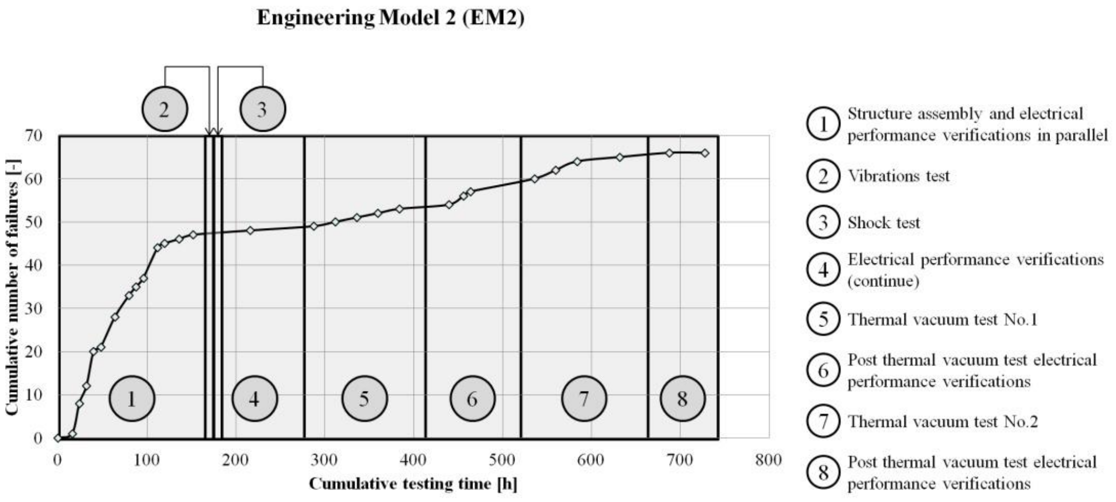


Figure 4-4. HORYU-IV cumulative number of failures vs. cumulative testing time during EM2 phase

HORYU-IV structural elements were ordered to a local manufacturer, whose proximity allowed for fast treatment of the design modifications requests and pieces could be re-manufactured or modified within 1 or 2 business days for the simple design to 5 business days for the more complicated design. Moreover, there were no charges for delivery to or from the manufacturer since the site was accessible by short car ride from the university.

After the EM satellite was assembled, electrical performance verifications as well as launch and space environment related testing took place. Though no failures occurred for the launch related testing, several failures were discovered during the thermal vacuum tests. It should be noted that the testing time indicated as “thermal vacuum” on Figure 4-4 includes electrical performance verifications prior and after the actual start of chamber’s vacuum and temperature cycles and most of failures were actually discovered during these pre- and post-test verifications. When the chamber’s vacuum and temperature cycles were running, only three new failures were discovered at 360h and 384h during thermal vacuum No.1 and at 584h during thermal vacuum No.2.

From EM AIT phase, there are three main lessons learned:

1) mechanical and electrical interfaces verification should not be overlooked and satellite design should not rely solely on CAD;

2) sufficient margin should be assumed to ensure harness and its associated curvature radius can be accommodated within the space between different structural elements and PCBs; and

3) it seems very desirable from a time and cost standpoint that lean satellites developers select local manufacturers whenever possible.

For FM (Figure 4-5), there were in total three sets of PCBs manufactured for each sub-system to be able to perform AIT processes in parallel. One set of PCBs was dedicated to be mounted with the FM satellite structure and be flown in space. The second set was dedicated to electrical functional verifications testing, i.e. all PCBs connected together without following satellite structure, namely flat satellite testing. The third set was dedicated to payloads individual electrical performance verifications and their space related testing, such as LEO simulated environment testing. Unlike for EM1 or EM2, new failures discovery rate for FM is relatively smooth until about 600h of testing. This is mainly explained by three factors: 1) lessons learned during EM phase were implemented and for example less structural/electrical interface failures occurred, 2) electrical performance verifications were thoroughly performed during EM and only a few new failures emerged mainly due to final impedance adjustments, pins assignments correction in software, or FM external parts inspection upon delivery, and 3) after EM, the hardware design was well-defined and tested, and modifications in FM were mostly driven by software modifications, which were difficult to record. Though it should be noted that the latter results in error of the presented data for the cumulated number of failures, the trend of the failures discovery is assumed to be accurate since software failures were also under documented in EM phase. Therefore, EM and FM phases failures were recorded based on the same error and hence are comparable data sets.

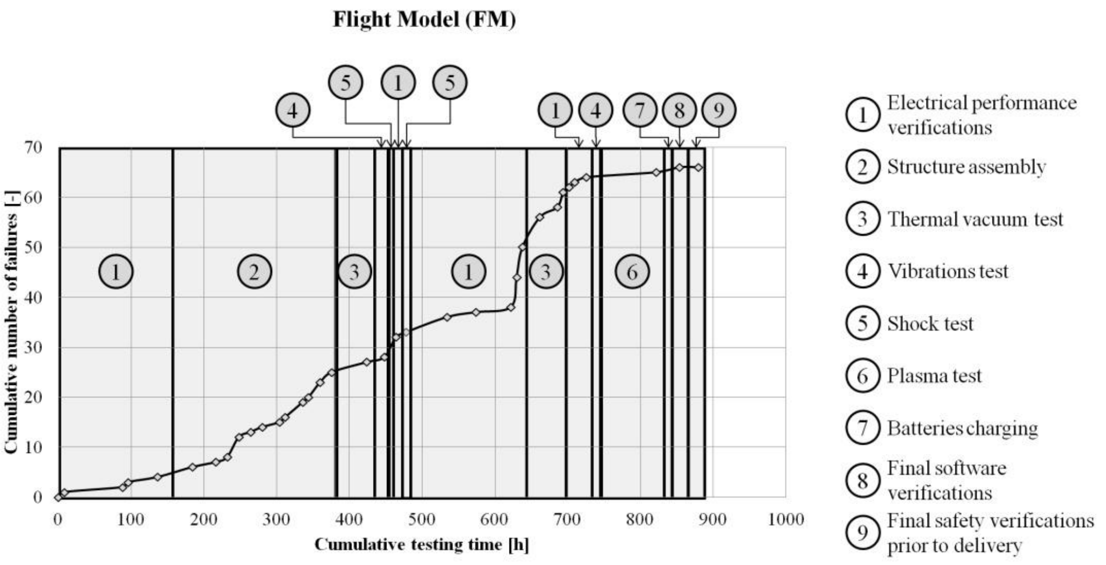


Figure 4-5. HORYU-IV cumulative number of failures vs. cumulative testing time for FM phase

For the first thermal vacuum test, it should be noted that a failure that could have incapacitated the satellite during its first eclipse time was discovered during the first cycle at low temperature. This demonstrates the importance of carrying out space environment testing even for a small number of cycles.

From the end of the third electrical performance verifications test, around 600h, until the end of the second vibrations test around 730h, the profile shows a sudden increase of the failure rate. This is mainly due to the fact that during this period, payloads functions performances were tested more deeply to ensure mission minimum success. For examples, stress was put on integrated OBO, AVC, and S-band sub-systems since they are essential to the satellite main mission minimum success achievement. Stress was also put on L-band to ensure satellite can be reset even if all other implemented reset systems fail. Moreover, the ground station (GS) compatibility with the satellite was tested for the first time, which resulted in several modifications of the GS software for decoding satellite data.

After 730h of testing, the failure rate became constant and not many new failures were discovered upon testing. However, it should be noted that for the plasma testing period, the plot shows only one new failure discovery, which could indicate that plasma testing might be unnecessary. However, the discovered failure was the inability of the main mission's on-board oscilloscope to be triggered when discharges occurred. In case this failure would not have been uncovered during plasma testing, this would have resulted in the failure of the main mission after the satellite launch. Hence, HORYU-IV minimum success criteria would not have been satisfied. Moreover, the last phase of safety review was under preparation in parallel of plasma testing and failures record could not be as thorough as it was in the previous development phases. As a result, many failures related to software development were discovered and modifications were implemented accordingly, but those were not recorded and do not appear on Figure 4-5. Therefore, for HORYU-IV, plasma testing was very useful in revealing failures that would have impeded the mission minimum success and satellite reliability after launch.

Overall, in case the lean satellite program involves missions depending on the actual plasma conditions of the considered orbit, plasma testing seems to be helpful to unveiling failures and improving satellite reliability after launch.

From FM AIT phase, there are two main lessons learned:

1) even though the satellite was well-tested during EM phase, it is necessary to re-test the whole system as FM to ensure new components functionalities soundness. This is true even in the case that design or components batch are the same and this can take a non negligible amount of time. In HORYU-IV case, it took about 75 testing days; and
2) thorough integrated testing of all functionalities of sub-systems critical to mission minimum success is necessary for the improvement of the satellite reliability after launch.

Overall, HORYU-IV experimental data demonstrate that thorough short AIT cycles are critical for the discovery of failures and therefore for the satellite reliability improvement after launch. Moreover, space environment related tests are critical to unveil failures that could cost the satellite life within its first hours of operation after launch. Then, not only interfaces between two electrical sub-systems should be verified, but also interfaces between mechanical and electrical sub-systems should be carefully monitored and tested as early as possible in case modifications are required.

Another interesting observation concerns the failures occurrence rate. From the failure rate (Equation 3-1), the time before failure (TBF) can be estimated using

Equation 4-1 [58] and the results are presented in Figure 4-6.

$$\mathrm{TBF} = \frac{1}{\mathrm{r}(\mathrm{t})}$$

Equation 4-1. Definition of the time before failure

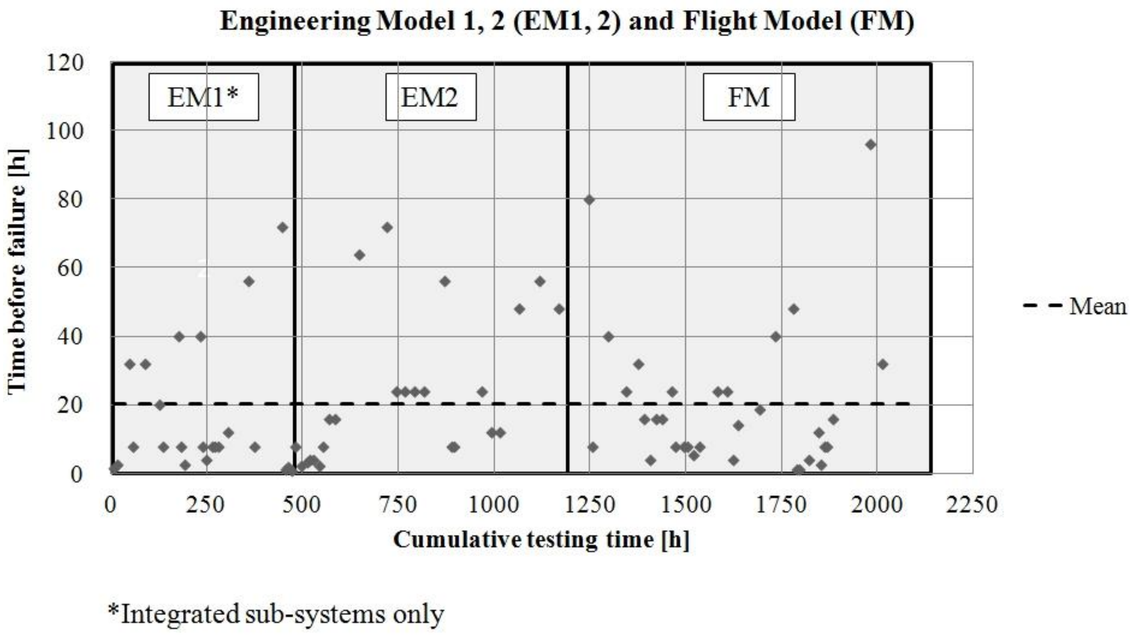


Figure 4-6. HORYU-IV time before failure against the cumulative testing time

Though the TBF varies during the project, it can be observed that overall the TBF is increasing. The TBF is best improved by about 37% between EM1 and EM2, whereas it remains almost constant between EM2 and FM at an average TBF of 22h and 21h, respectively. On average, for HORYU-IV, a new failure occurred every 20h of AIT processes. Even when new interfaces are tested or when the tested system complexity increases, new failures were discovered as fast as every 16h. These results tend to indicate that, apart from the launch environment related tests, a minimum of 20h of AIT processes is necessary for a failure to emerge.

Finally, as a case in point, for HORYU-IV case the total AIT processes time spent during EM and FM accounted for almost 2/3 of the total development time for these two phases combined.

4.1.2. Determination of shape and scale parameters

In this section, the cumulative failure rate, C(t), against the cumulative testing time and the resulting Duane plot were plotted for the overall project development (Figure 4-7). As shown by Figure 4-7b, HORYU-IV experimental data can be fitted by a straight line with a negative slope, which indicates the satellite reliability improvement. From the obtained data and using Equation 4-2 [58], the scale parameter, λ, and the shape parameter, β, are estimated to be respectively 0.57 and 0.83 for HORYU-IV case.

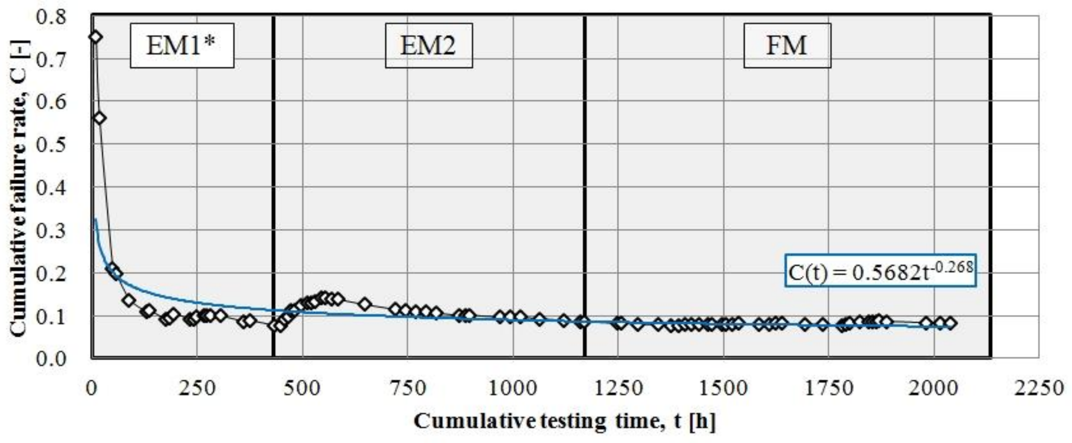
$$C(t) = \frac{N(t)}{t} = \lambda t^{-\alpha} = \lambda t^{\beta-1}$$

Equation 4-2. Cumulative failure rate, C(t)

With C(t), the cumulative failure rate; N(t), the cumulative number of failures; t, the time in hours; λ, the scale parameter; α, the growth parameter; and β=1-α, the shape parameter.

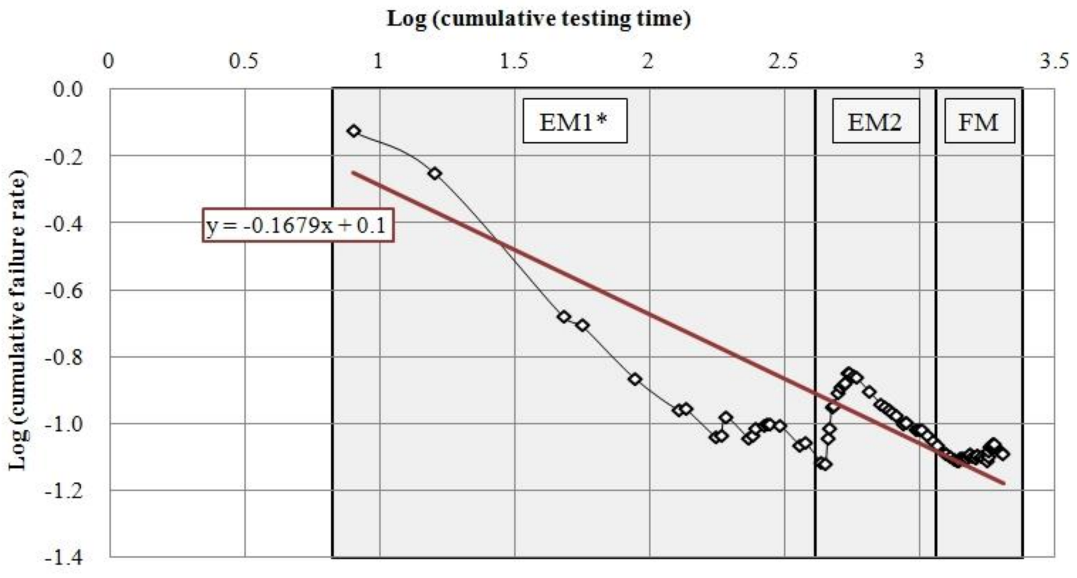
Compared to data survey of small satellites from [18], the obtained shape parameter, which characterizes infant mortality seems high. However, it should be kept in mind that in HORYU-IV case it corresponds to data sets that were taken prior to launch, while the satellite can still be considered a repairable system. Therefore, many failures emerged and infant mortality of the system is indeed high.

a)



*Integrated sub-systems only

b)



*Integrated sub-systems only

Figure 4-7. HORYU-IV cumulative failure rate against cumulative testing time (a) and resulting Duane plot (b)

4.1.3. System's reliability

As part of HORYU-IV lean philosophy approach, one of the management concerns was to ensure that the main mission had sufficient reliability to be successfully executed in space ensuring satellite's minimum success criteria will be achieved.

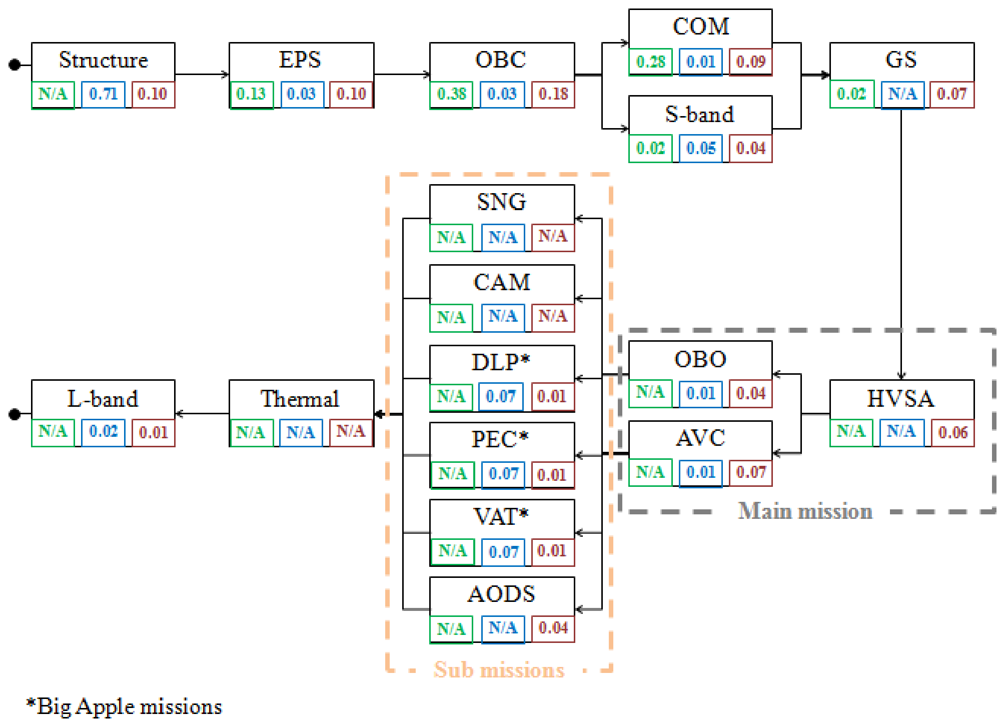
Based on this management strategy, reliability block diagrams (RBD) of the satellite system, bus sub-systems, and payload sub-systems were established (Figure 4-8). Overall, the RBDs are a representation of the different sub-systems interaction from a reliability standpoint. The arrow direction shows the sub-systems order to be able to achieve satellite minimum success requirements. The RBDs start on the left side with the most important sub-system from a reliability standpoint. For example, if we look at the whole system RBD, the first sub-system is structure. Without a reliable structure, the satellite cannot be launched; hence, the missions cannot be executed and the satellite's minimum success criteria cannot be achieved. In other words, the loss of the structure will jeopardize the entire satellite and ensuring structure reliability is critical to the achievement of the satellite's minimum success criteria. After structure, the most critical sub-system is EPS. Without a sufficient EPS reliability, power generation, storage, and supply cannot be ensured and electronics necessary to handling information internally through OBC or externally through COM or S-band cannot be turned ON and again it would jeopardize the entire satellite. The other sub-systems are similarly ordered until the least important sub-system from a reliability standpoint has been identified.

It can be noticed that some sub-systems are displayed in parallel. This means that these sub-systems are equally important from a reliability standpoint. For example for the whole system RBD, COM and S-band are in parallel because from a system standpoint at least one of the two needs to work properly to ensure the satellite communication and therefore ensure the satellite's minimum success criteria are achieved. However, in the bus sub-systems RBD, S-band and GS appear more important than COM because one of the minimum success criteria is to acquire a picture of a discharge. This picture is too large to be downloaded by UHF and S-band, which offers a higher data speed rate, is required. Therefore, from a bus sub-systems standpoint, S-band and its associated GS are more critical to the achievement of the satellite’s minimum criteria than COM.

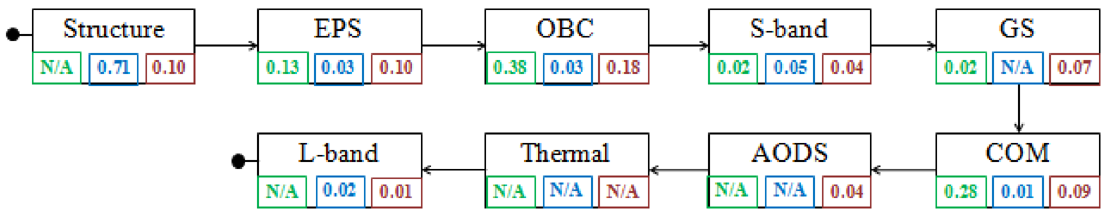
The numbers in the bottom of each sub-system box indicate the sub-system contribution to failures at the end of EM1 (green), EM2 (blue), and FM (red) phases. These numbers come directly from the taxonomy analysis detailed in section 4.1.4.

Based on these, the reliability, expressed as the probability of non failure of a considered sub-system can be estimated. Then, the reliability of the whole system, the bus sub-systems, and the payload sub-systems at the different AIT phases was calculated. The numerical results are shown in Table 4-1 and the evolution of the reliability against cumulative testing time is shown in Figure 4-9.

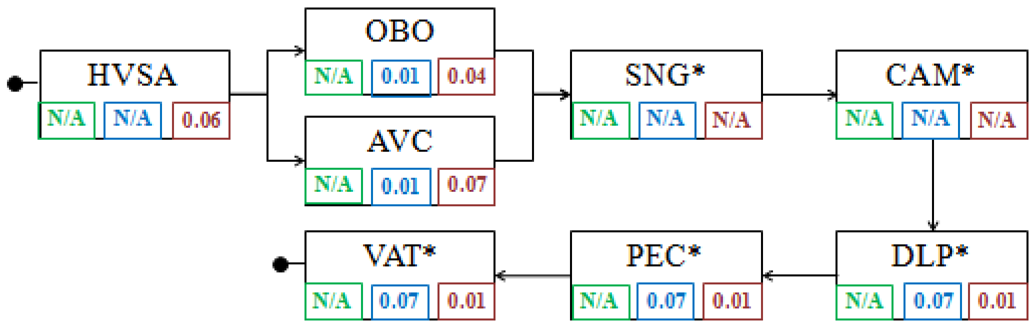
a) Whole system



b) Bus sub-systems



c) Payload sub-systems



*Sub-missions are arranged as they were executed or are planned to be executed on orbit

Figure 4-8. HORYU-IV reliability block diagrams of a) whole system, b) bus sub-systems, and c) payload sub-systems

Table 4-1. HORYU-IV reliability after the different AIT phases for the whole system, bus sub-systems, and payload sub-systems

	EM1	EM2	FM
Testing time [h]	432	728	880
Whole system reliability	0.31	0.18	0.35
Bus sub-systems reliability	0.37	0.25	0.51
Payload sub-systems reliability	N/A	0.79	0.81

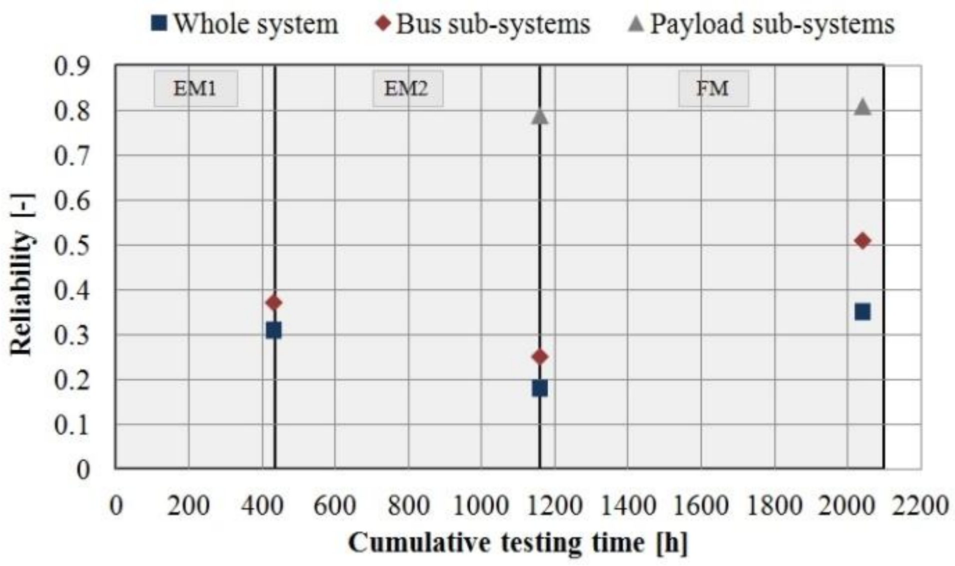


Figure 4-9. HORYU-IV reliability at the end of the different testing phases

From Figure 4-9 and Table 4-1, it can be observed that the reliability of the whole system drops by 42% from EM1 and EM2 phases. This is driven by the bus sub-systems, and more specifically the structure. At EM2, structure and electronic related parts were assembled for the first time, which resulted in many structure driven interface failures discovery. This further demonstrates the importance of interfaces verifications not only between electrical interfaces, but also between mechanical and electronic interfaces. Overall, the whole system and bus sub-systems reliability was respectively improved by 13% and 38% between EM1 and FM AIT phases. For the payload sub-systems, the reliability was improved by 2.5% between EM2 and FM AIT phases. This points out that sufficient on-ground AIT processes can help improve system overall reliability prior to launch.

4.1.4. Failures taxonomy

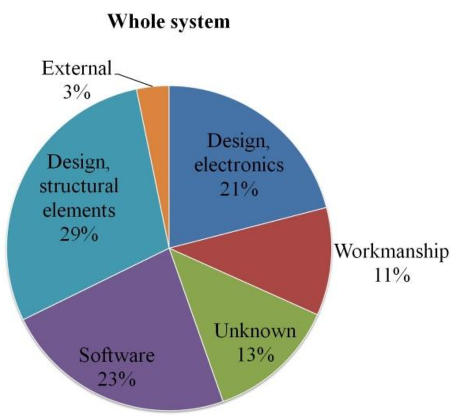
A taxonomy of the different failures observed during HORYU-IV project development is proposed in this section. In total, six categories were identified and Figure 4-10 present the results for the overall project as well as for each development phase.

For the overall project (Figure 4-10a-b), about 70% of the failures were driven within similar proportions by electronics design, structural elements design, and software failures. However, when looking at the project different stages of development, it appears that EM1 was driven at 43% by failures in the design of electronics (Figure 4-10c-d). Then, there was a shift in the failures origin and EM2 was driven at 71% by failures in the design of the structural elements since this development phase corresponds to the first time the PCBs and different elements were assembled together to realize the satellite EM version (Figure 4-10e-f). For FM, there was a new shift in the failures type and software failures represented 39% of the total number of recorded failure for this phase (Figure 4-10g-h). This is logical since after the assembly of the FM satellite it is very risky to modify hardware and HORYU-IV philosophy was to ensure hardware functionalities prior to FM assembly, whereas software could be modified until the satellite delivery.

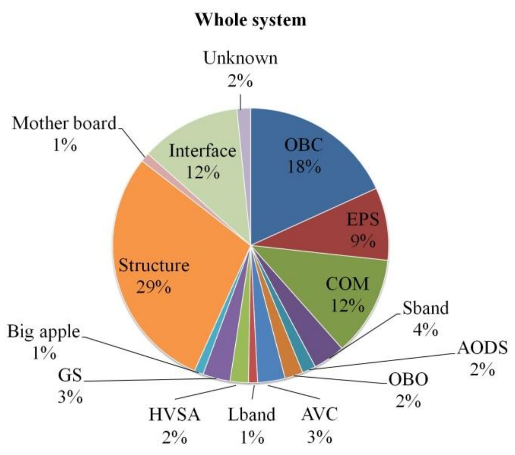
Moreover, it can be noted that for the whole system, workmanship-type failures accounted for about 10%. Those were especially discovered during EM1 and FM phases. To prevent workmanship-type failures, the satellite project should not rely on one's knowledge, but have clearly defined checklists and proper documentation. Yet, the reader should note that only critical documentation should be identified to limit unnecessary documents, spare time, prevent "death by administration", and favor efficient knowledge transfer.

It is important to note that, though payloads or sub-systems other than OBC, EPS, or COM seem to be minor failures driver, the failures that emerged during their AIT processes were nonetheless critical and their discovery undoubtedly contributed to the achievement of the satellite minimum success. Therefore, AIT processes should not be overlooked and overall it is my opinion that when lean satellite developers have to chose between different AIT scenarios, they should chose the scenario that is most suitable for achieving the satellite minimum success even if that choice might mean to put aside or even giving up on some other functionalities.

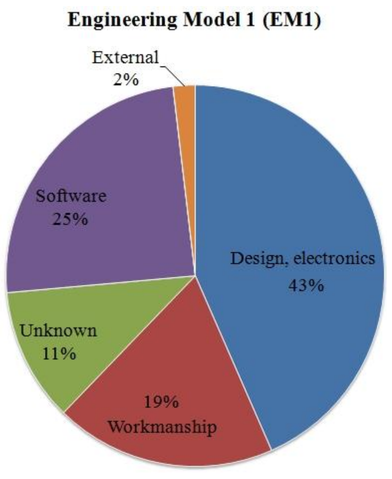
a)



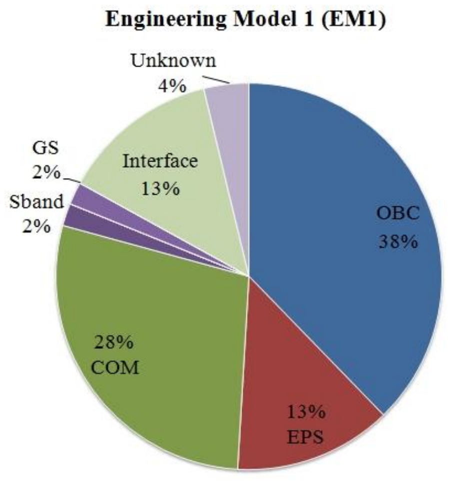
b)



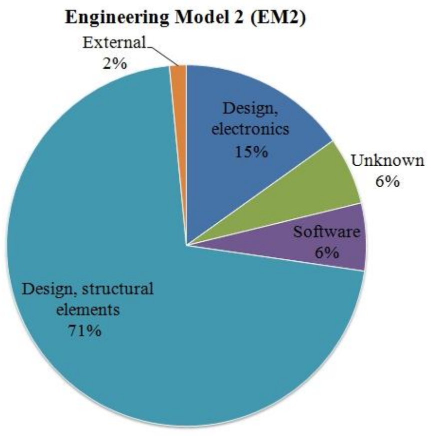
c)



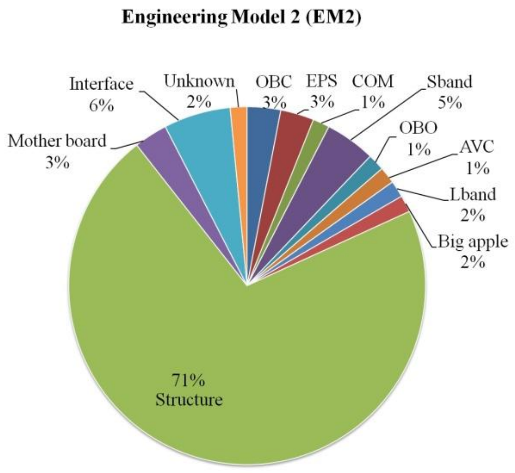
d)



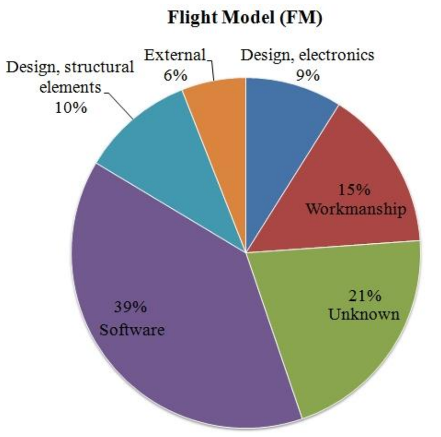
e)



f)



g)



h)

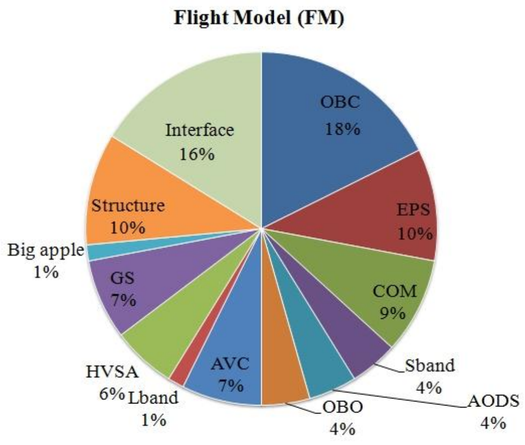


Figure 4-10. HORYU-IV failures taxonomy for a-b) overall system, c-d) EM1, e-f) EM2, and g-h) FM

4.2. Simulations results and discussion

4.2.1. Time considerations

Simulations results examples for the seven AIT processes with initial testing time, Ti, of 20h, 50h, 100h, and 200h are presented in Figure 4-11. From these results, it can be observed that most of failures are discovered at the early stage of testing for each considered AIT process. This is in agreement with the trend that was observed with the experimental results (Figure 4-1).

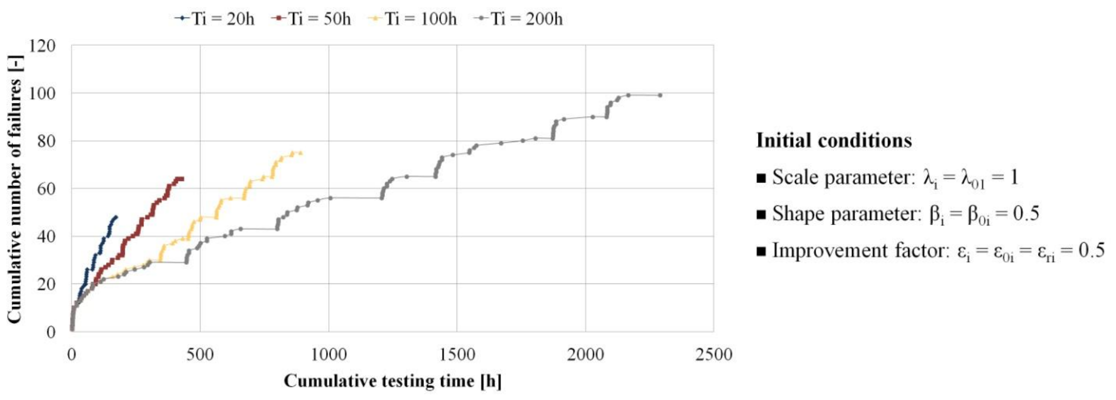


Figure 4-11. Example of simulations results comparison for the seven AIT processes with different initial testing time, Ti=20h, 50h, 100h, or 200h

It appears that for any decided initial testing time, the actual average total testing due to failures discovery will be about 11 times longer for seven AIT processes. For example, for a decided initial testing time of 200h, the actual total testing time is 2061h. With the assumption that testing is carried out for 8 hours per business day, i.e. 22 days are considered in a month, these 2061 hours of testing can be translated into a minimum of 12 months of AIT processes. Considering lean satellite programs are typically developed over 2 years, this means half of the program should be dedicated to AIT processes. Though this might be feasible in some lean satellite programs, this might be too constraining in others and the decision on initial testing time based on a trade-off with desired reliability after launch would be more suitable as shown in section 4.2.2.

Results presented in Figure 4-11 only take into account failure modes and therefore only show failure modes impact on AIT phases. When failures criticality is also taken into account, the impact of failures discovery on the overall project schedule can be studied. This is shown in Figure 4-12.

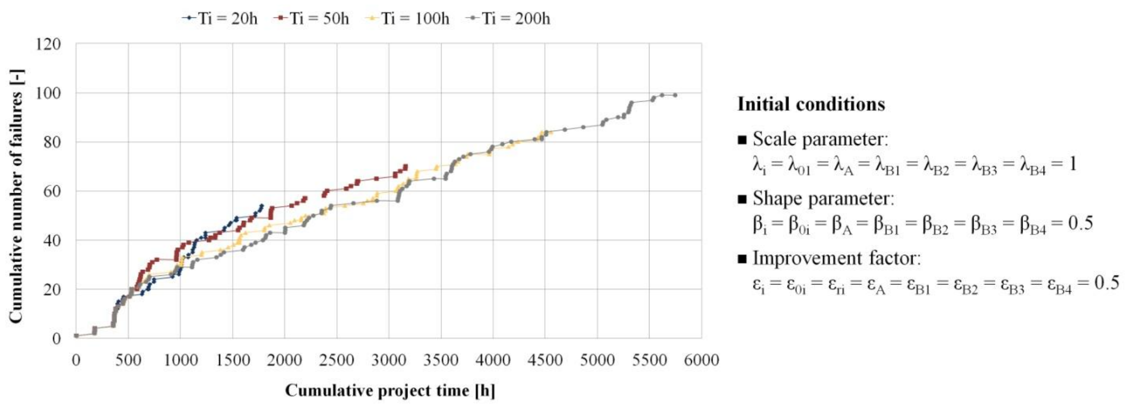
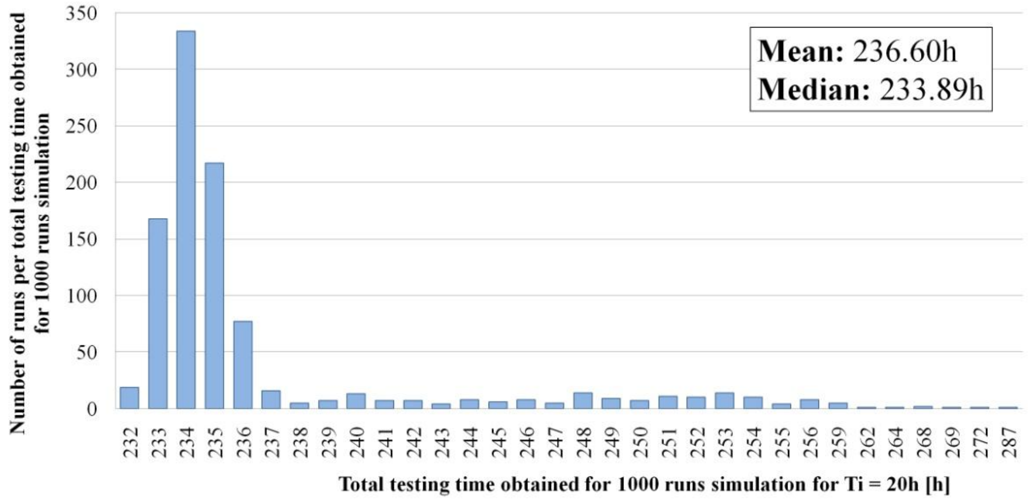


Figure 4-12. Example of simulations results comparison for project total time with different initial testing time, Ti=20h, 50h, 100h, or 200h and with 7 AIT processes

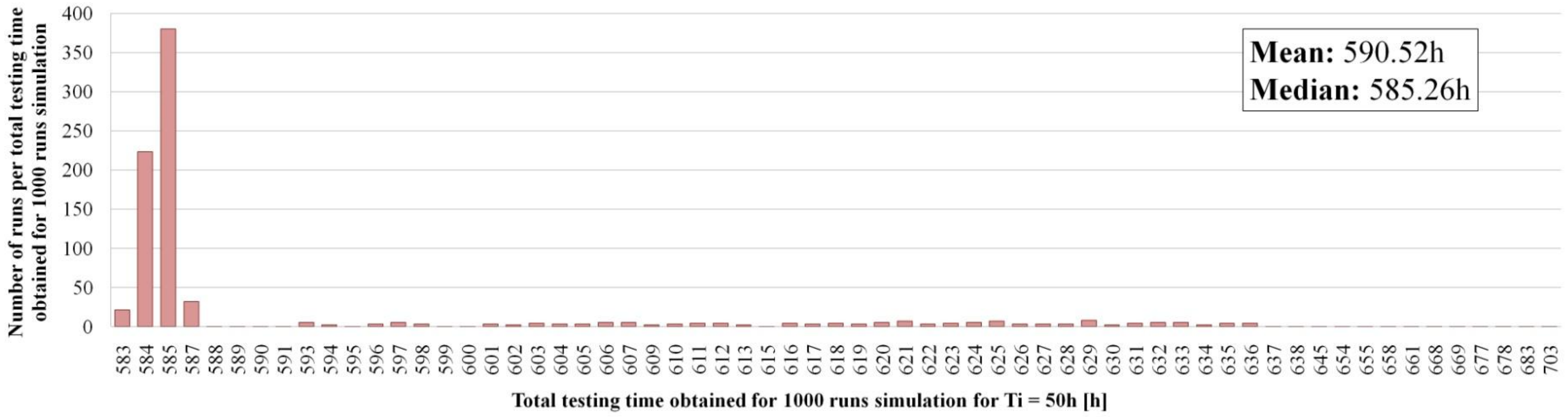
From Figure 4-12, it appears that independently of the initial AIT processes time chosen, the number of total failures discovered at the end of the lean satellite project is of the same order. If the AIT processes time is increased by 10 times from 20h to 200h, only 1.7 more failures are discovered on average. However, since the failures reparation time is taken into account, the total project time becomes on average 2.5 times longer, which can be very constraining for some lean satellite programs.

In Figure 4-13, the distribution of the total testing times, TtAIT, obtained for each of the 1000 runs is presented for the different initial testing times, T_i. Details of the numerical values for the different data sets are given in Appendix C.

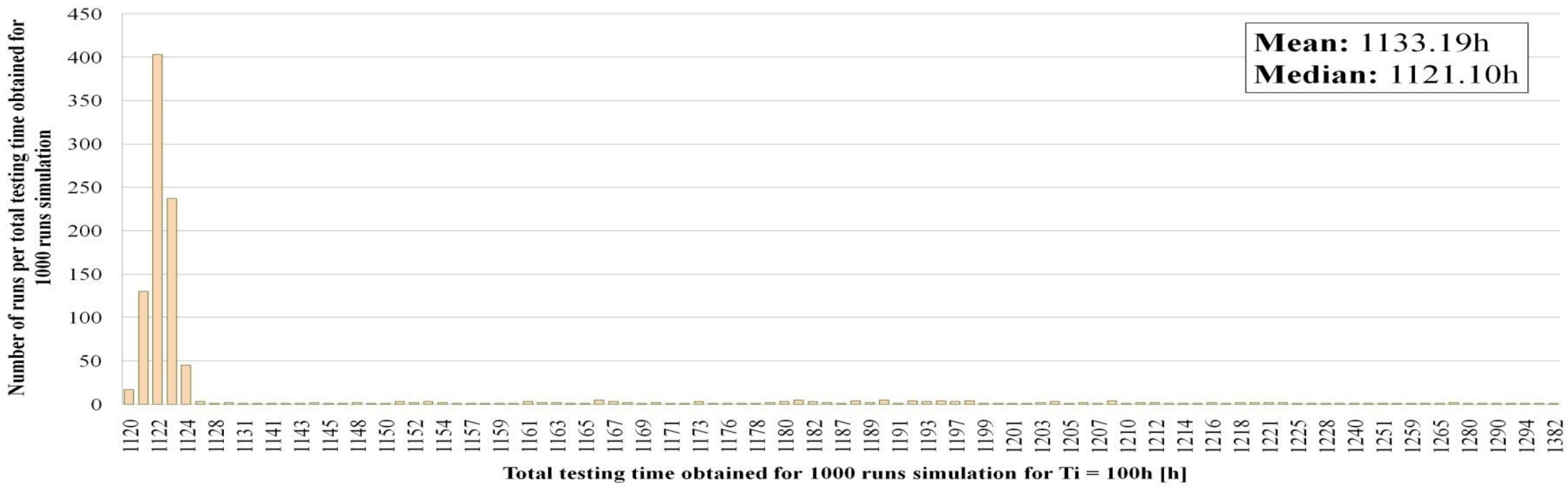
a)



b)


Mean: 590.52h
Median: 585.26h
Number of runs per total testing time obtained for 1000 runs simulation
Total testing time obtained for 1000 runs simulation for Ti = 50h [h]

c)


Mean: 1133.19h
Median: 1121.10h
Number of runs per total testing time obtained for 1000 runs simulation
Total testing time obtained for 1000 runs simulation for Ti = 100h [h]

d)

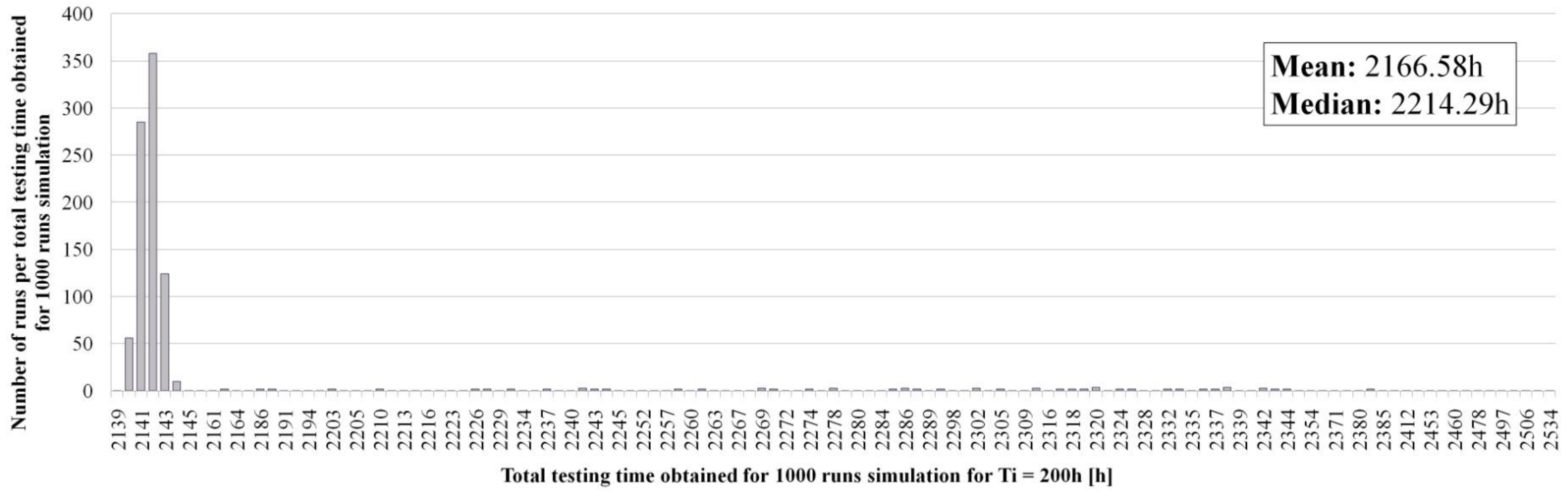


Figure 4-13. Total testing time distribution over the 1000 runs simulation for a) Ti = 20h, b) Ti = 50h, c) Ti = 100h, and d) Ti = 200h

4.2.2. Reliability consideration

For the reliability after launch, each AIT process was run 1000 times and the resulting average values of λ_{0last}, λ_{ilast}, β_{ilast}, β_{0last}, θr_{last}, $TAIT_i$, and TtAIT were used for its calculation. The results are presented in Figure 4-14 and the evolution of the reliability for different times after launch depending on the total AIT processes time is presented in Figure 4-15.

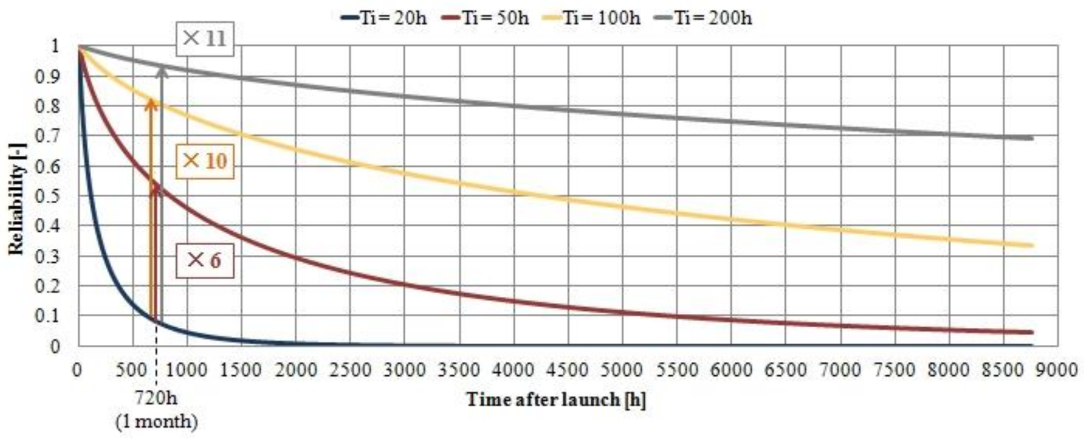


Figure 4-14. Comparison of reliability simulations results from 1000 simulation runs for each of the seven AIT processes with different initial testing times, Ti

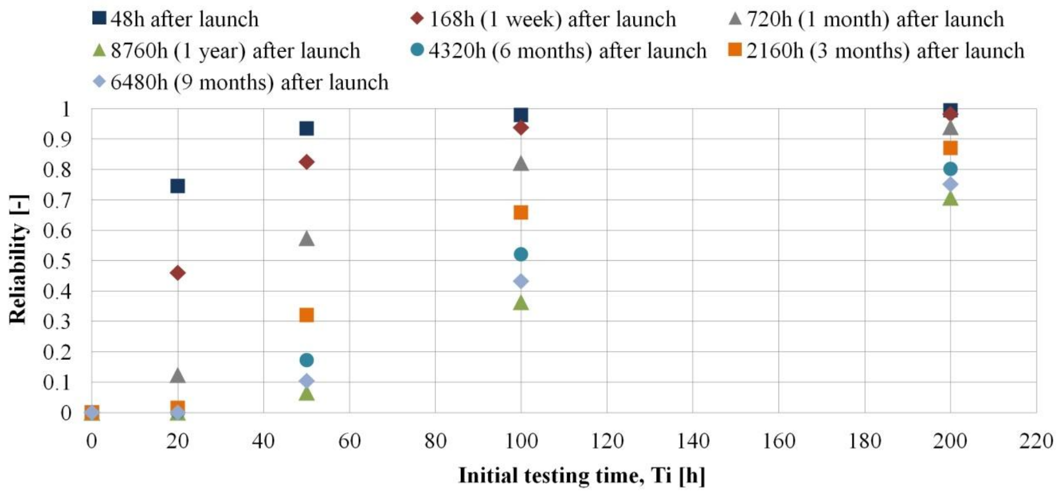


Figure 4-15. Reliability after launch from 1000 simulation runs for different times against initial testing time, Ti = 20h, 50h, 100h, or 200h

The simulation results of reliability after launch show that when the initial AIT processes time is increased by 10 times, the reliability at one month after launch can be improved by 11 times. Similarly, when the initial time is increased by 5 times and 2.5 times, the reliability at 1 month after launch is respectively improved by 10 times and 6 times. From these data combined with Figure 4-11 and Figure 4-12 data, it is clear that increasing AIT processes time is beneficial to improve reliability of the satellite after launch. Yet, a too large increase of the initial AIT processes time can impede the project overall development time without clearly improving the system from a reliability standpoint as shown in Figure 4-15.

These observations further point out that for a lean satellite program to be able to achieve a minimum desired reliability for a certain time after launch, a resulting minimum time should be spent for AIT processes. Yet, drastically increasing the initial AIT processes time will not drastically increase the reliability of the system. Each lean satellite program developer should find the best trade-off between cost, development time, and reliability depending on the considered program resources.

It should be noted that only the qualitative relationship between the reliability and the initial testing time, T_i, is of interest and the quantitative values of the reliability should be taken with caution. The shape and scale parameters, β_i and λ_i, used to calculate the reliability are obtained from the AIT processes and much more stress is put during on-ground testing than the satellite will endure during its orbital lifetime, thus in actual case, the scale parameter, λ, after launch should be smaller than the final scale parameter obtained at the end of the seven AIT processes and so should be the shape parameter, β.

Numerical values of the total number of failures discovered, total testing time, and reliability at different times after launch for the different initial testing times are given in Table 4-2.

Table 4-2. Simulations results summary from 1000 simulation runs and for different initial testing times, T_i

Initial testing time, T_i [h], per AIT process	20	50	100	200
Average of total number of failures discovered for the seven AIT processes combined	57.8	75.4	87.6	100.1
Average of total testing time [h] for the seven AIT processes combined	236.60	590.52	1133.19	2166.58
Average reliability at 48h after launch	0.716	0.935	0.980	0.994
Average reliability at 168h (1 week) after launch	0.415	0.825	0.939	0.982
Average reliability at 720h (1 month) after launch	0.096	0.569	0.823	0.940
Average reliability at 2160h (3 months) after launch	0.009	0.313	0.659	0.871
Average reliability at 4320h (6 months) after launch	8.13×10^{-4}	0.165	0.521	0.803
Average reliability at 6480h (9 months) after launch	1.08×10^{-4}	0.098	0.431	0.752
Average reliability at 8760h (1 year) after launch	1.65×10^{-5}	0.060	0.361	0.708
Maximum time on-orbit for R = 0.5 [h]	121	979	4760	26852
Maximum time on-orbit for R = 0.6 [h]	81	625	2955	16401
Maximum time on-orbit for R = 0.7 [h]	52	377	1712	9221
Maximum time on-orbit for R = 0.8 [h]	30	202	867	4429
Maximum time on-orbit for R = 0.9 [h]	13	81	319	1473
Maximum time on-orbit for R = 0.95 [h]	6	36	134	567

4.2.3. Cost considerations

In addition to time and reliability, cost is another important factor to take into account for lean satellite programs and based on HORYU-IV experience, cost drivers per sub-system were identified (Table 4-3).

Table 4-3. HORYU-IV average cost per main sub-systems components

Sub-system	Component	Average cost/unit [JPY][a]
Structure	Satellite part[b]	20,000
	Screws, washers, etc	200
	Surface treatment	15,000
Mother board	Solar cells type A[c]	126,000[d]
	Batteries[c]	6,000/pack (1 pack = 6 cells)
	PCB[e]	240,000
	Transmitter	405,000
	Receiver	262,000
	Consulting	723,600/month
GS	S-band ground station	26,000,000[f]
	Consulting	70,000/month
	L-band ground station	3,990,000[f]
S-band/SNG	Transmitter	1,000,000
	Patch antenna	50,000
	PCB[e]	120,000
Thermal	Thermal sensor	13,000
	Sheet heater	12,000
	Black paint	25,000
L-band	Patch antenna	220,000
	Transmitter	500,000
HVSA	Solar cells type B[c]	100,000/panel[g]
	Solar cells type C	200,000/panel[h]
	PCB[e]	90,000
OBO	PCB[e]	150,000
AVC	Camera[i]	80,000

AVC	PCB[e]	200,000
AODS/ CAM	PCB[e]	180,000
	Camera[i]	20,000
	GPS receiver	440,000
	GPS antenna[j]	45,000
	Power divider	45,000
	Gyro sensor	15,000
	Sun sensor[k]	4,000
	Hysteresis damper	10,000
Big apple	PCB[e]	180,000
Interfaces	RTV	100,000/100g
	Harness, connectors	10,000
Others	Air table	220,000
	Rate table	1,350,000
	Software license	570,000/year
	Thermal vacuum test (LN2)	300,000/month
	Satellite transportation[l]	335,000/1 way
	Satellite mock-up	410,000

[a]Values are rounded

[b]Average cost for the manufacturing of 1 part of HORYU-IV structure, not the whole satellite structure

[c]Includes transportation, additional work such as assembly of different elements, etc

[d]There are 34 cells mounted on the satellite

[e]Includes electronic components purchase, mounting, soldering, and PCB manufacturing

[f]Total development cost (did not exist at Kyutech prior to HORYU-IV project)

[g]There are five panels mounted on the satellite

[h]There are two panels mounted on the satellite

[i]There are two cameras mounted on the satellite

[j]There are two GPS antennas mounted on the satellite

[k]There are six sun sensors mounted on the satellite

[l]Transportation by truck, 1 way ≈ 1000km

From this evaluation, induced cost depending on the AIT process considered was estimated (Table 4-4). From this, it can be seen that, as expected, the cost increases as the satellite goes through the different AIT processes since more and more elements are involved. Overall, the development cost induced at the end of the 7-th AIT process is nearly two times more important than the cost induced for the 1-st AIT process.

Table 4-4. Minimum cost induced/AIT process based on HORYU-IV development cost

AIT	Min. induced cost [MJPY]/Ti		Cost drivers
1) Electrical performance verifications (flat satellite)	20h	3.728	- Batteries - Electronic components[a] - Transmitters - Receiver - Consulting (mother board) - Cameras
	50h	4.490	
	100h	5.476	
	200h	7.073	
2) Structure assembly	20h	10.633	- Structure - Solar cells - Batteries - Transmitters - Receiver - Antennas - S-band/SNG - Thermal - L-band - HVSA - OBO - AVC - AODS - Big apple - Interfaces
	50h		
	100h		
	200h		
3) Electrical performance verifications (assembled satellite)	20h	11.941	- Same as above - Consulting[b] (mother board)
	50h	12.446	
	100h	12.960	
	200h	13.685	

4) Thermal vacuum test	20h	10.814	- Same as above (no consulting) - LN2
	50h	10.884	
4) Thermal vacuum test	100h	10.947	
	200h	11.049	
5) Plasma test	20h	11.941	- Same as 3)
	50h	12.446	
	100h	12.960	
	200h	13.685	
6) Software performance verifications	20h	13.013	- Same as 3) - GS[b]
	50h	13.848	
	100h	14.772	
	200h	16.113	
7) End-to-end missions simulation test	20h	14.293	- Same as 6)
	50h	15.625	
	100h	17.043	
	200h	19.078	

[a]Estimated to represent half the PCB cost

[b]Consulting and total S-band and L-band ground stations cost were evaluated for 1h

Though HORYU-IV was a university based satellite project, it was a semi-professional project involving engineers and researchers. The number of the professionals involved during the project depending on the different AIT processes and their corresponding contributing cost for the project are given in Table 4-5.

Table 4-5. Induced personnel cost for each AIT process

AIT	Number of professionals		Induced personnel project cost [MJPY]/Ti	
	Engineer (1500JPY/h)	Researcher (2500JPY/h)		
1) Electrical performance verifications (flat satellite)	1	1	20h	1.338
			50h	2.080
			100h	3.039
			200h	4.593

2) Structure assembly	2	0	20h	0.973
			50h	1.378
2) Structure assembly	2	0	100h	1.705
			200h	2.275
3) Electrical performance verifications (assembled satellite)	1	1	20h	1.273
			50h	1.764
			100h	2.265
			200h	2.970
4) Thermal vacuum test	1	1	20h	1.273
			50h	1.766
			100h	2.213
			200h	2.930
5) Plasma test	0	1	20h	0.805
			50h	1.107
			100h	1.383
			200h	1.825
6) Software performance verifications	0	1	20h	0.794
			50h	1.072
			100h	1.381
			200h	1.828
7) End-to-end missions simulation test	0	1	20h	0.779
			50h	1.081
			100h	1.382
			200h	1.803

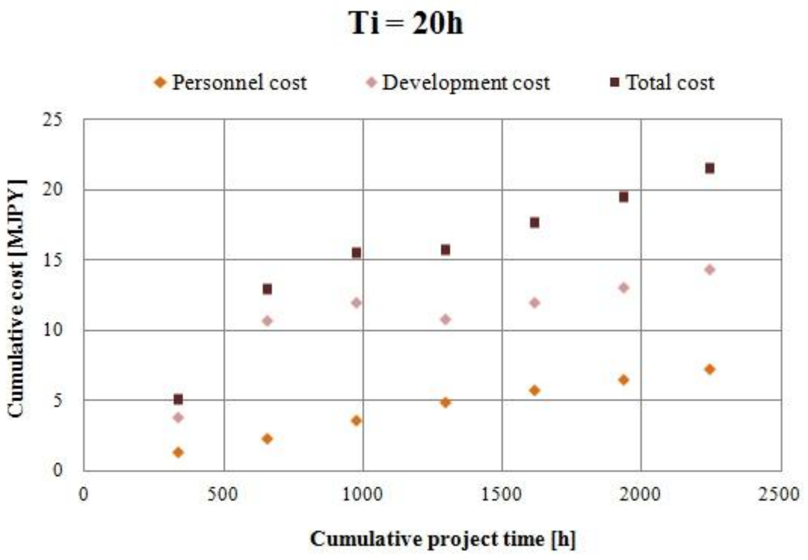
In Figure 4-16, plots of cumulative costs against the cumulative project time are shown for the different initial testing times, T_i. The development cost and personnel cost plots respectively correspond to the costs as presented in Table 4-4 and Table 4-5. The total cost plot corresponds to the sum of the development and personnel costs. As expected and aforementioned, the total cost from the 1-st AIT process to the 7-th AIT process increases. For any of the T_i considered, the project cost at the end of the 7-th AIT process is three times the cost at the end of the 1-st AIT process. Moreover, if the initial testing time, T_i, is increased by 10 times from 20h to 200h, the project overall total cost will be increased by more than 80%. However, if T_i is increases by 2.5, from 20h to 50h, the overall total cost will only be increased by 24%.

Another observation is that for high initial AIT processes time, personnel cost becomes the driver of the project overall cost. For university-based satellite programs, where the main workforce is students, this might not be a problem. However, for commercial entities developing lean satellites, the personnel cost associated with testing might lead the project toward cost overruns and critical tests might be overlooked to contain the testing cost overruns. This is not a desirable outcome and one option to prevent personnel cost overruns while performing sufficient testing could be to develop smart automated testing for different development phases.

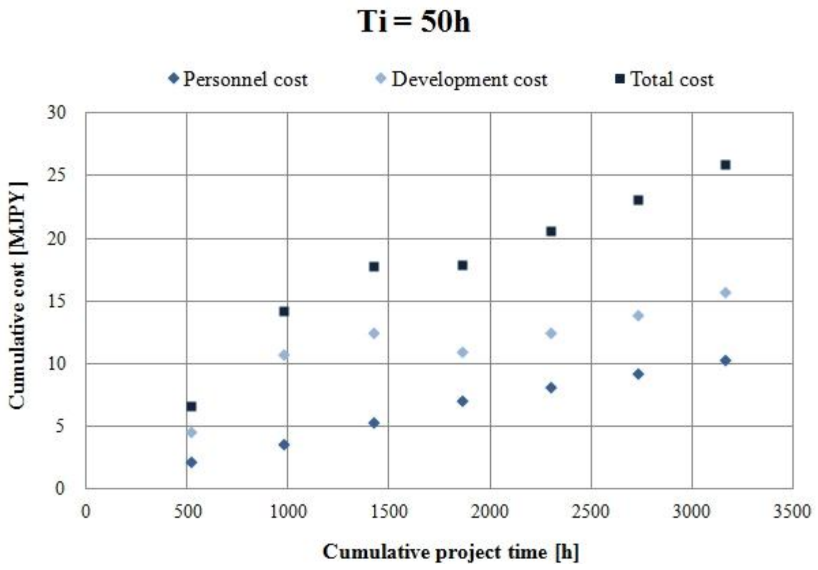
Overall, the findings point out that AIT processes should be performed optimally and efficiently not only to prevent lean satellite program overrunning, but also unnecessary cost.

It should be noted, that as compared to a lean satellite developed by a private company or space agency, the number of engineers and researchers involved in HORYU-IV project is minimum and results presented are for illustration only. More data should be collected on other types of lean satellite projects before making any final conclusions. Moreover, the hourly cost evaluation is somewhat low because junior engineers and researchers were employed.

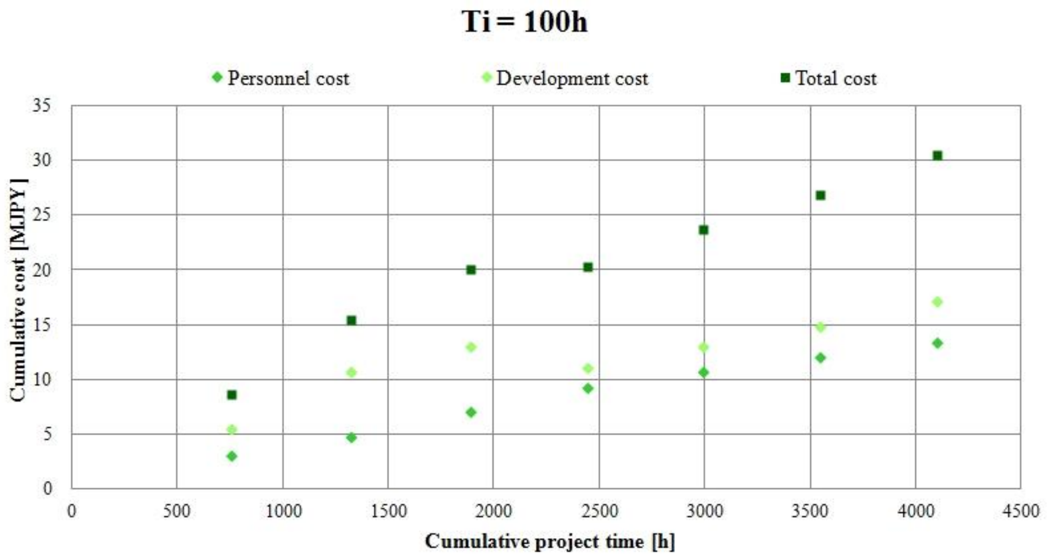
a)


Ti = 20h
◆ Personnel cost
◆ Development cost
■ Total cost
Cumulative cost [MJPY]
Cumulative project time [h]
0
5
10
15
20
25
0
500
1000
1500
2000
2500

b)


Ti = 50h
◆ Personnel cost
◆ Development cost
■ Total cost
Cumulative cost [MJPY]
Cumulative project time [h]
0
5
10
15
20
25
30
0
500
1000
1500
2000
2500
3000
3500

c)


Ti = 100h
◆ Personnel cost
◆ Development cost
■ Total cost
Cumulative cost [MJPY]
Cumulative project time [h]
0
5
10
15
20
25
30
35
0
500
1000
1500
2000
2500
3000
3500
4000
4500

d)

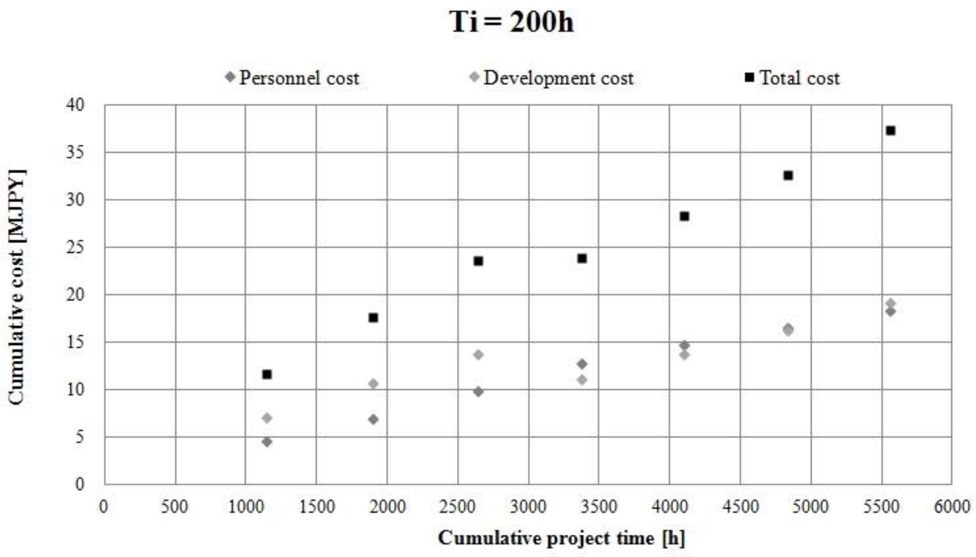


Figure 4-16. Plots of the cumulative cost against the cumulative project time for a) Ti = 20h, b) Ti = 50h, c) Ti = 100h, and d) Ti = 200h

CHAPTER 5 – Research Outlook

In the previous chapters, multi-criteria were assessed through experimental data and computations. From the assessment outcomes, different recommendations were established for the optimization of lean satellite programs managerial decision-making. However, there are limitations in the current research and these are addressed in the first part of this chapter along with their countermeasures. In the second part of this chapter, it is shown how the results obtained can be used, and be useful, in a professional manner through the proposal of a practical application.

5.1. Current work limitations and countermeasures

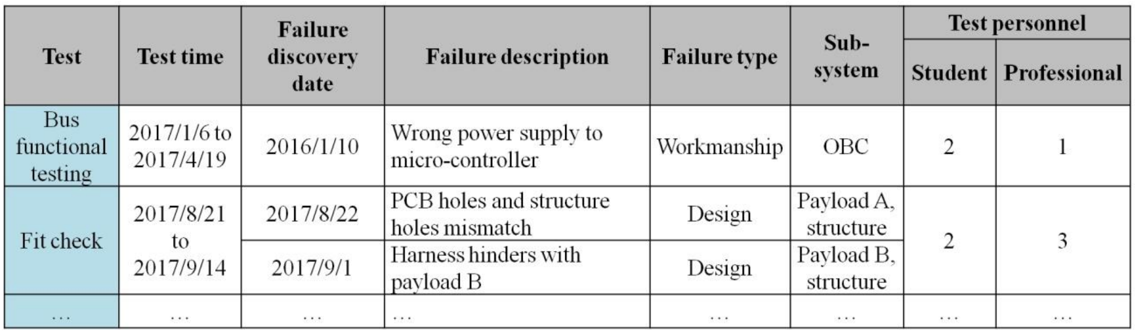
Within this research time frame, data from only one university based lean satellite program could be used and analyzed. This is a good starting point to draft the trends lean satellite programs might follow, but this is insufficient to establish general conclusions for any lean satellite program considered. There is therefore a need for inputs from other lean satellite programs. Those additional inputs should not only come from university based lean satellite programs, but also from private and governmental entities from all over the world to take into account the diversity of development philosophies.

As a countermeasure to the lack of case studies in this research current form, I propose to lean satellite programs managers to fill in during the AIT phases a simple and easy to use “failures track sheet” gathering data on:

- AIT processes denomination,
- AIT processes dates,

- failure discovery date,
- failure description,
- failure type,
- sub-system(s) associated to failure,
- number of people involved in AIT processes.

The entries listed above are non-exhaustive and should be refined according to the users' needs. An example of a "failures track sheet" is given in Figure 5-1 and in Appendix D, the actual failures information gathered during HORYU-IV project AIT phases are presented.



Test	Test time	Failure discovery date	Failure description	Failure type	Sub-system	Test personnel	
						Student	Professional
Bus functional testing	2017/1/6 to 2017/4/19	2016/1/10	Wrong power supply to micro-controller	Workmanship	OBC	2	1
Fit check	2017/8/21 to 2017/9/14	2017/8/22	PCB holes and structure holes mismatch	Design	Payload A, structure	2	3
		2017/9/1	Harness hinders with payload B	Design	Payload B, structure		
…	…	…	…	…	…	…	…

Figure 5-1. Example of "failures track sheet" for lean satellite programs

As a complement to the "failures track sheet", it would also be of interest to establish a database of the lessons learned proper to the "LeanSat community". NASA established a lessons learned database [60], but it only concerns NASA related projects and it cannot be updated by personnel outside the space agency.

For lean satellite programs, I believe the database should be open and accessible to any person registered to the "LeanSat community". Upon registration, the registered person could add lesson(s) learned related to his/her project(s) and make modifications to his/her previous entries. Regarding the database organization, entries could be divided depending on the entity the lean satellite program is carried on: private, government, or university. The entries could also be divided depending on the sub-system(s) associated with the lessons learned.

At the beginning, for the framework definition of the practical establishment of the “LeanSat” database, it could be part of the continuous work of the IAA Study Group 4.18 members. Then, the database organization and day-to-day management could be taken over by a private entity or individual.

An example of lessons learned extracted from the HORYU-IV project is presented in Appendix E.

Through the data compilation from the “failures track sheets” and the establishment of an open “LeanSat” lessons learned database, the trends presented in this research could be refined and the simulations could be improved from a qualitative and quantitative standpoint.

5.2. Proposal for a practical professional use of the research outcomes

From the refinement of the trends and simulations improvement, a model dedicated to lean satellite programs and integrating cost, schedule, and reliability could be developed.

Different tools are already available to satellite programs managers. Some are dedicated to reliability engineering [61] or cost estimation [62] and others can be used as assistant for scheduling [63, 64]. However, none of them shows the interdependence between those parameters. Moreover, their approach is mostly based on data from traditional satellites experiences and is therefore not well-suited for lean satellite programs.

In the last 10 years, a few researches focused on developing risk management tools dedicated to lean satellites, but due to the scarce number of information about such programs, their development and use are still at their infancy stage. In 2007, Deems [65] tackled the importance of risk management and how it can be more efficiently utilized within university based satellite programs through the establishment of master logic diagrams to identify failure modes. However, as it is pointed out by the author, there are no quantitative risk assessments. Moreover, the interdependence between risk assessment, cost, schedule, or other parameters is not evaluated. In 2015, these two main drawbacks were tackled by Brumbaugh Gamble [66], who developed two risk management tools: CubeSat Risk Analysis and CubeSat Decision Advisor. With these two tools, mission risks and their associated mitigation techniques can be evaluated in terms of cost, personnel, and time resources. However, these tools are based on only 52 satellite projects limited to CubeSats and the tools outputs are the same regardless of whether a university, a private company, or a government satellite was considered.

There is a clear international need for an integrated management tool dedicated to lean satellites. Based on data collection from the “LeanSat community”, the work established by Brumbaugh Gamble could be a good starting point for such tool. First, data from more various satellite program types could be gathered and analyzed to tailor risk assessments specific to each satellite program type. Then based on this research’s outcomes, interdependence not only between cost and schedule, but also between AIT processes time and reliability could be quantitatively evaluated.

From the development of such an integrated management tool, we could expect to rectify a lean satellite program strategy during early development stages eventually leading to more reliable, cost-effective, and efficient satellites.

CHAPTER 6 – Conclusions and Recommendations

The research was led according to a two-step action plan. In the first step, actual experimental data from an actual satellite program were collected, analyzed, and interpreted. The main observations from the experimental data are listed below.

- Half to 2/3 of failures are discovered at an early stage of the assembly, integration, and testing processes.
- Integration of new sub-systems, electrical or mechanical, increases the failure rate and thorough interfaces verification and testing is therefore critical.
- Modifications of the assembly, integration, and testing strategy, such as putting emphasis on specific sub-systems, increase the failure rate.
- On average, a failure was discovered every 20h of the assembly, integration, and testing processes.
- From the Duane plot, HORYU-IV shape parameter was calculated to be 0.83, which indicates a positive reliability growth thanks to the assembly, integration, and testing processes the satellite underwent.
- From the establishment of reliability block diagrams, it was calculated that overall HORYU-IV system reliability was improved by 13% between EM1 and FM.
- Upon establishment of the failures taxonomy, it appeared that in its initial development phase, HORYU-IV failures were mainly due to mistakes in the electrical design and mainly driven by OBC and COM. Then, failures that emerged during the intermediate development phase were mainly due to mistakes in the mechanical design. Finally, in the last development phase,

failures were mainly due to software tuning, which was mostly driven by OBC. Overall, OBC and structure were identified as the main drivers for HORYU-IV infant mortality prior to launch.

- In the case of HORYU-IV, the total time spent for assembly, integration, and testing processes during EM and FM accounted for almost 2/3 of the total development time for these two phases combined.

The second step of the research action plan was to conduct simulations based on a well-established methodology to obtain trends on the interactions between assembly, integration, and testing processes time, project schedule, reliability, and cost. The main conclusions from the establishment of these trends are listed below.

- As it was observed with the experimental data, most of failures are discovered at an early stage of the assembly, integration, and testing processes.
- Reliability at 1 month after launch is improved from 6 to 11 times if the initial assembly, integration, and testing processes time is increased from 2.5 to 10 times.
- When failure criticality is taken into account and failures reparation time is thus accounted for, the overall project becomes 2.5 times longer when the initial assembly, integration, and testing processes time is increased by 10 times, from 20h to 200h.
- When the initial assembly, integration, and testing processes time is increased by 10 times, from 20h to 200h, only 1.7 times more failures are discovered.
- A long initial assembly, integration, and testing processes time can impede the project overall development time without clearly improving the system from a

reliability standpoint.

- For any of the initial assembly, integration, and testing processes times considered, the project overall cost at the end of the 7-th AIT process is three times the cost at the end of the 1-st AIT process.
- In HORYU-IV case, if the initial assembly, integration, and testing processes time is increased by 10 times from 20h to 200h, the project overall total cost is increased by more than 80%.
- In HORYU-IV case, if the initial assembly, integration, and testing processes time is increased by 2.5, from 20h to 50h, the overall total cost is increased by less than 25%.
- A drastic increase of the initial assembly, integration, and testing processes time does not result in a drastic increase of the system reliability after launch, but does drastically increase the project cost.

From the experimental and simulation data observations, recommendations as described below can be established.

1) Even for a lean program with limited resources, a minimum time should be spent on assembly, integration, and testing processes to discover failures that could drive satellite to its premature death on-orbit.
2) From a time standpoint, it is worthy for lean satellite developers to allocate sufficient testing time to prove the soundness of a sub-system individually prior to its integration with other sub-systems.
3) Interfaces between sub-systems should be verified as early as possible during the development phase.

4) For lean satellite reliability growth, space environment testing, especially thermal vacuum test, is critical to identify major failures that could only emerged when the satellite is exposed to simulated on-orbit conditions and should therefore be performed even for a small time period.
5) Testing of sub-systems critical to mission minimum success is necessary for the improvement of the satellite reliability.
6) Favoring local manufacturers can be advantageous from a time and financial standpoint.
7) When possible, long lead items should be ordered in sufficient quantity to guaranty a quick response in case of failure and prevent excess of non-valuable waiting time.
8) For the prevention of workmanship-type failures, the project should not rely on an individual’s knowledge. Instead, critical and proper documentation should be identified and established.
9) Assembly, integration, and testing processes should ensure functions critical to the success of the satellite’s minimum success criteria.
10) During assembly, integration, and testing processes, cost overruns due to personnel cost could be prevented by developing smart automated processes.
11) The most suitable testing strategy for achieving satellite minimum success should be adopted even though it might mean to give up on some other functions.
12) Overall, assembly, integration, and testing processes should be performed optimally and efficiently not only to prevent lean satellite program overrunning, but also unnecessary cost.

The research outcomes intend to be used as guidance by lean satellite developers and managers for better planning of assembly, integration, and testing processes and resources allocation in order to develop more reliable, and therefore more sustainable, lean satellite systems independently of the constraints on a given program.

Yet, it is important to note that the reader should not extract final quantitative conclusions from the presented work since the data collected concern one satellite project only. Further data from different lean satellite programs are necessary to complement and improve the presented work. Two proposals for collecting the data are the “failures track sheet” and the “LeanSat community” lessons learned database.

Finally, from the data collection of the various lean satellite programs developers, an integrated management tool including qualitative and quantitative assessment of criteria such as time, cost, reliability and their interdependence could be developed. The development and use of such tool will eventually foster more reliable, cost-effective, and efficient lean satellite programs.

REFERENCES

CHAPTER 1 - Introduction

[1] ScienceDirect website: http://www.sciencedirect.com/ (last accessed on October 11th, 2016)

[2] J.R. Wertz, Space Missions Communities in: J.R. Wertz, D.F. Everett and J.J. Puschell (Eds.), *Space Mission Engineering: The New SMAD*, Microcosm Inc., Hawthorne (2011), p.38

[3] Space Works, 2014 Nano/Microsatellite market assessment. http://www.sei.aero/eng/ (last accessed on October 13th, 2016)

[4] G.F. Dubos, J-F. Castet and J.H. Saleh, *Statistical reliability analysis of satellites by mass category: Does spacecraft size matter?*, Acta Astronautica 67 (2010), pp.584-595

CHAPTER 2 – Background and Literature

[5] D.C. Dale and G.P. Whitcomb, ESA Bulletin, *Small satellite missions in context of the ESA scientific programme*, http://www.esa.int/esapub/bulletin/bullet80/dale80.htm (last accessed on August 17, 2016)

[6] The CubeSat Program, Cal Poly SLO, *CubeSat Design Specifications Rev.13* (2014)

[7] C. Galeazzi, *PRIMA: a new, competitive small satellite platform*, Acta Astronautica 46 (2000), pp.379-388

[8] Y-K. Chang, K-L. Hwang and S-J. Kang, *SEDT (System Engineering Design Tool) development and its application to small satellite conceptual design*, Acta Astronautica 61 (2007), pp.676-690

[9] Kyushu Institute of Technology, Center for Nanosatellite Testing,

http://cent.ele.kyutech.ac.jp/ (last accessed on October 20, 2016)

[10] D. Isabel, M. Hardin and J. Underwood, Mars Climate Orbiter team finds likely cause of loss, https://mars.jpl.nasa.gov/msp98/news/mco990930.html, NASA JPL (1999) (last accessed on October 20, 2016)

[11] Keiko Chino, *Why did Hitomi end up as space junk?*, The Japan New by The Yomiuri Shimbun, Opinion & Analysis, October 15, 2016

[12] R. Lloyd, Metric mishap caused loss of NASA orbiter, http://edition.cnn.com/TECH/space/9909/30/mars.metric.02/index.html?_s=PM:TECH CNN Space, September 30, 1999 (last accessed on October 20, 2016)

[13] The Associated Press, JAXA loses touch with Hitomi/ASTRO-H telescope, http://www.cbc.ca/news/technology/astro-h-launch-1.3510591, CBCnews Technology & Science, March 29, 2016 (last accessed on October 20, 2016)

[14] S. Müncheberg, M. Krischke and N. Lemke, *Nanosatellites and micro systems technology – Capabilities, limitations and applications*, Acta Astronautica 39 (1997), pp.799-808

[15] R. Fleeter, *Management of small satellite programs – Lessons learned*, Acta Astronautica 32 (1994), pp.667-673

[16] Proc. of the Int. Workshop on Small-scale Satellite Standardization, Kitakyushu, Japan, November 17-19, 2014

[17] T. Ohno, *Toyota Production System - Beyond Large Scale Production*, Productivity Press (1988)

[18] J.P. Womack, D.T. Jones and D. Roos, *The Machine That Changed The World*, Harper Perennial (1991)

[19] Lean Aerospace Initiative, *The Lean Enterprise Model*, Massachusetts Institute of

Technology (2004)

[20] B. Oppenheim, D. Secor and J. Oehmen, I*NCOSE Lean Systems Engineering Working Group Charter*, INCOSE (2012)

[21] J.P. Womack and D.T. Jones, *Lean Thinking – Banish Waste and Create Wealth in Your Corporation*, Free Press (2003)

[22] E.M. Murman, *The Lean Aerospace Initiative – One Perspective of an Engineering Systems Undertaking*, Massachusetts Institute of Technology (2004)

[23] OneWeb, http://oneweb.world/ (last accessed on October 21, 2016)

[24] M. Safyan, *Overview of the Planet Labs Constellation of Earth Imaging Satellites – In Space to Help Life on Earth*, Planet Labs (2015)

[25] J. Boyle, *Working "Faster, Better, Cheaper": a Federal Research Agency in Transition*, Dissertation of Doctor of Philosophy in Human Development, Virginia Polytechnic Institute and State University (2002)

[26] D.A. Kring, USRA Center for Lunar Science and Exploration, http://www.lpi.usra.edu/exploration/multimedia/NASABudgetHistory.pdf (last accessed on October 26, 2016)

[27] NASA Office of Inspector General, *Faster, Better, Cheaper: Policy, Strategic Planning, and Human Resource Alignment*, Audit Report (March 13, 2001)

[28] S.M. Krimigis and G.E. Cameron, *John Hopkins APL paradigm in smallsat management*, Acta Astronautica 46 (1999), pp187-197

[29] T. Lafon, *JASON 1: lessons learned from the development and 1 year in orbit*, Acta Astronautica 56 (2005), pp.45-49

[30] H.E. McCurdy, *Learning from history: low-cost project innovation in the U.S. National Aeronautics and Space Administration*, Int. Journal of Project Management 31

(2013), pp.705-711

[31] E. Gill, P. Sundaramoorthy, J. Bouwmeester, B. Zandbergen and R. Reinhard, *Formation flying within a constellation of nano-satellites: the QB50 mission*, Acta Astronautica 82 (2013), pp.110-117

[32] F. Graziani, F. Piergentili and F. Santoni, A space standards application to university-class microsatellites: the UNISAT experience, Acta Astronautica 66 (2010), pp.1534-43

[33] HORYU program at Kyushu Institute of Technology, http://kitsat.ele.kyutech.ac.jp/index.html (last accessed on October 26, 2016)

[34] HODOYOSHI program at The University of Tokyo, http://park.itc.u-tokyo.ac.jp/nsat/hodo1_e.html (last accessed on October 26, 2016)

[35] ESA website, Fly Your Satellite! programme, http://www.esa.int/Education/CubeSats_-_Fly_Your_Satellite (last accessed on October 25, 2016)

[36] Satellite Design Contest Executive Committee, 24th Satellite Design Contest Application Guidebook, http://www.satcon.jp/en/doc/1_applicationguidebook.pdf (last accessed on October 21, 2016)

[37] Mission Idea Contest for Micro/Nano-satellite Utilization, http://www.spacemic.net/ (last accessed on October 25, 2016)

[38] T. Pham, H. Sims and J.S. Perez Cano, *Small Satellite Missions: Capabilities and Challenges in Space Communications*, Panel on SPACOMM (2015)

[39] D.A. Bearden, *A complexity-based risk assessment of low-cost planetary missions: when is a mission too fast and too cheap?*, Acta Astronautica 52 (2003), pp.371-379

[40] L.J. Paxton, *"Faster, better, cheaper" at NASA: lessons learned in managing and*

accepting risk, Acta Astronautica 61 (2007), pp.954-963

[41] National Aeronautics and Space Administration-orbital Debris Quarterly News, *Monthly number of objects in Earth orbit by object type*, Volume 20, Issues 1&2, April 2016, p.14

[42] M. Emanuelli, L. Bettiol, M. Grulich, J.E. Gramajo Gonzalez, J. Atchison, L. Leon Perez and J. Sotudeh, *Nanosatellites and their demand for changes in space policy*, Proc. of the 67th International Astronautical Congress (2016)

[43] G.E. Peterson, A.B. Jenkin, M.E. Sorge and J.P. McVey, *Implications of proposed small satellite constellations on space traffic management and long-term debris growth in near-Earth environment*, Proc. of the 67th International Astronautical Congress (2016)

[44] G.F. Dubos, J-F. Castet and J.H. Saleh, *Statistical reliability analysis of satellites by mass category: does spacecraft size matter?*, Acta Astronautica 67 (2010), pp.584-595

[45] M. Langer and J. Bouwmeester, *Reliability of CubeSats – Statistical data, developers' beliefs and the way forward*, Proc. of the 30th Annual AIAA/USU Conf. on Small Satellites (2016)

[46] J. Bouwmeester and J. Guo, *Survey of worldwide pico- and nanosatellite missions, distributions and subsystem technology*, Acta Astronautica 67 (2010), pp.854-862

[47] J-F. Castet and J.H. Saleh, *Satellite reliability, Statistical data analysis and modeling*, Journal of Spacecraft and Rockets 46 (2009)

[48] M. Tafazoli, *A study of on-orbit spacecraft failures*, Acta Astronautica 64 (2009), pp.195-205

[49] J. Guo, L. Monas and E. Gill, *Statistical analysis and modeling of small satellite reliability*, Acta Astronautica 98 (2014), pp.97-110

CHAPTER 3 - Methodology

[50] P. Faure, A. Tanaka and M. Cho, *Toward lean satellites reliability improvement using HORYU-IV project as case study*, Acta Astronautica, accepted for publication (2016), DOI: 10.1016/j.actaastro.2016.12.030

[51] P. Faure, A. Tanaka, M. Cho and HORYU-IV Team, T*oward the improvement of lean satellites reliability through testing – The HORYU-IV (AEGIS) Nano-satellite case study,* Proc. of the 67th International Astronautical Congress (2016)

[52] H. Fukuda, T. Shimizu, K. Toyoda and M. Cho, *Development of a miniature oscilloscope and current probe for measurement of arc current on-board HORYU-3 in low earth orbit*, Proc. of the 13th Spacecraft Charging Technology Conference (2014)

[53] T. Shimizu, H. Fukuda, K. Toyoda and M. Cho, D*evelopment of in-orbit high voltage experiment platform: HORYU-4*, Proc. of the 13th Spacecraft Charging Technology Conference (2014)

[54] P. Faure, HORYU-IV Team and M. Cho, *Preliminary overview of the arc event generator and investigation satellite (AEGIS) – HORYU-IV*, Proc. of the 58th Conference on Space and Science (2014)

[55] M. Cho and P. Faure, *Reliability growth of a nano-satellite through assembly, integration and testing*, Proc. of the 66th International Astronautical Congress (2015)

[56] M. Krasich, *Reliability prediction using flight experience – Weibull adjusted probability of survival, WAPS method,* Lecture at the Reliability Seminar, University of Maryland (1995)

[57] J-F. Castet and J.H. Saleh, *Satellite and satellite subsystems reliability: Statistical data analysis and modeling*, Reliability Engineering and System Safety 94 (2009), pp.1718-28

[58] L.H. Crow, *Reliability growth planning, analysis and management*, Annual Reliability and Maintainability Symposium (2011)

CHAPTER 4 – Results and Discussion

[59] M. Cho, *Reliability growth of small-scale satellites through testing: Monte Carlo simulation*, Proc. of the International Workshop on Small-scale Satellite Testing Standardization (2013)

CHAPTER 5 – Research Outlook

[60] NASA Lessons Learned system, https://llis.nasa.gov/ (last accessed January 6, 2017)

[61] ReliaSoft website, http://www.reliasoft.com/ (last accessed December 19, 2016)

[62] E. Mahr, A. Tu and A. Gupta, *Development of the Small Satellite Cost Model 2014 (SSCM14)*, IEEE Aerospace Conference (2016)

[63] GanttProject website, http://www.ganttproject.biz/ (last accessed December 19, 2016)

[64] OpenProj, https://sourceforge.net/projects/openproj/, Open source desktop project management (last accessed December 19, 2016)

[65] E. Deems, Risk Management of Student-Run Small Satellite Programs, Master of Sciences in Aeronautics and Astronautics dissertation, Massachussets Institute of Technology (2007)

[66] K. Brumbaugh Gamble, *A Software Tool Suite for Small Satellite Risk Management*, Doctor of Philosophy dissertation, The University of Texas (2015)

APPENDIX A – MATLAB Program for Testing Time and Project Time Estimations

```
%Note: unlike in the manuscript, the random number 'ηi1' and 'ηi2' are defined as 'randomi1' and 'randomi2' in the program

M = 1;
AIT = 7; %Define to number of AITs to take into consideration

A = 1;
B = 0.001118; %=0.01*lambdar*betar*T^((betar)-1)

while M<=AIT
N = 1;
Run = 1000; %Define number of simulation runs
T = 200; %Define initial testing time, Ti

lambda0 = A; %Last average lambda0 obtained after the N simulation runs
thetar = B; %Last average thetar obtained after the N simulation runs

matAllFail = [];
matAllFailA = [];
matAllFailB1 = [];
matAllFailB2 = [];
matAllFailB3 = [];
matAllFailB4 = [];
matAllFailSingle = [];
matAllFailCross = [];
matAllFailRand = [];
matLambda02 = [];
matLambda2 = [];
matLambdaA2 = [];
matLambdaB12 = [];
matLambdaB22 = [];
matLambdaB32 = [];
matLambdaB42 = [];
```

```
matProjectT = [];
matProjectTt = [];
matTAIT = [];
matThetar2 = [];
matTimeA = [];
matTimeB1 = [];
matTimeB2 = [];
matTimeB3 = [];
matTimeB4 = [];
matTotSingle = [];
matTtAIT = [];

testN = [];
testTstep = [];

while N<=Run

    lambdar = 1;
    betar = 0.5;

    if M == 1
        lambda0 = 1;
        thetar = 0.01*lambdar*betar*T^((betar)-1);
        TtAIT = 0;
    else
        lambda0 = A;
        thetar = B;
        TtAIT = C;
    end

    lambda = 1;
    lambdaA = 1;
    lambdaB1 = 1;
    lambdaB2 = 1;
    lambdaB3 = 1;
    lambdaB4 = 1;
```

```
beta0 = 0.5;
beta = 0.5;
betaA = 0.5;
betaB1 = 0.5;
betaB2 = 0.5;
betaB3 = 0.5;
betaB4 = 0.5;
cumFail = 0;
deltaTA = 0;
deltaTB1 = 0;
deltaTB2 = 0;
deltaTB3 = 0;
deltaTB4 = 0;
deltaTs = 0;
deltaTr = 0;
epsilon0 = 0.5;
epsilon = 0.5;
epsilonA = 0.5;
epsilonB1 = 0.5;
epsilonB2 = 0.5;
epsilonB3 = 0.5;
epsilonB4 = 0.5;
epsilonr = 0.5;
failA = 0;
failAITany = 0;
failAITonly = 0;
failB1 = 0;
failB2 = 0;
failB3 = 0;
failB4 = 0;
failRandom = 0;
Tstep = 0;
timeA1 = 0;
timeB11 = 0;
timeB21 = 0;
timeB31 = 0;
```

```
timeB41 = 0;

cumFail1 = [];
failAITany1 = [];
failAITonly1 = [];
failRandom1 = [];
matLambda01 = [];
matLambda1 = [];
matLambdaA1 = [];
matLambdaB11 = [];
matLambdaB21 = [];
matLambdaB31 = [];
matLambdaB41 = [];
matLambdar = [];
matThetar1 = [];
R1 = [];
rand1 = [];
rand2 = [];
rand3 = [];
saveDeltaTA = [];
saveDeltaTB1 = [];
saveDeltaTB2 = [];
saveDeltaTB3 = [];
saveDeltaTB4 = [];
saveDeltaTr = [];
saveDeltaTs = [];
saveF1 = [];
saveF2 = [];
saveF3 = [];
saveF4 = [];
saveF5 = [];
saveF6 = [];
saveFailA = [];
saveFailB1 = [];
saveFailB2 = [];
saveFailB3 = [];
```

```
saveFailB4 = [];
saveR = [];
saveRandom1 = [];
saveRandom21 = [];
saveRandom22 = [];
saveRandom23 = [];
saveT1 = [];
saveT2 = [];
saveT3 = [];
saveT4 = [];
saveTimeA1 = [];
saveTimeB11 = [];
saveTimeB21 = [];
saveTimeB31 = [];
saveTimeB41 = [];
saveTimeProj = [];
saveTtot1 = [];
saveTtot2 = [];
saveTtot3 = [];
saveTtot4 = [];
Tstep1 = [];

while Tstep<=T

    Rsingle = exp(-(lambda).*(Tstep+deltaTs+deltaTr).^(beta));
    Rcross =
exp(-(lambda0).*(Tstep+deltaTs+deltaTr+TtAIT).^(beta0)+(lambda0).*(TtAIT).^(b
eta0));
    Rrandom= exp(-(thetar)*(Tstep+deltaTs+deltaTr));
    R = Rsingle*Rcross*Rrandom;
    R1 = [R1,R];

    random1 = rand(1,1);
    rand1 = [rand1,random1];

    if random1>R
```

```
        saveT1 = [saveT1,Tstep];
        saveRandom1 = [saveRandom1,random1];
        saveR = [saveR,R];

        cumFail = cumFail + 1;
        cumFail1 = [cumFail1,cumFail];

        timeProj = Tstep + deltaTr + deltaTs + deltaTA + deltaTB1 + deltaTB2 + deltaTB3 + deltaTB4; %Cumulative time until last failure, not until test ends
        saveTimeProj = [saveTimeProj,timeProj];
        Ttot1 = Tstep + deltaTr + deltaTs;
        saveTtot1 = [saveTtot1,Ttot1];

        %To determine failure category
        F1 = ((lambda).*(beta).*(Ttot1).^((beta)-1))/((lambda).*(beta).*(Ttot1).^((beta)-1)+(lambda0).*(beta0).*(Ttot1+TtAIT).^((beta0)-1)+thetar);
        saveF1 = [saveF1,F1];
        F2 = ((lambda).*(beta).*(Ttot1).^((beta)-1)+(lambda0).*(beta0).*(Ttot1+TtAIT).^((beta0)-1))/((lambda).*(beta).*(Ttot1).^((beta)-1)+(lambda0).*(beta0).*(Ttot1+TtAIT).^((beta0)-1)+thetar);
        saveF2 = [saveF2,F2];

        random2 = rand(1,1);
        rand2 = [rand2,random2];

        %Failure common to AIT No.1 only (single-AIT failure mode)
        if random2<F1
            saveT2 = [saveT2,Tstep];
            saveRandom21 = [saveRandom21,random2];

            failAITonly = failAITonly + 1;
            failAITonly1 = [failAITonly1,failAITonly];

```

```
lambda = lambda*epsilon;
  matLambda1 = [matLambda1,lambda];

deltaTs = deltaTs + 1;
saveDeltaTs = [saveDeltaTs,deltaTs];

Ttot2 = Tstep + deltaTs + deltaTr;
saveTtot2 = [saveTtot2,Ttot2];

if Tstep == 0
        Tstep = 0;
else
        Tstep = Tstep - 1;
end

%To determine failure type (A, B1, B2, B3, or B4)
F3 = ((lambdaA).*(betaA).*(Ttot2).^((betaA)-1))/((lambdaA).*(betaA).*(Ttot2).^((betaA)-1)+(lambdaB1).*(betaB1).*(Ttot2).^((betaB1)-1)+(lambdaB2).*(betaB2).*(Ttot2).^((betaB2)-1)+(lambdaB3).*(betaB3).*(Ttot2).^((betaB3)-1)+(lambdaB4).*(betaB4).*(Ttot2).^((betaB4)-1));
saveF3 = [saveF3,F3];
F4 = ((lambdaA).*(betaA).*(Ttot2).^((betaA)-1)+(lambdaB1).*(betaB1).*(Ttot2).^((betaB1)-1))/((lambdaA).*(betaA).*(Ttot2).^((betaA)-1)+(lambdaB1).*(betaB1).*(Ttot2).^((betaB1)-1)+(lambdaB2).*(betaB2).*(Ttot2).^((betaB2)-1)+(lambdaB3).*(betaB3).*(Ttot2).^((betaB3)-1)+(lambdaB4).*(betaB4).*(Ttot2).^((betaB4)-1));
saveF4 = [saveF4,F4];
F5 = ((lambdaA).*(betaA).*(Ttot2).^((betaA)-1)+(lambdaB1).*(betaB1).*(Ttot2).^((betaB1)-1)+(lambdaB2).*(betaB2).*(Ttot2).^((betaB2)-1))/((lambdaA).*(betaA).*(Ttot2).^((betaA)-1)+(lambdaB1).*(betaB1).*(Ttot2).^((betaB1)-1)+(lambdaB2).*(betaB2).*(Ttot2).^((betaB2)-1)+(lambdaB3).*(betaB3).*(Ttot2).^((betaB3)-1)+(lambdaB4).*(betaB4).*(Ttot2).^((betaB4)-1));
saveF5 = [saveF5,F5];
F6 =
((lambdaA).*(betaA).*(Ttot2).^((betaA)-1)+(lambdaB1).*(betaB1).*(Ttot2).^((beta
B1)-1)+(lambdaB2).*(betaB2).*(Ttot2).^((betaB2)-1)+(lambdaB3).*(betaB3).*(Ttot
2).^((betaB3)-1))/((lambdaA).*(betaA).*(Ttot2).^((betaA)-1)+(lambdaB1).*(betaB1
).*(Ttot2).^((betaB1)-1)+(lambdaB2).*(betaB2).*(Ttot2).^((betaB2)-1)+(lambdaB3)
.*(betaB3).*(Ttot2).^((betaB3)-1)+(lambdaB4).*(betaB4).*(Ttot2).^((betaB4)-1));
                    saveF6 = [saveF6,F6];

                    random3 = rand(1,1);
                    rand3 = [rand3,random3];

                    %Type A single-AIT failure
                    if random3<F3
                        failA = failA + 1;
                        saveFailA = [saveFailA,failA];

                        lambdaA = lambdaA*epsilonA;
                        matLambdaA1 = [matLambdaA1,lambdaA];

                        timeA1 = timeA1;
                        saveTimeA1 = [saveTimeA1,timeA1];

                        deltaTA = deltaTA;
                        saveDeltaTA = [saveDeltaTA,deltaTA];

                    %Type B1 single-AIT failure
                    elseif F3<=random3 && random3<F4
                        failB1 = failB1 +1;
                        saveFailB1 = [saveFailB1,failB1];

                        lambdaB1 = lambdaB1*epsilonB1;
                        matLambdaB11 = [matLambdaB11,lambdaB1];

                        timeB11 = timeB11 + 1;
                        saveTimeB11 = [saveTimeB11,timeB11];

                        deltaTB1 = deltaTB1 + 1;
```

```
            saveDeltaTB1 = [saveDeltaTB1,deltaTB1];

        %Type B2 single-AIT failure
        elseif F4<=random3 && random3<F5
            failB2 = failB2 + 1;
            saveFailB2 = [saveFailB2,failB2];

            lambdaB2 = lambdaB2*epsilonB2;
            matLambdaB21 = [matLambdaB21,lambdaB2];

            timeB21 = timeB21 + 8;
            saveTimeB21 = [saveTimeB21,timeB21];

            deltaTB2 = deltaTB2 + 8;
            saveDeltaTB2 = [saveDeltaTB2,deltaTB2];

        %Type B3 single-AIT failure
        elseif F5<=random3 && random3<F6
            failB3 = failB3 + 1;
            saveFailB3 = [saveFailB3,failB3];

            lambdaB3 = lambdaB3*epsilonB3;
            matLambdaB31 = [matLambdaB31,lambdaB3];

            timeB31 = timeB31 + 40;
            saveTimeB31 = [saveTimeB31,timeB31];

            deltaTB3 = deltaTB3 + 40;
            saveDeltaTB3 = [saveDeltaTB3,deltaTB3];

        %Type B4 single-AIT failure
        elseif F6<=random3
            failB4 = failB4 +1;
            saveFailB4 = [saveFailB4,failB4];

            lambdaB4 = lambdaB4*epsilonB4;
```

```
                    matLambdaB41 = [matLambdaB41,lambdaB4];

                    timeB41 = timeB41 + 176;
                    saveTimeB41 = [saveTimeB41,timeB41];

                    deltaTB4 = deltaTB4 + 176;
                    saveDeltaTB4 = [saveDeltaTB4,deltaTB4];

                end

            %Failure common to any AIT (cross-AIT failure mode)
            elseif F1<=random2 && random2<F2
                saveT3 = [saveT3,Tstep];
                saveRandom22 = [saveRandom22,random2];

                failAITany = failAITany + 1;
                failAITany1 = [failAITany1,failAITany];

                lambda0 = lambda0*epsilon0;
                matLambda01 = [matLambda01,lambda0];

                Ttot3 = Tstep + deltaTs + deltaTr;
                saveTtot3 = [saveTtot3,Ttot3];

                Tstep = Tstep + 1;

            %Failure is random (random failure mode)
            elseif random2>=F2
                saveT4 = [saveT4,Tstep];
                saveRandom23 = [saveRandom23,random2];

                failRandom = failRandom + 1;
                failRandom1 = [failRandom1,failRandom];

                thetar = thetar*epsilonr;
                matThetar1 = [matThetar1,thetar];

                deltaTr = deltaTr + Tstep;
                saveDeltaTr = [saveDeltaTr,deltaTr];

                Ttot4 = Tstep + deltaTs + deltaTr;
                saveTtot4 = [saveTtot4,Ttot4];

                Tstep = 0;
            end

        else
            Tstep = Tstep + 1;
            Tstep1 = [Tstep1,Tstep];
        end

    end

    allFail = cumFail;
    matAllFail = [matAllFail,allFail];

    allFailSingle = failAITonly;
    matAllFailSingle = [matAllFailSingle,allFailSingle];

    allFailCross = failAITany;
    matAllFailCross = [matAllFailCross,allFailCross];

    allFailRand = failRandom;
    matAllFailRand = [matAllFailRand,allFailRand];

    allFailA = failA;
    matAllFailA = [matAllFailA,allFailA];

    allFailB1 = failB1;
    matAllFailB1 = [matAllFailB1,allFailB1];

    allFailB2 = failB2;
```

```
matAllFailB2 = [matAllFailB2,allFailB2];

allFailB3 = failB3;
matAllFailB3 = [matAllFailB3,allFailB3];

allFailB4 = failB4;
matAllFailB4 = [matAllFailB4,allFailB4];

timeB1 = timeB11;
matTimeB1 = [matTimeB1,timeB1];

timeB2 = timeB21;
matTimeB2 = [matTimeB2,timeB2];

timeB3 = timeB31;
matTimeB3 = [matTimeB3,timeB3];

timeB4 = timeB41;
matTimeB4 = [matTimeB4,timeB4];

TAIT = T + deltaTs + deltaTr; %Total testing time for AIT No.1 only
matTAIT = [matTAIT,TAIT];

totSingle = timeA1 + timeB11 + timeB21 + timeB31 + timeB41;
matTotSingle = [matTotSingle,totSingle];

projectT = T + deltaTs + deltaTr + totSingle;
matProjectT = [matProjectT,projectT];

TtAIT = TAIT + TtAIT; %Total testing time of all AIT until AIT No.1
matTtAIT = [matTtAIT,TtAIT];

projectTt = TtAIT + deltaTA + deltaTB1 + deltaTB2 + deltaTB3 + deltaTB4;
matProjectTt = [matProjectTt,projectTt];

lambda2 = lambda;
```

```
matLambda2 = [matLambda2,lambda2];

lambda02 = lambda0;
matLambda02 = [matLambda02,lambda02];

lambdaA2 = lambdaA;
matLambdaA2 = [matLambdaA2,lambdaA2];

lambdaB12 = lambdaB1;
matLambdaB12 = [matLambdaB12,lambdaB12];

lambdaB22 = lambdaB2;
matLambdaB22 = [matLambdaB22,lambdaB22];

lambdaB32 = lambdaB3;
matLambdaB32 = [matLambdaB32,lambdaB32];

lambdaB42 = lambdaB4;
matLambdaB42 = [matLambdaB42,lambdaB42];

thetar2 = thetar;
matThetar2 = [matThetar2,thetar2];

N = N + 1;

end

meanAllFail = mean(matAllFail);

meanAllFailSingle = mean(matAllFailSingle);
meanAllFailCross = mean(matAllFailCross);
meanAllFailRand = mean(matAllFailRand);

meanAllFailA = mean(matAllFailA);
meanAllFailB1 = mean(matAllFailB1);
meanAllFailB2 = mean(matAllFailB2);
```

```
meanAllFailB3 = mean(matAllFailB3);
meanAllFailB4 = mean(matAllFailB4);

meanTAIT = mean(matTAIT);

meanTotSingle = mean(matTotSingle);
meanTimeB1 = mean(matTimeB1);
meanTimeB2 = mean(matTimeB2);
meanTimeB3 = mean(matTimeB3);
meanTimeB4 = mean(matTimeB4);

meanTtAIT = mean(matTtAIT);

meanProjectT = mean(matProjectT);
meanProjectTt = mean(matProjectTt);

meanLambda2 = mean(matLambda2);
meanLambda02 = mean(matLambda02);

meanLambdaA2 = mean(matLambdaA2);
meanLambdaB12 = mean(matLambdaB12);
meanLambdaB22 = mean(matLambdaB22);
meanLambdaB32 = mean(matLambdaB32);
meanLambdaB42 = mean(matLambdaB42);

meanThetar2 = mean(matThetar2);

%Write calculations results in Excel
xlswrite('Cost_1R.xlsx',transpose(matLambda2),M,'A5');
xlswrite('Cost_1R.xlsx',transpose(matLambda02),M,'D5');
xlswrite('Cost_1R.xlsx',transpose(matLambdaA2),M,'G5');
xlswrite('Cost_1R.xlsx',transpose(matLambdaB12),M,'J5');
xlswrite('Cost_1R.xlsx',transpose(matLambdaB22),M,'M5');
xlswrite('Cost_1R.xlsx',transpose(matLambdaB32),M,'P5');
xlswrite('Cost_1R.xlsx',transpose(matLambdaB42),M,'S5');
xlswrite('Cost_1R.xlsx',transpose(matThetar2),M,'V5');
xlswrite('Cost_1R.xlsx',transpose(matTAIT),M,'Y5');
xlswrite('Cost_1R.xlsx',transpose(matTtAIT),M,'AB5');
xlswrite('Cost_1R.xlsx',transpose(matTotSingle),M,'AE5');
xlswrite('Cost_1R.xlsx',transpose(matTimeB1),M,'AH5');
xlswrite('Cost_1R.xlsx',transpose(matTimeB2),M,'AK5');
xlswrite('Cost_1R.xlsx',transpose(matTimeB3),M,'AN5');
xlswrite('Cost_1R.xlsx',transpose(matTimeB4),M,'AQ5');
xlswrite('Cost_1R.xlsx',transpose(matProjectT),M,'AT5');
xlswrite('Cost_1R.xlsx',transpose(matProjectTt),M,'AW5');
xlswrite('Cost_1R.xlsx',transpose(matAllFail),M,'AZ5');
xlswrite('Cost_1R.xlsx',transpose(matAllFailSingle),M,'BC5');
xlswrite('Cost_1R.xlsx',transpose(matAllFailCross),M,'BF5');
xlswrite('Cost_1R.xlsx',transpose(matAllFailRand),M,'BI5');
xlswrite('Cost_1R.xlsx',transpose(matAllFailA),M,'BL5');
xlswrite('Cost_1R.xlsx',transpose(matAllFailB1),M,'BO5');
xlswrite('Cost_1R.xlsx',transpose(matAllFailB2),M,'BR5');
xlswrite('Cost_1R.xlsx',transpose(matAllFailB3),M,'BU5');
xlswrite('Cost_1R.xlsx',transpose(matAllFailB4),M,'BX5');

%Write average values in Excel
xlswrite('Cost_1R.xlsx',meanLambda2,M,'B5');
xlswrite('Cost_1R.xlsx',meanLambda02,M,'E5');
xlswrite('Cost_1R.xlsx',meanLambdaA2,M,'H5');
xlswrite('Cost_1R.xlsx',meanLambdaB12,M,'K5');
xlswrite('Cost_1R.xlsx',meanLambdaB22,M,'N5');
xlswrite('Cost_1R.xlsx',meanLambdaB32,M,'Q5');
xlswrite('Cost_1R.xlsx',meanLambdaB42,M,'T5');
xlswrite('Cost_1R.xlsx',meanThetar2,M,'W5');
xlswrite('Cost_1R.xlsx',meanTAIT,M,'Z5');
xlswrite('Cost_1R.xlsx',meanTtAIT,M,'AC5');
xlswrite('Cost_1R.xlsx',meanTotSingle,M,'AF5');
xlswrite('Cost_1R.xlsx',meanTimeB1,M,'AI5');
xlswrite('Cost_1R.xlsx',meanTimeB2,M,'AL5');
xlswrite('Cost_1R.xlsx',meanTimeB3,M,'AO5');
xlswrite('Cost_1R.xlsx',meanTimeB4,M,'AR5');
xlswrite('Cost_1R.xlsx',meanProjectT,M,'AU5');
xlswrite('Cost_1R.xlsx',meanProjectTt,M,'AX5');
xlswrite('Cost_1R.xlsx',meanAllFail,M,'BA5');
xlswrite('Cost_1R.xlsx',meanAllFailSingle,M,'BD5');
xlswrite('Cost_1R.xlsx',meanAllFailCross,M,'BG5');
xlswrite('Cost_1R.xlsx',meanAllFailRand,M,'BJ5');
xlswrite('Cost_1R.xlsx',meanAllFailA,M,'BM5');
xlswrite('Cost_1R.xlsx',meanAllFailB1,M,'BP5');
xlswrite('Cost_1R.xlsx',meanAllFailB2,M,'BS5');
xlswrite('Cost_1R.xlsx',meanAllFailB3,M,'BV5');
xlswrite('Cost_1R.xlsx',meanAllFailB4,M,'BY5');

%For reliability calculation
xlswrite('Cost_1R.xlsx',beta,M,'B1');

A = meanLambda02;
B = meanThetar2;
C = meanTtAIT;

M = M + 1;
end
```

APPENDIX B – MATLAB Program for Reliability Calculation

```
AIT = 7;
M = AIT;

lambda0 = xlsread('Cost_1R.xlsx',M,'E5');

lambda1 = xlsread('Cost_1R.xlsx',1,'B5');
lambda2 = xlsread('Cost_1R.xlsx',2,'B5');
lambda3 = xlsread('Cost_1R.xlsx',3,'B5');
lambda4 = xlsread('Cost_1R.xlsx',4,'B5');
lambda5 = xlsread('Cost_1R.xlsx',5,'B5');
lambda6 = xlsread('Cost_1R.xlsx',6,'B5');
lambda7 = xlsread('Cost_1R.xlsx',7,'B5');

beta1 = xlsread('Cost_1R.xlsx',1,'B1');
beta2 = xlsread('Cost_1R.xlsx',2,'B1');
beta3 = xlsread('Cost_1R.xlsx',3,'B1');
beta4 = xlsread('Cost_1R.xlsx',4,'B1');
beta5 = xlsread('Cost_1R.xlsx',5,'B1');
beta6 = xlsread('Cost_1R.xlsx',6,'B1');
beta7 = xlsread('Cost_1R.xlsx',7,'B1');

beta0 = 0.5;

thetar = xlsread('Cost_1R.xlsx',M,'W5');

T = 50000;

T1 = xlsread('Cost_1R.xlsx',1,'Z5');
T2 = xlsread('Cost_1R.xlsx',2,'Z5');
T3 = xlsread('Cost_1R.xlsx',3,'Z5');
T4 = xlsread('Cost_1R.xlsx',4,'Z5');
T5 = xlsread('Cost_1R.xlsx',5,'Z5');
T6 = xlsread('Cost_1R.xlsx',6,'Z5');
T7 = xlsread('Cost_1R.xlsx',7,'Z5');

```

```
T0 = xlsread('Cost_1R.xlsx',M,'AC5');

matR = [];
matTafter = [];

for Tafter=0:T

      R1 = exp(-(lambda1*(Tafter+T1).^beta1)+(lambda1*(T1).^beta1));
      R2 = exp(-(lambda2*(Tafter+T2).^beta2)+(lambda2*(T2).^beta2));
      R3 = exp(-(lambda3*(Tafter+T3).^beta3)+(lambda3*(T3).^beta3));
      R4 = exp(-(lambda4*(Tafter+T4).^beta4)+(lambda4*(T4).^beta4));
      R5 = exp(-(lambda5*(Tafter+T5).^beta5)+(lambda5*(T5).^beta5));
      R6 = exp(-(lambda6*(Tafter+T6).^beta6)+(lambda6*(T6).^beta6));
      R7 = exp(-(lambda7*(Tafter+T7).^beta7)+(lambda7*(T7).^beta7));

      Rcross = exp(-(lambda0*(Tafter+T0).^beta0)+(lambda0*(T0).^beta0));
      Rrandom = exp(-(thetar)*(Tafter));

      R = R1*R2*R3*R4*R5*R6*R7*Rcross*Rrandom;

      matR = [matR,R];

      Tafter = Tafter;
      matTafter = [matTafter,Tafter];

end


x = transpose(matTafter);
y = transpose(matR);

plot(x,y)

%To save data to Excel file
xlswrite('Cost_1R.xlsx',x,8,'B3');
xlswrite('Cost_1R.xlsx',y,8,'C3');
```

APPENDIX C – Data Sets Details of the Total Testing Time, TtAIT, Distribution over the 1000 Simulation Runs

C-1. Data set for Ti = 20h

	Mean [h]			235.60	Median [h]			233.89
TtAIT value during the 1000 runs [h]	232	233	234	235	236	237	238	239
TtAIT occurrence frequency	19	168	334	217	77	16	5	7
TtAIT value during the 1000 runs [h]	240	241	242	243	244	245	246	247
TtAIT occurrence frequency	13	7	7	4	8	6	8	5
TtAIT value during the 1000 runs [h]	248	249	250	251	252	253	254	255
TtAIT occurrence frequency	14	9	7	11	10	14	10	4
TtAIT value during the 1000 runs [h]	256	259	262	264	268	269	272	287
TtAIT occurrence frequency	8	5	1	1	2	1	1	1

C-2. Data set for Ti = 50h

	Mean [h]			590.52	Median [h]			585.26
TtAIT value during the 1000 runs [h]	583	584	585	587	588	589	590	591
TtAIT occurrence frequency	21	223	380	32	1	1	1	1
TtAIT value during the 1000 runs [h]	593	594	595	596	597	598	599	600
TtAIT occurrence frequency	5	2	1	3	5	3	1	1
TtAIT value during the 1000 runs [h]	601	602	603	604	605	606	607	609
TtAIT occurrence frequency	3	2	4	3	3	5	5	2
TtAIT value during the 1000 runs [h]	610	611	612	613	615	616	617	618
TtAIT occurrence frequency	3	4	4	2	1	4	3	4
TtAIT value during the 1000 runs [h]	619	620	621	622	623	624	625	626
TtAIT occurrence frequency	3	5	7	3	4	5	7	3
TtAIT value during the 1000 runs [h]	627	628	629	630	631	632	633	634
TtAIT occurrence frequency	3	3	8	2	4	5	5	2

TtAIT value during the 1000 runs [h]	635	636	637	638	645	654	655	658
TtAIT occurrence frequency	4	4	1	1	1	1	1	1
TtAIT value during the 1000 runs [h]	661	668	669	677	678	683	703	-
TtAIT occurrence frequency	1	1	1	1	1	1	1	-

C-3. Data set for Ti = 100h

	Mean [h]		1133.19	Median [h]		1121.10
TtAIT value during the 1000 runs [h]	1120	1121	1122	1123	1124	1125
TtAIT occurrence frequency	17	130	403	237	45	3
TtAIT value during the 1000 runs [h]	1128	1130	1131	1140	1141	1142
TtAIT occurrence frequency	1	2	1	1	1	1
TtAIT value during the 1000 runs [h]	1143	1144	1145	1146	1148	1149
TtAIT occurrence frequency	1	2	1	1	2	1
TtAIT value during the 1000 runs [h]	1150	1151	1152	1153	1154	1155
TtAIT occurrence frequency	1	3	2	3	2	1
TtAIT value during the 1000 runs [h]	1157	1158	1159	1160	1161	1162
TtAIT occurrence frequency	1	1	1	1	3	2
TtAIT value during the 1000 runs [h]	1163	1164	1165	1166	1167	1168
TtAIT occurrence frequency	2	1	1	5	3	2
TtAIT value during the 1000 runs [h]	1169	1170	1171	1172	1173	1175
TtAIT occurrence frequency	1	2	1	1	3	1
TtAIT value during the 1000 runs [h]	1176	1177	1178	1179	1180	1181
TtAIT occurrence frequency	1	1	1	2	3	5
TtAIT value during the 1000 runs [h]	1182	1183	1187	1188	1189	1190
TtAIT occurrence frequency	3	2	1	4	2	5
TtAIT value during the 1000 runs [h]	1191	1192	1193	1195	1197	1198
TtAIT occurrence frequency	1	4	3	4	3	4
TtAIT value during the 1000 runs [h]	1199	1200	1201	1202	1203	1204
TtAIT occurrence frequency	1	1	1	1	2	3
TtAIT value during the 1000 runs [h]	1205	1206	1207	1209	1210	1211
TtAIT occurrence frequency	1	2	1	4	1	2
TtAIT value during the 1000 runs [h]	1212	1213	1214	1215	1216	1217

TtAIT occurrence frequency	2	1	1	1	2	1
TtAIT value during the 1000 runs [h]	1218	1219	1221	1222	1225	1226
TtAIT occurrence frequency	2	2	2	2	1	1
TtAIT value during the 1000 runs [h]	1228	1230	1240	1248	1251	1252
TtAIT occurrence frequency	1	1	1	1	1	1
TtAIT value during the 1000 runs [h]	1259	1262	1265	1273	1280	1284
TtAIT occurrence frequency	1	1	1	2	1	1
TtAIT value during the 1000 runs [h]	1290	1291	1294	1297	1382	-
TtAIT occurrence frequency	1	1	1	1	1	-

C-4. Data set for Ti = 200h

	Mean [h]		2166.58	Median [h]			2214.29
TtAIT value during the 1000 runs [h]	2139	2140	2141	2142	2143	2144	2145
TtAIT occurrence frequency	1	56	285	358	124	10	1
TtAIT value during the 1000 runs [h]	2158	2161	2163	2164	2169	2186	2190
TtAIT occurrence frequency	1	1	2	1	1	2	2
TtAIT value during the 1000 runs [h]	2191	2193	2194	2200	2203	2204	2205
TtAIT occurrence frequency	1	1	1	1	2	1	1
TtAIT value during the 1000 runs [h]	2208	2210	2212	2213	2214	2216	2217
TtAIT occurrence frequency	1	2	1	1	1	1	1
TtAIT value during the 1000 runs [h]	2223	2225	2226	2228	2229	2232	2234
TtAIT occurrence frequency	1	1	2	2	1	2	1
TtAIT value during the 1000 runs [h]	2235	2237	2239	2240	2241	2243	2244
TtAIT occurrence frequency	1	2	1	1	3	2	2
TtAIT value during the 1000 runs [h]	2245	2248	2252	2254	2257	2258	2260
TtAIT occurrence frequency	1	1	1	1	1	2	1
TtAIT value during the 1000 runs [h]	2262	2263	2264	2267	2268	2269	2271
TtAIT occurrence frequency	2	1	1	1	1	3	2
TtAIT value during the 1000 runs [h]	2272	2273	2274	2275	2278	2279	2280
TtAIT occurrence frequency	1	1	2	1	3	1	1
TtAIT value during the 1000 runs [h]	2281	2284	2285	2286	2287	2289	2291
TtAIT occurrence frequency	1	1	2	3	2	1	2

TtAIT value during the 1000 runs [h]	2298	2301	2302	2303	2305	2308	2309
TtAIT occurrence frequency	1	1	3	1	2	1	1
TtAIT value during the 1000 runs [h]	2313	2316	2317	2318	2319	2320	2322
TtAIT occurrence frequency	3	1	2	2	2	4	1
TtAIT value during the 1000 runs [h]	2324	2326	2328	2331	2332	2333	2335
TtAIT occurrence frequency	2	2	1	1	2	2	1
TtAIT value during the 1000 runs [h]	2336	2337	2338	2339	2341	2342	2343
TtAIT occurrence frequency	2	2	4	1	1	3	2
TtAIT value during the 1000 runs [h]	2344	2349	2354	2363	2371	2379	2380
TtAIT occurrence frequency	2	1	1	1	1	1	1
TtAIT value during the 1000 runs [h]	2383	2385	2402	2412	2418	2453	2459
TtAIT occurrence frequency	2	1	1	1	1	1	1
TtAIT value during the 1000 runs [h]	2460	2466	2478	2479	2497	2499	2506
TtAIT occurrence frequency	1	1	1	1	1	1	1
TtAIT value during the 1000 runs [h]	2508	2534	-	-	-	-	-
TtAIT occurrence frequency	1	1	-	-	-	-	-

APPENDIX D – HORYU-IV's Failures Track Sheet

D-1. Engineering model 1 (EM1) assembly, integration, and testing phase

EM_v1: 2015/02/end - 2015/05/end	OBC testing days: 02/24, 25, 27
System integration starting date: 2015/03/01	EPS testing days: 02/24-27
Missions integration starting date: 2015/04/09	COM testing days: 02/24, 26, 28, 04/13, 21

Failure discovery date	Failure description	Failure type	Sub-system
2015/2/24	Share flash to be pulled up	Design, electronics	OBC
"	3-state buffer to be pulled up	Design, electronics	OBC
"	OR IC Vcc and GND to be reversed	Workmanship	OBC
2015/2/24	Soldering of 2*1Ω resistors	Design, electronics	EPS
2015/2/25	H8 MAIN replaced	Workmanship	OBC
"	Reset from MAIN to COM cannot be executed	Unknown	OBC
2015/2/26	Resistor change from 100Ω to 100kΩ	Design, electronics	COM
"	CLK delay adjustment	Software	COM
"	PCB pattern mistake	Workmanship	EPS
2015/2/27	Resistors adjustment for X panel	Design, electronics	EPS
"	Resistors adjustment for Y panels	"	"
"	Resistors adjustment for Z panels	"	"
"	H8 reset not working	"	"
"	OR IC needs 5V power supply	"	"
"	TC74VHC125 pins to reverse	Workmanship	OBC
"	Resistor near share flash to be pulled	Design,	OBC

	up	electronics	
"	Resistors adjustment	Design, electronics	EPS
2015/2/28	CW transmission problem	Unknown	COM
"	CW cannot be cut, replace PIC	Unknown	COM
"	HK cannot be received, replace PIC	Unknown	COM
"	1200 PTT problem	Unknown	COM
"	9600 PTT not working	Unknown	COM
"	DEMUX removed	Design, electronics	OBC
2015/3/1	Weak soldering and GND was unstable	Workmanship	EPS
"	H8 reset not working	Unknown	Unknown
"	Satellite reset problem	Design, electronics	OBC
"	Resistors value adjustment	Design, electronics	EPS
"	ADC problem, pull up resistance touching another line	Workmanship	OBC
"	Reset	Unknown	Unknown
"	HK not sent, timing to be adjusted	Software	COM
2015/3/2	Resistor/condenser combination	Design, electronics	Unknown
"	H8 reset problem, made low pass filter	Unknown	Unknown
"	H8 reset problem, 3-state buffer connected to 5V	Unknown	Unknown
"	Resistors adjustment for satellite reset	Unknown	Unknown
"	ADC cannot be read, EN and GND pins to be reversed	Workmanship	OBC
"	ADC problem	Unknown	Unknown
"	SPI entering H8MAIN and COM to be reversed (input/output mistake)	Workmanship	OBC
2015/3/6	Pins assignment to be modified	Software	OBC
2015/3/7	1200 PTT issue	Unknown	Unknown
"	CW and FM not decoded	Software	COM

"	Rx and Tx lines reversed	Workmanship	COM
2015/3/9	9600bps problem	Unknown	Unknown
2015/3/12	CS pin correction	Software	OBC
"	1200bps problem	Unknown	Unknown
"	9600bps problem, change of capacitor/resistor combination	Unknown	Unknown
2015/3/19	PIC and MODEM CLK need to be synchronized	Software	COM
"	Resistors adjustment	Design, electronics	EPS
2015/3/20	Program writing to H8s not working	External	OBC
2015/3/23	1200bps problem, MODEM changed	Unknown	Unknown
2015/3/27	MUX changed for digital type	Design, electronics	COM
2015/3/30	Flash memory cannot be read, waiting time to be adjusted	Software	OBC
2015/3/31	ADC reading, CS value to be changed	Software	OBC
"	Program mistake	Software	OBC
"	Data cannot be decoded	Software	Unknown
2015/4/7	OP amp needed	Design, electronics	OBC
2015/4/8	Scramble needed on data	Software	COM
"	Scramble needed not only on data but on the whole packet	"	"
2015/4/9	Transistor from Sband to be removed	Design, electronics	Sband
"	Flash cannot be read when integrated with Sband	Design, electronics	Interface Sband, payload
2015/4/13	SNG problem	Design, electronics	Interface SNG, OBC
"	SNG problem	"	"
"	HVSA problem	Design, electronics	Interface HVSA, OBC
2015/4/14	Big problem	Design, electronics	Interface Big, OBC

"	AODS problem	Design, electronics	Interface AODS, OBC
2015/04/19,	ADC needs to be supplied with 3.3V	Workmanship	OBC
"	HVSA problem	Design, electronics	Interface HVSA, OBC
2015/4/21	Uplink problem,change resistor/capacitor	Design, electronics	COM
"	U/L, decode from GS command	Software	GS
2015/4/26	Diode before each mission MUX necessary	Design, electronics	Interface
2015/4/27	Change capacitor between H8 and H8 reset (1uF -> 10uF)	Design, electronics	OBC
"	AODS-CAM problem	Design, electronics	Interface CAM, OBC
2015/4/29	U/L problem, CS mistake in program	Software	COM
"	Wrong IRQN connection	Workmanship	COM

D-2. Engineering model 2 (EM2) assembly, integration, and testing phase

EM_v2: 2015/06/11 - 2015/08/20	Thermal vacuum test 1: 07/11-18
Integration test 1: 2015/06/11, no failure	Thermal vacuum test 2: 08/03-09
Center box assembly: 2015/06/15-28	Vibrations test: 07/02-04, no failure
Integration test 2: 2015/06/29-07/10	Shock test: 07/05-06, no failure

Failure discovery date	**Failure description**	**Failure type**	**Sub-system**
2015/6/12	Screw problem *1	Design, structure	Structure
2015/6/13	Screw problem *5	Design, structure	Structure
"	Cable probem *1	Design, structure	Structure
"	Other problem *1	Design, structure	Structure

2015/6/14	Screw problem *3	Design, structure	Structure
"	Other problem *1	Design, structure	Structure
2015/6/15	Screw problem *4	Design, structure	Structure
"	Hole problem *1	Design, structure	Structure
"	Other problem *3	Design, structure	Structure
2015/6/16	Screw problem *1	Design, structure	Structure
2015/6/18	Screw problem *5	Design, structure	Structure
"	Other problem *1	Design, structure	Structure
"	Panel modification *1	Design, structure	Structure
2015/6/21	Screw problem *2	Design, structure	Structure
"	Other problem *3	Design, structure	Structure
2015/6/22	Screw problem *2	Design, structure	Structure
2015/6/23	Screw problem *1	Design, structure	Structure
"	Other problem *1	Design, structure	Structure
2015/6/25	Screw problem *4	Design, structure	Structure
"	Other problem *3	Design, structure	Structure
2015/6/26	Screw problem *1	Design, structure	Structure
2015/6/30	Other problem *1	Design, structure	Structure

2015/7/2	Screw problem *1	Design, structure	Structure
2015/7/10	OBO problem	Design, electronics	OBO
2015/7/13	Watch PIC problem	Software	COM
2015/7/14	Sband short-circuit	Unknown	Sband
2015/7/15	AODS problem	Unknown	Interface AODS, OBC
2015/7/16	CW stopped	Software	Mother board
2015/7/17	CW stopped	Software	Mother board
2015/7/23	Memory cannot be read	Design, electronics	OBC
2015/7/27	Over-current protection adjustment Sband TX	Design, electronics	EPS
"	Pull down resitor for PEC mission timing	Design, electronics	Big Apple
2015/7/28	Comparator adjustment	Unknown	Unknown
2015/8/4	CAM problem	Design, electronics	EPS
"	HVSA, SNG, AVC, Sband problem (2V remaining)	Design, electronics	Interface Sband, payload
"	Sband turned ON in 2 steps	Design, electronics	Unknown
2015/8/5	SNG problem (busy pin problem)	Software	Interface SNG, OBC
"	AVC problem	Design, electronics	AVC
2015/8/6	Sband high data loss	External	Sband
"	Sband problem	Unknown	Sband
2015/8/8	Leak from OBC	Design, electronics	OBC
2015/8/13	Leak from Lband	Unknown	Lband

D-3. Flight model (FM) assembly, integration, and testing phase

FM: 2015/08/27 - 2016/01/15	Thermal vacuum test 1: 10/29-30
Structure assembly: 2015/09/28-10/28	Thermal vacuum test 2: 12/10-11
Center box assembly: 2015/10/09-10/28	Vibrations test 1: 11/02-03, no failure
Functional test 1: 08/27-10/28	Vibrations test 2: 12/18, no failure
Functional test 2: 11/01	Shock test: 11/04, no failure
Functional test 3: 11/05-06	Shock test Tsukuba: 11/09-12
Functional test 4: 11/13-12/09	Plasma test: 12/21-01/04 (approximation)
Functional test 5: 12/13-17	Batteries charging: 01/05-08
	Delivery: 01/09
	Final checks Tsukuba: 01/12-15

Failure discovery date	**Failure description**	**Failure type**	**Sub-system**
2015/8/27	GPS plate modification	Design, structure	AODS
2015/9/10	OCP problem	Design, electronics	EPS
2015/9/12	OBO/AVC couldn't acquire data during testing	Unknown	OBO, AVC
2015/9/18	Solar cell broken	Workmanship	EPS
2015/9/28	Batteries possible leakage	External	EPS
2015/9/28	Sun sensor boxes to be painted in black	Design, structure	AODS
2015/10/2	Satellite center box frame design mistake	Design, structure	Structure
2015/10/6	Alodine mistake	Design, structure + workmanship	Structure
2015/10/8	Resistances adjustment 1	Design, electronics	EPS
2015/10/8	Resistances adjustment 2	Design, electronics	EPS
2015/10/8	Pin assignment error	Software	OBC

2015/10/8	Through hole position < 1mm from GND	Workmanship	EPS
2015/10/11	AVC male/female connector	Design, structure	Structure
2015/10/13	S-band TX	External	Sband
2015/10/16	AODS-CAM lines to exit satellite after final closing	Design, electronics	Interface AODS-CAM
2015/10/17	S-band busy pin cannot be activated by OBC	Software	Sband
2015/10/21	Sun sensor not fitting well	Design, structure	Interface AODS, structure
2015/10/21	Bottom frame holes not aligned	External	Structure
2015/10/21	HVSA L-shape PCB cannot fit center box	Design, structure	HVSA
2015/10/22	JAXA simplified PAF cannot be used	External	Structure
2015/10/26	OBC program wrong setting of satellite reset pin	Software	OBC
2015/10/26	Program for D/L	Software	COM
2015/10/26	S-band	Unknown	Sband
2015/10/28	OBO and AVC Ch.2 cannot be read	Unknown	OBO, AVC
2015/10/28	Reset cannot be executed if AODS is ON	Unknown	Interface OBC, AODS
2015/10/30, 10:49	Share flash cannot be read by H8 MAIN or COM	Unknown	OBC
2015/10/30, 11:15	Infinite reset at low temperature	Design, electronics	OBC
2015/11/4	Kill SW command execution problem	Software	OBC
2015/11/6, 10:20	OBC program modification	Software	OBC
2015/11/6, 10:44	Sat log cmd problem	Software	COM
2015/11/6,	Cmd reception (HVSA, kill SW,	Software	COM

10:52-11:23	GPS data) problem		
2015/11/6, 13:36-14:47	MAIN/COM plugging mistake	Workmanship	OBC
2015/11/12	PEEK screw shift	Workmanship	Structure
2015/11/23	PEC problem	Unknown	Big
2015/11/23	AVC problem	Software	AVC
2015/11/23	AVC problem	Software	AVC
2015/11/28	HVSA overcurrent problem	Workmanship	HVSA
2015/12/7	Interface problem	Workmanship	EPS
2015/12/8	50mm clearance requirement between separation surface and lowest part of satellite not satisfied	Workmanship	Structure
2015/12/8, 11:40	MAIN reset everytime command sent to SNG	Software	Interface OBC, SNG
2015/12/8, 11:50	SNG cannot read flash memory	Unknown	Interface OBC, SNG
2015/12/8	Sband process problem with AVC	Software	Interface Sband, payload
2015/12/8	Sband data D/L problem with AVC	Software	Interface Sband, payload
2015/12/8	OBC program modification	Software	OBC
2015/12/9	L-bank leakage	Design, electronics	Lband
2015/12/9, 19:19	D/L stopped	Software	COM
2015/12/9	Kill SW monitoring	Software	GS
2015/12/9	Data not decoded by GS	Software	GS
2015/12/9	Sat. reset not executed through Lband	Unknown	Interface L, S, OBC
2015/12/9	Watch PIC reset not executed through Lband	Unknown	Interface L, S, OBC
2015/12/10, 10:13	Command interface problem, when AODS ON, HVSA ON too	Software	GS
2015/12/10, 10:15	TX stopped due to HVSA oversurrent	Workmanship	Interface HVSA, COM

2015/12/10, 16:49	Command for sat. log	Software	OBC
2015/12/10	HK data analysis problem	Software	GS
2015/12/10	Sensor data D/L problem, H8 reset	Software	OBC
2015/12/10	No command acknowledgment	Unknown	COM
2015/12/11, 10:17	Missing function in GS interface	Software	GS
2015/12/11, 17:11	Irregular reset	Software	OBC
2015/12/13	HVSA problem	Unknown	HVSA
2015/12/13	HVSA problem	Unknown	HVSA
2015/12/13	AODS data D/L stopped	Software	COM
2015/12/14	AODS not turned ON	Workmanship	AODS
2015/12/15	AVC problem	Unknown	AVC
2015/12/17	OBO trigger not performing	Software	OBO
2016/1/2	Missing data from CAM	Unknown	Interface S, payload
2016/1/7	OBC program modification	Software	OBC

APPENDIX E – HORYU-IV’s Lessons Learned

	Team member A	Team member B	Team member C
Structure and thermal	・When possible, favor direct connection between components instead of using connectors ・To prevent misunderstandings between satellite developer and external manufacturer, provide simple and imaged explanations	・To prevent mistakes, communication between different team members should be constant and regular ・To attach thermal sensors, avoid using RTV because it needs to be coupled with polyimide tape, which interacts with the read value	-
Bus sub-systems (OBC, COM, EPS)	・To prevent work overload on one team member, develop one PCB per sub-system	-	-
Missions	・To prevent work overload on one team member, develop one PCB per sub-system	・Number of missions should be minimized ・Main mission should be transparent on its progress since lack of transparency can affect the whole satellite architecture and schedule	-
Interfaces	-	・Proper project management is necessary to ensure all different parts	-

		can properly move as one system, not as an assembly of systems	
Ground station	-	・All members should know how to operate ground station to facilitate on-orbit operations	-
Team	・To prevent loss of knowledge, avoid team members turn over during project ・For a forward moving project, ensure team unity through non-project related activities ・International team helps understand different ways of thinking and overall helps an individual to grow ・There should be at least one person who understands the whole satellite at its technical level to facilitate knowledge acquisition	・To prevent loss of knowledge, avoid team members turn over during project	・To prevent loss of knowledge, avoid team members turn over during project ・For a forward moving project, ensure team unity through non-project related activities ・International team helps understand different ways of thinking and overall helps an individual to grow ・There should be at least one person who understands the whole satellite at its technical level to facilitate knowledge acquisition ・There should be at least one person able to guide and explain logic of decision-making and processes
Project	・To prevent emails misunderstanding, favor local manufacturers to be able to interact directly face-to-face	・Each team should have a responsible leader to prevent project manager work overload in addition to managing	・No excuse should be accepted to ensure deadlines are satisfied ・Prepare long term schedule, including

	・Through multiple mission types and inherent satellite complexity, knowledge can be improved in various fields		development steps and tests, to visualize the impact of a delay on the overall project ・Weekly meetings are necessary to share progress, ideas for improvement, and problems ・To prevent mistakes, constantly and regularly revise requirements allocation sheet ・To prevent misunderstandings, always ask questions and confirm what you think is obvious ・Breakdown of each team member roles and responsibilities is necessary
Resources	・To prevent time and money loss, organize tools in an easy and proper manner	・Proper software should be acquired to prevent schedule conflict when have to borrow from other laboratories	-

	Team member D	Team member E	Team member F
Structure and thermal	・When possible, mount as many as possible temperature sensors on the different sub-systems	・When designing structure, always keep in mind how to improve it for easy assembly/desassembly and tests compatibility ・When possible, mount as many as possible temperature sensors on the different sub-systems	-
Bus sub-systems (OBC, COM, EPS)	-	・To reduce harness and facilitate testing good to have several sub-systems integrated on one PCB ・To facilitate testing in parallel and to prevent work overload on one team member, develop one PCB per sub-system ・Always include level converter for SPI line ・Use past projects' developed bus system to facilitate development and launcher safety coordination ・To prevent leakage from one sub-system to another, use ADCs with backflow prevention diodes ・To prevent erasing all data of a flash	・Reset should be on power line directly not on OBC power line

		memory when overwriting, use FRAM or EEPROM type memory ・Minimize the number of reset types ・Minimize the number of communication systems ・To facilitate signal reception on bus side, ensure high impedance when the signal goes from a sub-system to the bus ・To prevent surge, mount coils ・To facilitate development and being able to focus on payload and interfaces only, order already made bus sub-systems ・OBC program requires constant modifications so the person in charge should be an internal member of the entity developing the satellite, not external ・When possible, OBC hardware and software developers should be the same person	
Missions	-	・To reduce harness and facilitate testing good to have several sub-systems integrated on one PCB ・Always include level converter for SPI	・Developing sub-systems in-house favors learning-by-doing ・Quadrant photodiodes are easy to use, but they have low FOV

		line ・To prevent erasing all data of a flash memory when overwriting, use FRAM or EEPROM type memory ・To facilitate signal reception on bus side, ensure high impedance when the signal goes from a sub-system to the bus	・COTS gyro can be very noisy and lack of accuracy ・Should have considered turning ON PEC mission without information from sun sensors
Interfaces	・End-to-end test should be carried out thoroughly for each mode and this can take a non-negligible time to be accounted for in the project schedule	・Useful to have different interfaces connected through connectors for easy plug-in/plug-out ・OBC should include all the digital data, while analog data such as transponder and sensors should be external to OBC	-
Ground station	-	-	・Automate S-band GS data reception and data analysis
Team	・For a forward moving project, ensure team unity through non-project related activities ・There should be at least one person who understands the whole satellite at its technical level to facilitate knowledge acquisition	・For each sub-system, define one responsible ・There should be at least one person who understands the whole satellite at its technical level to facilitate knowledge acquisition ・OBC program requires constant modifications so the person in charge	・To prevent loss of knowledge, avoid team members turn over during project

		should be an internal member to the entity developing the satellite, not external ・When possible, OBC hardware and software developers should be the same person	
Project	・Prepare long term schedule, including development steps and tests, to visualize the impact of a delay on the overall project ・Weekly meetings are necessary to share progress, ideas for improvement, and problems ・To prevent mistakes, ensure proper knowledge transfer, and ensure proper verifications, documentation creation and maintenance throughout the whole project is necessary	・Breakdown of each team member roles and responsibilities is necessary	・Breakdown of each team member roles and responsibilities is necessary ・Weekly meetings are necessary to share progress, ideas for improvement, and problems ・Prepare long term schedule, including development steps and tests, to visualize the impact of a delay on the overall project ・To prevent mistakes, ensure proper knowledge transfer, and ensure proper verifications, documentation creation and maintenance throughout the whole project is necessary
Facilities	-	・When possible, have testing facilities where the satellite is developed to facilitate schedules coordination and minimize waste of time and money	-

Resources	・To prevent time and money loss, organize tools in an easy and proper manner ・To prevent time and money loss, organize electronic parts in an easy and proper manner	・Parts procurement should be managed by one person	-
Improvement ideas	-	-	・For AODS, instead of information coming from the solar panels, use coarse (3 needed) or fine sun sensors (6 needed) ・Increase number of permanent magnets to fasten satellite stabilization
Others	・When students are involved, satellite project should match graduation thesis themes ・Through international satellite project, can learn practical systems engineering and improve skills in various engineering and other fields	-	・GPS test should be sufficiently performed outside when there is clear view

	Team member G	Team member H	Team member I
Structure and thermal	-	-	・Always take into account harness and its interaction with the other structural and electrical elements
Bus sub-systems (OBC, COM, EPS)	-	-	・OBC program requires constant modifications so the person in charge should be an internal member of the entity developing the satellite, not external
Missions	-	・Have one micro-controller per mission	・To improve efficiency and satellite quality, main mission should be integrated with other sub-systems as early as possible
Interfaces	-	・All team members should understand interfaces	・Integration tests should be started as early as possible
Ground station	-	-	・GS software should be started as early as possible and by the time the space segment development reached FM, GS software sould be 80% complete ・GS is an integral part of satellite project and should be considered as important as space segment development ・Have a central server to easen information exchange with amateur radio community

Team	・There should be at least one person who understands the whole satellite at its technical level to facilitate knowledge acquisition ・To improve knowledge transfer, include all members of one sub-system in the conversation	・To facilitate communication, information dissemination, and verifications, all team members should be gathered in one room	・The number of members to be accepted for a project should be limited depending on the project complexity ・To prevent loss of knowledge, avoid team members turn over during project ・To facilitate communication, information dissemination, and verifications, all team members should be gathered in one room
Project	・Weekly meetings are necessary to share progress, ideas for improvement, and problems ・To prevent mistakes, ensure proper knowledge transfer, and ensure proper verifications, documentation creation and maintenance throughout the whole project is necessary ・In students satellite projects, design decisions, within certain limits, should be driven by the students, not the professors ・To prevent misunderstandings, always ask questions and confirm what you think is obvious	・Whenever possible, do work in parallel ・Breakdown of each team member roles and responsibilities is necessary ・Weekly meetings are necessary to share progress, ideas for improvement, and problems ・Prepare long term schedule, including development steps and tests, to visualize the impact of a delay on the overall project	・To prevent mistakes, ensure proper knowledge transfer, and ensure proper verifications, documentation creation and maintenance throughout the whole project is necessary
Resources	・Ensure financial support as early and fast as	-	・Parts procurement should be managed by

	possible		one person
Improvement ideas	・For international projects, define core working time to prevent schedules coordination issues between different work cultures	-	・When it is identified team members cannot perform task (programming, electrical circuit designing, others) held training sessions during BBM
Others	･When students are involved, satellite project should match graduation thesis themes	・When students are involved, satellite project should match graduation thesis themes	・To keep motivation high after satellite launch, write peer-reviewed papers from on-orbit results ・All information should be properly documented and open to all team members, no secret should be accepted ･As much as possible. do testing as you fly prior to satellite delivery ・For data and knowledge exchange, join network of satellite projects such as UNISEC

	Team member J	Team member K	Team member L
Bus sub-systems (OBC, COM, EPS)	-	-	・Improve efficiency by replacing old micro-controllers used in OBC and AODS
Missions	・Do by yourself as much as possible to broaden skills and knowledge	-	-
Ground station	・All members should know how to operate ground station to facilitate on-orbit operations	・Have a central server to easen information exchange with amateur radio community	-
Team	・The number of members to be accepted for a project should be limited depending on the project complexity ・To prevent loss of knowledge, avoid team members turn over during project	・To prevent loss of knowledge, avoid team members turn over during project ・Everybody should be held responsible, no excuse should be accepted ・For a forward moving project, ensure team unity through non-project related activities	・To prevent loss of knowledge, avoid team members turn over during project ・International team helps understand different ways of thinking and overall helps an individual to grow ・There should be at least one person who understands the whole satellite at its technical level to facilitate knowledge acquisition ・For a forward moving project, ensure team unity through non-project related activities
Project	・Space engineering using hands-on activities through a satellite project helps improve understanding of the	・To prevent mistakes, ensure proper knowledge transfer, and ensure proper verifications, documentation creation and	・For international projects, ensure all necessary and relevant documentation is in English to facilitate and enhance knowledge

	theory taught during classes	maintenance throughout the whole project is necessary ・When project manager is not a student, consider having a student as sub-PM for him/her to understand satellite project management ・Prepare long term schedule, including development steps and tests, to visualize the impact of a delay on the overall project	transfer ・Weekly meetings are necessary to share progress, ideas for improvement, and problems ・To prevent mistakes, ensure proper knowledge transfer, and ensure proper verifications, documentation creation and maintenance throughout the whole project is necessary
Facilities	-	-	・Ensure each function (ex.: soldering, assembly, functional test, others) has its dedicated location to prevent equipment mobility and loss
Resources	-	-	・To prevent time and money loss, organize tools in an easy and proper manner ・To prevent time and money loss, organize electronic parts in an easy and proper manner
Improvement ideas	-	-	・To improve satellite overall efficiency and capability, implement step-by-step newer technology as back-up of a sub-system (ex.: micro-controller, integrated switches,

			others) ・Have re-work station to easily remove ICs and do corrections on PCB when needed
Others	・Do testing including margins to investigate possible on-ground/space results differences prior to satellite launch ・Through international satellite project, can learn practical systems engineering and improve skills in various engineering and other fields	・Have a central, well-managed database where all information relevant to the project can easily be accessible ・All information should be properly documented and open to all team members, no secret should be accepted ・Keep website active and updated ・As much as possible. do testing as you fly prior to satellite delivery	・Have a central, well-managed database where all information relevant to the project can easily be accessible ・All information should be properly documented and open to all team members, no secret should be accepted