Product Assurance in Agile Space Systems Development

Master's Thesis
University of Turku
Department of Information Technology
Software Engineering
2015
Minna Isomäki

Supervisors:
Kaisa Könnölä
Ville Leppänen
Ville Rantala

UNIVERSITY OF TURKU
Department of Information Technology

ISOMÄKI, MINNA: Product assurance in agile space systems development

Master's thesis, 90 p., 12 appendix.
Software Engineering
October 2015

Product assurance is an essential part of product development process if developers want to ensure that final product is safe and reliable. Product assurance can be supported with risk management and with different failure analysis methods.

Product assurance is emphasized in system development process of mission critical systems. The product assurance process in systems of this kind requires extra attention. In this thesis, mission critical systems are space systems and the product assurance process of these systems is presented with help of space standards.

The product assurance process can be supported with agile development because agile emphasizes transparency of the process and fast response to changes. Even if the development process of space systems is highly standardized and reminds waterfall model, it is still possible to adapt agile development in space systems development. This thesis aims to support the product assurance process of space systems with agile development so that the final product would be as safe and reliable as possible.

The main purpose of this thesis is to examine how well product assurance is performed in Finnish space organizations and how product assurance tasks and activities can be supported with agile development. The research part of this thesis is performed in survey form.

Key words: Product assurance, space systems, ECSS-standards, agile development

TURUN YLIOPISTO
Informaatioteknologian laitos

ISOMÄKI, MINNA: Tuotevarmennus avaruussysteemien ketterässä kehityksessä

Diplomityö, 90 s., 12 liites.
Ohjelmistotekniikka
Lokakuu 2015

Tuotevarmennus on oleellinen osa tuotekehitysprosessia kun halutaan varmistua siitä, että lopputuote on turvallinen ja luotettava. Tuotevarmuutta voidaan tukea erityisesti riskienhallinalla ja erilaisilla virheiden analysointimenetelmillä.

Tuotevarmennuksen merkitys on korostunut tehtäväkriittisten systeemien tuotekehityksessä, koska systeemien toiminta ja mahdollisesti pitkä käyttöikä vaativat sen, että systeemi toimii alusta alkaen niin kuin on tarkoitettu. Tässä diplomityössä tehtäväkriittisiksi systeemeiksi on valittu avaruusalan systeemit ja niiden tuotevarmistukseen tutustutaan avarustandardiston avulla.

Ketterän kehityksen avulla voidaan tuotteen kehitysprosessia tehostaa sillä ketteryys painottaa etenkin prosessin läpinäkyvyyttä ja nopeampaa reagointia muutoksiin. Vaikka avaruussyteemien kehitysprosessi on vahvasti standardisoitu ja muistuttaa hyvin paljon vesiputousmallia, on ketteryyttä kuitenkin mahdollista soveltaa osana avaruusprojektia. Tässä työssä avaruussyteemien tuotevarmuusprosessia pyritään tehostamaan ketteryyden avulla niin, että lopputuote olisi mahdollisimman turvallinen.

Tämän diplomityön tarkoituksena on selvittää miten hyvin tuotevarmuus on toteutettu suomalaisissa avaruusalan yrityksissä ja miten tuotevarmuuden työtehtäviä ja aktiviteetteja voitaisiin tukea ketterien menetelmien avulla. Tässä työssä tutkimus on suoritettu kyselytutkimuksen muodossa.

Avainsanat: Tuotevarmennus, avaruussysteemit, ECSS-standardit, ketterä kehitys

The originality of this thesis has been checked in accordance with the University of Turku quality assurance system using the Turnitin OriginalityCheck service.

Turun yliopiston laatujärjestelmän mukaisesti tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck -järjestelmällä.

Contents

List of Figures

Chapter 1

Introduction

This thesis was produced as a part of AgiSpacES (Agile Development Methods for Embedded Systems in Space Industry) project in Technology Research Center of University of Turku.

The purpose of this thesis is to introduce the term product assurance and concentrate on space product assurance which is mainly controlled by standards. This thesis also introduces agile development and the purpose of the agile development is to act as supplement which offers good practises that would support the performance of space product assurance activities.

Product assurance is presented in Chapter 2 and is in an important role in system development where safety and reliability act in the key roles. Risk management and failure analysis methods are presented as activities that support product assurance process because with risk management and failure analysis methods it is possible to identify and eliminate possible problems before they even occur.

Chapter 3 presents the product assurance of space technology. Product assurance is in an essential role in space systems because they are critical systems which must be safe and reliable. In space industry, product assurance is supported with standards which are briefly presented in Chapter 3. These standards offer guidelines for successful product assurance process.

The main focus of the Chapter 4 is in agile development which is a development method which emphasizes transparency of the development process and fast response to changes. Agile defines different methods that follow agile principles and values. In this thesis, presented methods are Scrum, Lean, Kanban, and Extreme programming

The main purpose of these three theory chapters is to give the reader an understanding of topics: product assurance, product assurance in space technology and agile development.

This thesis aims to answer the following two research question

RQ1 How well product assurance is performed in Finnish space organizations generally?

RQ2 How space product assurance can be supported with agile?

Chapters 5 and 6 are survey chapters. In this thesis the survey is based on report "SPACE-2000, Development of PA/QA for space instruments"[1] made in 1996. The report was made by AL Safety design and the main purpose of the report was to examine how product and quality assurance were implemented in Finnish space industries at that time.

The answers for RQ1 and RQ2 can be found for the survey. The purpose of this thesis is to update the previous survey and examine how well product assurance is performed in Finnish space organizations and how product assurance activities can be supported with agile methods and practices of them.

Because of these RQ's the survey also consists of two different sections. The first section answers to RQ1 and the second section answers to RQ2. In this thesis the respondents of the survey are the experts of the space system development so that the result of survey would be as reliable as possible.

Chapter 2

Product Assurance

Product assurance, also known as PA, is in a major role in projects where reliability and safety are required properties. The main goal of product assurance is to take care that the product or system fulfils the technical and design requirements and the needs of the customer. If for some reason the system does not fulfil these requirements, a system might behave in an unexpected way and can cause an accident or disaster. [2]

It is relevant to eliminate *failures*, *faults* and *hazards* from the system. A failure is a situation where the product or system works in a different way that it should and a fault is a divergent state of the system which can be caused by a failure. A hazard is known as a potential situation that can cause or support damage. A hazard is a result of at least one cause which can be human error or an unpredictable event.[3] *Risk management* will be presented later in Section 2.3 and its main purpose is to support the elimination process of potential problems such as failures, faults and hazards. A risk management process itself can also be supported with different *failure analyses* that help identify the failures and causes of failures.

The most crucial meaning of the product assurance is to ensure that all products are safety and all possible causes that might cause harm are eliminated before they even occur. Product assurance is crucial in both hardware and software domains and it is especially important in *embedded systems* which are systems that consist of mechanic, electronic and software parts. In systems of this kind, the collaboration of different components

should be seamless. If development of safe and reliable systems is desired, it is essential to understand the safety and reliable aspects of each component.

2.1 Safe Software Engineering

Traditionally it is thought that safety issues only concern hardware components because the software does what it is programmed to do. Partly this is true because software itself cannot cause harm or damage but mistakes in requirements and programming behind the software are the key safety issues. [4].

Traditionally software requirements are defined at the beginning of the project because only then it is possible to make sure that these requirements are clear to all programmers. Design work is easier to do if the requirements are defined in an early phase and the result of the design work is better because it responds to the original requirements. However the requirement definition of this kind do not offer enough flexibility to change these requirements in late phases which might lead to weaker compliance with the requirements.

Programming mistakes, called *bugs*, are quite common and almost all software contains them. For that reason the main focus is in bug elimination because these bugs should be eliminated or minimized from the final software product. Because software is not a tangible product hidden bugs have to be found using software testing and testing should be performed throughout the project. [5]

Usually in testing context also *verification* and *validation* are mentioned. The purpose of verification is to ensure that the final product is correctly done. This means that the final product is in line with original design and architecture specifications and plans. The product can be said to be reliable if it is verified. If the product is not reliable, it is defective and includes bugs which can cause failures and undesirable behavior. Validation ensures that the right product is made. Precisely, this means that the product responds to the original needs and requirements of the customer. Therefore, the validated product can be considered useful. [5][6]

Verification and validation together ensure that the requirements that are set for the system or product are met. Because the product responds to these requirements it can be assumed that the product is safe. Even if the final product would be manufactured by following requirements and specification, software failures and hazards may still occur. One reason that might cause a software failure or hazard is change in circumstances. If the conditions of environment changes, it is possible that the system no longer works as desired. [4][7]. Software failure causes can be also a human mistake, problem in hardware or software and unexpected event or input.

Safety and security issues become more important because of the amount of software increases all the time. Software is one main component in systems so software needs to be safe. One important thing is to recognize when the software is safe or unsafe. It is also crucial to remember that the software safety is the responsibility of everyone, including project managers; software and system engineers; and software assurance and safety personnel. [3]

In a software development domain systems can be divided into two categories: security critical and safety critical software systems. The difference between these two is that safety critical system software might not cause harm to the environment which means that the safety software systems do not cause disasters that might damage environment or humans. The main focus of the systems of this kind is in system controlling. The definition of security critical systems is opposite and in a security critical system the environment might not cause harm to the software which means that the third party is not able to harm the software. The main focus of the systems of this kind is in system protection. [8]

In software domain one example of safety failure would be a calculating failure. The system fails because it unable for some reason to calculate results correctly and an example of security failure would be a virus which causes system failure. In this thesis, the main focus is in safety critical software systems.

2.2 Safe Hardware Engineering

Difference between hardware and software products is major. A hardware product is a tangible product that can consists of different components with different materials. Mechanical and electronic components can be made from materials such as: metal, plastic, ceramic and composite. Each of these materials has their own weaknesses and can fail in different ways. In addition it is crucial to know how these materials act in different environments and how the different chemical compounds change the properties of those components. [9]

Metallic components can for example corrode, fracture, deform and leak. Especially corrosion is a very common and well known failure property which is caused by electrochemical reaction between metal and environment. The main problem is that the corrosion changes the properties of the metal and for that reason component can fracture under stress. Corrosion can also increase the electronic resistance of the components which can downgrade the performance of the component. [9]

Component quality is a major issue in hardware safety. In safety critical systems only certified components should be used because only then it is certain the components are safe and properties are known. One way to measure the component quality is a *failure rate (Lambda)* which is a unique rate that tells the probability of component failure per time unit. [10]

Figure 2.1 shows that a component needs a burn in time before it works correctly and safely. When the first period has been completed, the useful life period starts. The length of this period depends on the quality of the used component. Usually, the value of the failure rate is quite low during this period because only random failures occur. Because components are physical products they do not last forever. After the useful life period, wear out period begins. In this last period, the component is not safe because the effects of use are starting to show in its performance and the components failure rate increases.

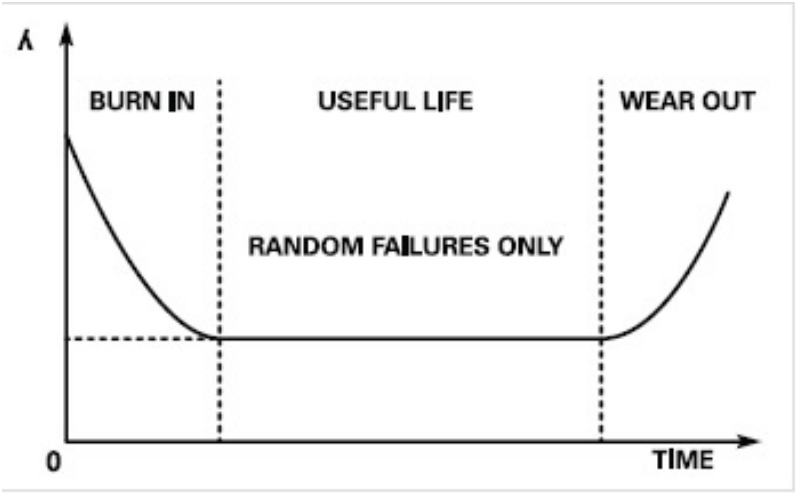


Figure 2.1: Failure rate of component. [10]

The quality of the component is important but it is crucial to be aware that the environment and changes in conditions can change the properties of a component. If the component is exposed to conditions in which it is not designed to be used, there is a reason to believe that the component does not work correctly and is unsafe.

2.3 Risk Management

Risk management is a function that leads multiple activities which ensure that requirements and goals are able to be achieved. Project management consist of three parts: Project control, system engineering, and safety and mission assurance. It is essential to successful risk management that the project team understands these three parts and implements them throughout the project lifecycle. Figure 2.2 shows that project control part concentrates on cost, schedule and people management issues. System engineering part is more technical and focuses on configuration management and technical aspects. In this thesis, the most interesting part is safety and mission assurance which focuses on environment and safety. [11]

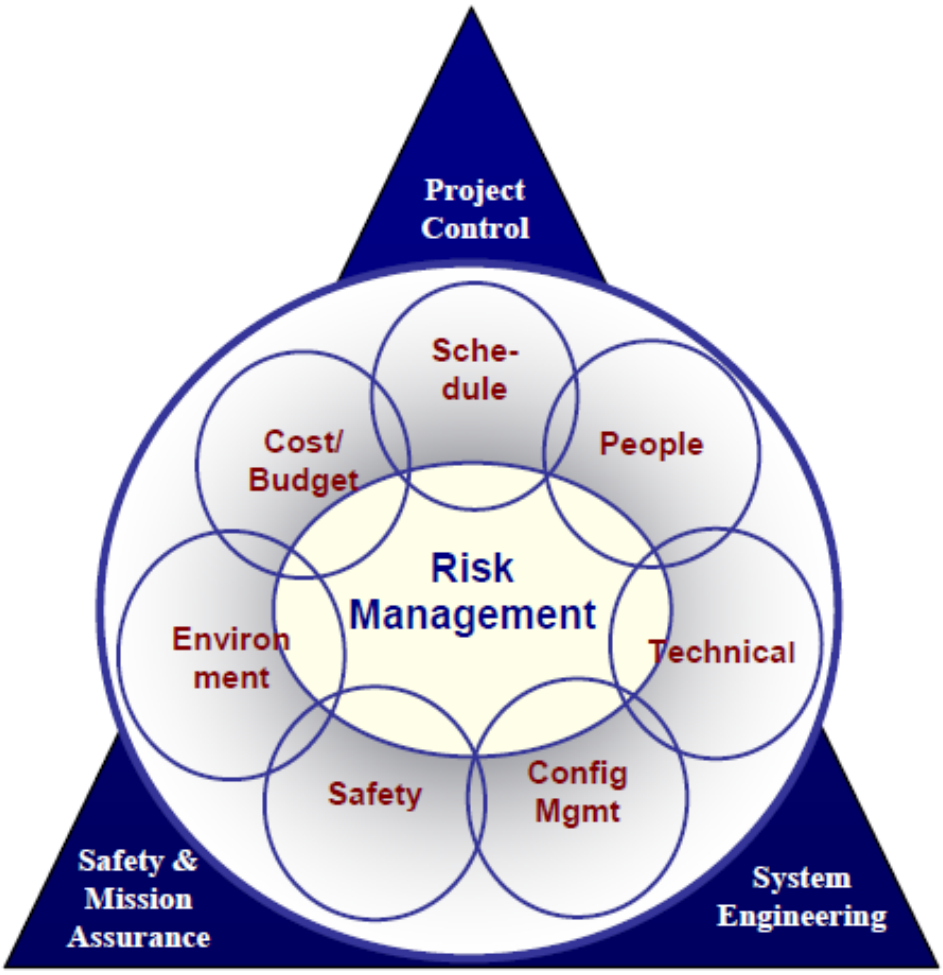


Figure 2.2: Parts of Project Management. [11]

Risk management is in a crucial role in product assurance because systems are barely ever risk-free. There is always a possibility of a crash and the risk management is best known for a paradigm in which a problematic event is identified and eliminated before it occurs. Use of safety risk management helps to identify failure and hazard models and if these models are recognized it is easier to specify the reasons, dangerousness and their incidence. [7]

Risks can be divided as follows [7]:

- *Total risk*: The total sum of all identified and unidentified risks.
- *Identified Risk*: A risk which is specified through different risk analyses.

- *Unacceptable Risk*: A sub-collection of identified risks which are controlled or eliminated.
- *Residual Risk*: Risk that is left after the system safety work is fully done and it is a sum of unacceptable, acceptable and unidentified risks.
- *Acceptable Risk*: A part of identified risk and it can persist without any engineering actions.
- *Unidentified Risk*: A real and important risk that has not been determined. Several of these risks are measured and identified after accidents but still some of the risks will never be determined.

Risk management starts with risk identification which can be based on the information that is obtained for example from failure analyses which are focused on later in Section 2.4. After risk identification, risks should be classified and prioritized. In classification, similar identified risks are classified to the same group although using previous risk division. The classification facilitates the understanding of the nature about the risks which ease the planning of mitigation actions. Also duplicates and equivalent risks can be found by classification; this eases the mitigation process and saves resources such as time because the same mitigation actions can be used for multiple risks. [11]

After the risk classification process, risks can be prioritized. One way to prioritize the risks is to exploit the likelihood and consequence of occurrence. Likelihood means the probability that an identified risk may occur and consequence is the worst possible potential result that risk can cause. The most important thing in prioritization is to determine which are the most probable and critical risks and are therefore eliminated first. [11]

Risk management continues with planning, tracking and controlling phases. The main purpose of the planning phase is to prepare a plan which tells how to concern different risks if they occur. Tracking and controlling phases monitor risks and use mitigation actions if risks occur. [11]

Risk management consist of these different phases which form a continuous circle as shown in Figure 2.3.

Figure 2.3: Continuous Risk Management cycle. [11]

Risk management and analysis are in an important role in product assurance because the main goal is to produce safe and reliable products. The main purpose of risk management is to offer more information about risks to the managers and developers. This documented information makes it easier to understand risks and control them in an efficient way. Successful risk management also requires clear, ongoing and open communication between project members. [11]

Even if risk management eliminates possible problems, it is not always possible to eliminate all risks. In such a situation it is important to identify the acceptable level of risk which ensures that the risks do not cause massive harm to the system. [11]

2.4 Failure Analysis

The simple failure solving process of the system could consist of the following steps [9]:

1. Accurate problem definition
2. The identification of potential failures causes
3. Likelihood evaluation of each failure cause
4. Failure cause prevention

In the first step it is important to know what the problem is because only then the cause of the problem is possible to be identified in step two. In step three, likelihood evaluation makes it is possible to get information about the incidence of failure cause and in the last step the final work is to prevent the cause of a failure using the best possible solution found.

It is common in many organizations that the failure analysis responsibility is assigned to a single department which is not a good idea. Failure analysis is more effective if the failure analysis team consists of experts with a different knowledge base such as engineers, quality specialists and manufacturing technicians. [9]

There are also several failure analysis methods which support and help failure solving and detection process. This thesis presents *Fault tree analysis (FTA)* and *Failure modes and effects analysis (FMEA)*.

2.4.1 Fault Tree Analysis

Fault tree analysis is a graphical method that focuses on undesired events that are called TOP events. Especially FTA is interested in all potential causes that can produce TOP events. [9] Simply, FTA could be presented using four logical symbols shown in Figure 2.4.

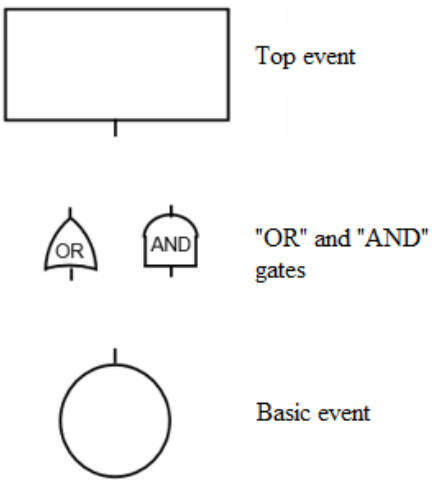


Figure 2.4: Symbols of FTA.

[12]

With these symbols it is possible to form a tree structure which is presented in Figure 2.5.

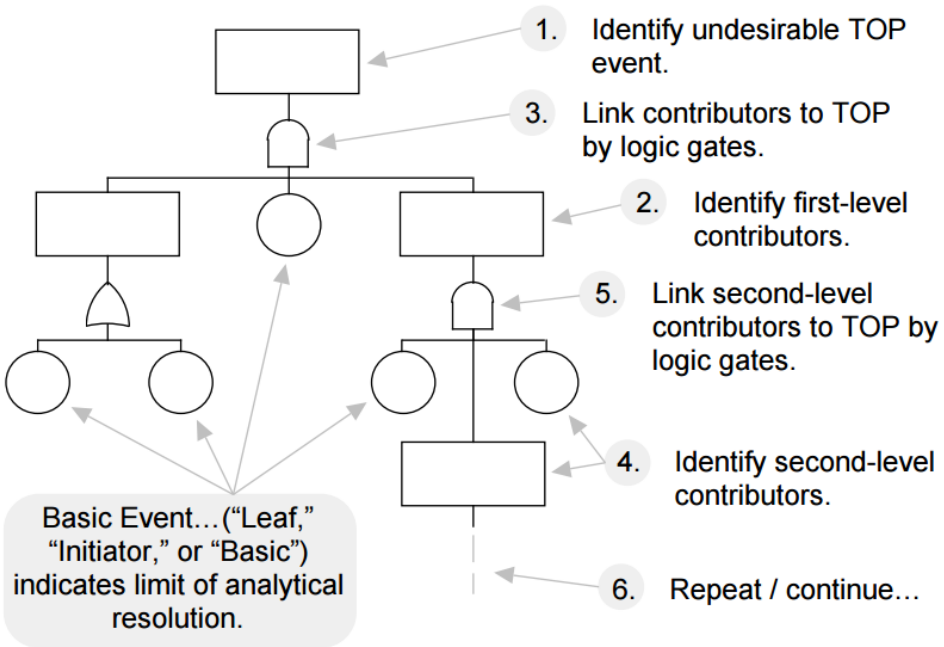


Figure 2.5: Structure of FTA.

[12]

The building of tree structure starts from the top and proceeds downward. Thus, FTA can be called a top-down analysis method. The first phase in FTA is the identification of TOP event. In this identification process, the investigation of earlier mission documents and the identification of failure contributors are important. After TOP event identification starts the second level contributor identification. In this tree structure, this TOP event should also link to the second level contributor by using logic and/or gates. After second level third level starts and this tree structure can be continued for as long as it is necessary. In other words, as long as all failure causes of the TOP event are found. [12]

2.4.2 Failure Modes and Effects Analysis

FMEA is an analysis method that has established itself as a part of reliability analysis. A *failure mode* is defined as a manner by which system failure can be observed. The main purpose of FMEA is to identify failure modes of the system, access their impact on the system and propose possible operations to prevent undesirable effects. FMEA aims to answer the following questions: What could go wrong with the systems and in a system creation process? How badly something could go wrong and what must be done to avoid failure? [13]

Typical flow of the FMEA [13]:

1. Identification of System and Functions
2. Identification of Failure Modes
3. Determination of Effects of the Failure Modes
4. Identification of Possible Causes
5. Documentation and Risk Reduction

In the first phase of FMEA flow, the identification of a system and functions that are under the analysis has to be done because it is necessary to know the system and functions that will be analysed. In phases 2 and 3, the identification and effects determination of failure

modes has to be done and in phase 4, the possible causes are identified. The final phase includes documentation and risk reduction activities which ensures that all observations are documented and all risk are eliminated or minimized.

FMEA can be applied in diverse environments from business management to spacecraft development. This analysis method is better known in hardware domain because it is easier to identify failure modes from a tangible product. In the hardware domain, the main task is to analyze the effect of these failure modes on the system. [13]

In a software domain, the main problem is that the software does not fail but the behaviour of it can be unexpected. That is the reason why the failure mode identification is hard in software context. However, application of this analysis method has increased in software side because the amount of software is increasing all the time. [13]

Failure modes, Effects and Criticality Analysis (FMECA) is an extension of FMEA and is also a reliability analysis tool where the main tasks are to identify potential failure modes of system design and offer possible mitigation solutions. FMECA also ensures that the design characteristics fulfill the reliability requirements and satisfy the needs of the customer. The main purpose of these two analysis methods is the same but FMECA differs from FMEA because it includes also criticality analysis. [14]

Chapter 3

Product Assurance in Space Technology

Product assurance, safety and reliability are in an important role in system development and especially in *mission-critical system* development. A mission-critical system can be defined as a system that can cause massive harm to the humans and environment if it does not work correctly.[4] In this thesis, presented mission-critical systems are *space systems*.

In space domain the main objective of product assurance is to ensure that all space products are safe and reliable as well as fullfill all mission objectives and requirements. *Space Product Assurance standards* offer guidelines for safe and reliable engineering and ensure that all product assurance issues are taken into account in development process. [15]

Space systems are long-lived, and their repair is difficult unless or even impossible. That is the reason why product assurance is emphasised in this kind of system development already in early development phases.

3.1 Space System Development

Space system is defined as an entity that consists of software, hardware and human resources that are needed in a space mission. Space system can be divided into two segments: *space segment* and *ground segment*. Space segment consists of the spacecraft and ground segment consists of controlling and managing system of the spacecraft and it is located on Earth. Figure 3.1 present how these segments are located in more detail. [16]

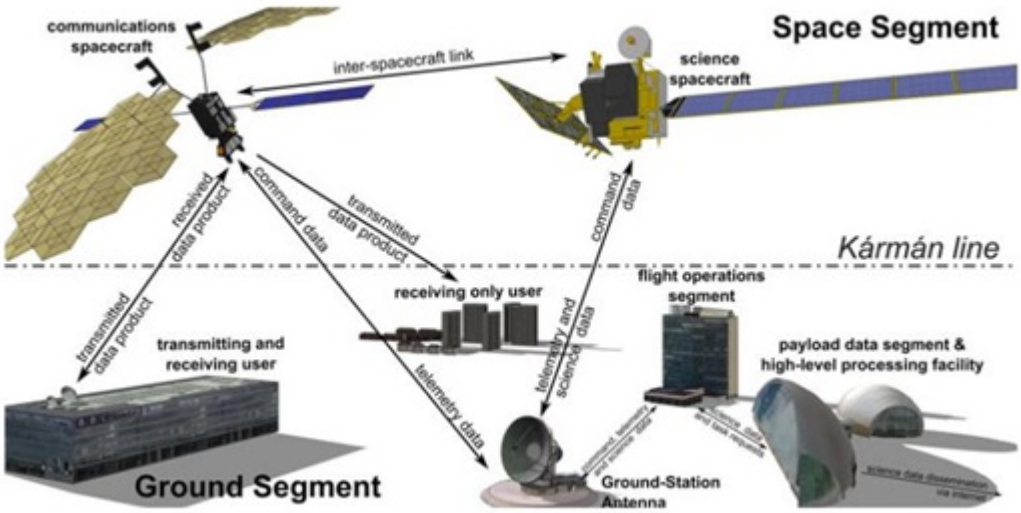


Figure 3.1: Space system segments.[16]

The ground segment is everything under the line called *Karman Line* and this segment includes all software, hardware and human resources that are needed for managing and controlling of the spacecraft. The ground segment can be divided into two subcomponents: *payload data segment* and *flight operations segment*. Of these two, the flight operation segment is independent in space mission and the main focus of it is in spacecraft controlling and commanding. Conversely, the payload data segment is accurately defined by mission goals.[16]

Space technology and system development follow standards that accurately define the guidelines and rules that allow producing safe and reliable space systems. In Europe, space standards are managed by *European cooperation for space standardization (ECSS)*.[16] The structure of ECSS-standards is presented in Section 3.2 in more detail.

Standards divide project life-cycle into seven phases. Theses phases include different actions as follow:[17][18]

- *Phase 0: Mission analysis-need identification*
  Supplier supports the customer to identify its needs and propose different possible systems concepts.

- *Phase A: Feasibility*

  The expression of customer needs should be finished and system solutions which respond to these needs should be proposed. The risks and criticality of these solutions should be identified. One of the proposed solutions is selected.

- *Phase B: Preliminary definition*

  Preliminary definition of selected system solution should be established. The supplier organization should demonstrate that the system solution meets the customer needs and technical requirements on schedule and follows budget limitations and organizational requirements.

- *Phase C : Detailed definition*

  Detailed definition of the system should define by system engineering organization. This organization should also demonstrate and evaluate its capability to meet the technical requirements that are specified to the system.

- *Phase D: Qualification and production*

  Development of the product is finished by acceptance and qualification.

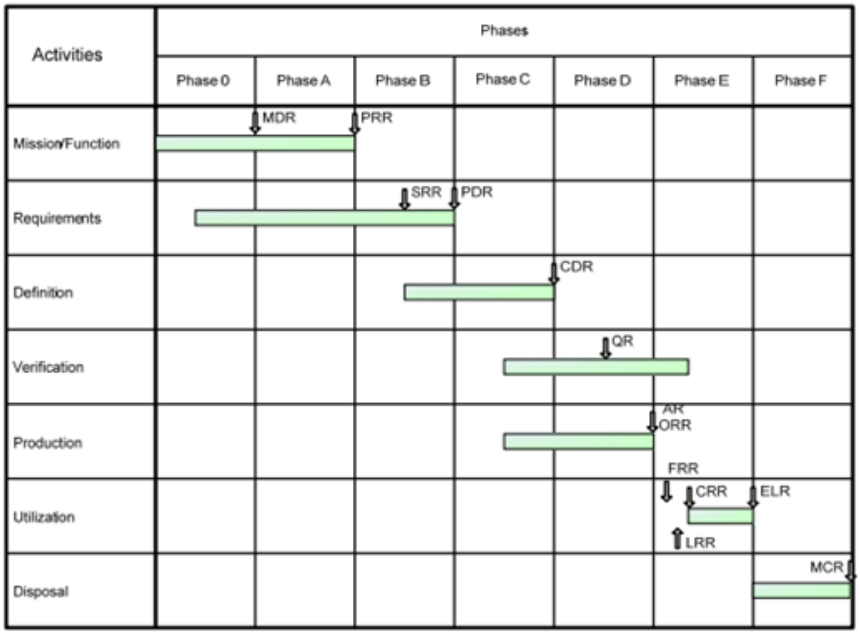
- *Phase E: Operations/utilization*

  Launch campaign is supported by system engineering organization. Includes activities which ensures that the dispose of launch is safe.

- *Phase F: Disposal*

  The end of the project. Includes maintenance and waste disposal activities that are documented and agreed upon in business agreement.

In addition to these activities, each phase should support the performance of milestones that are assigned to it. Each of these milestones define activities that should be done before the milestone. The most common milestones are presented in Figure 3.2.



MDR	Mission Definition Review
PRR	Preliminary Requirements Review
SRR	System Requirements Review
PDR	Preliminary Design Review
CDR	Critical Design Review
QR	Qualification Review
AR	Acceptance Review
ORR	Operational Readiness Review
FRR	Flight Readiness Review
CRR	Commisioning Result Review
LRR	Launch Readliness Review
ELR	End- of-Life Review
MCR	Mission Close-out Review

Figure 3.2: Space system development life cycle. [17]

Milestones MDR, PRR, PDR, CDR, AR, ORR, ELR and MCR are located to at the end of specific phase and the main purpose of these milestones is to review and evaluate project progression. If the progression is sufficient then the project can move to the next phase. At the beginning of each phase, requirements that are incomplete or contradictory should be resolved with customer.[18]

3.2 Space Standards

ECSS-standards are in an important role in European space industry because they offer all guidelines and requirements for all European space activities. Development of ECSS-standard system as early as in 1993 when the *European Space Agency (ESA)* and other

national space industries decided to replace old PSS standard system with a new one. The change was necessary because there was a need for a coherent and accepted standard system.[16][19]

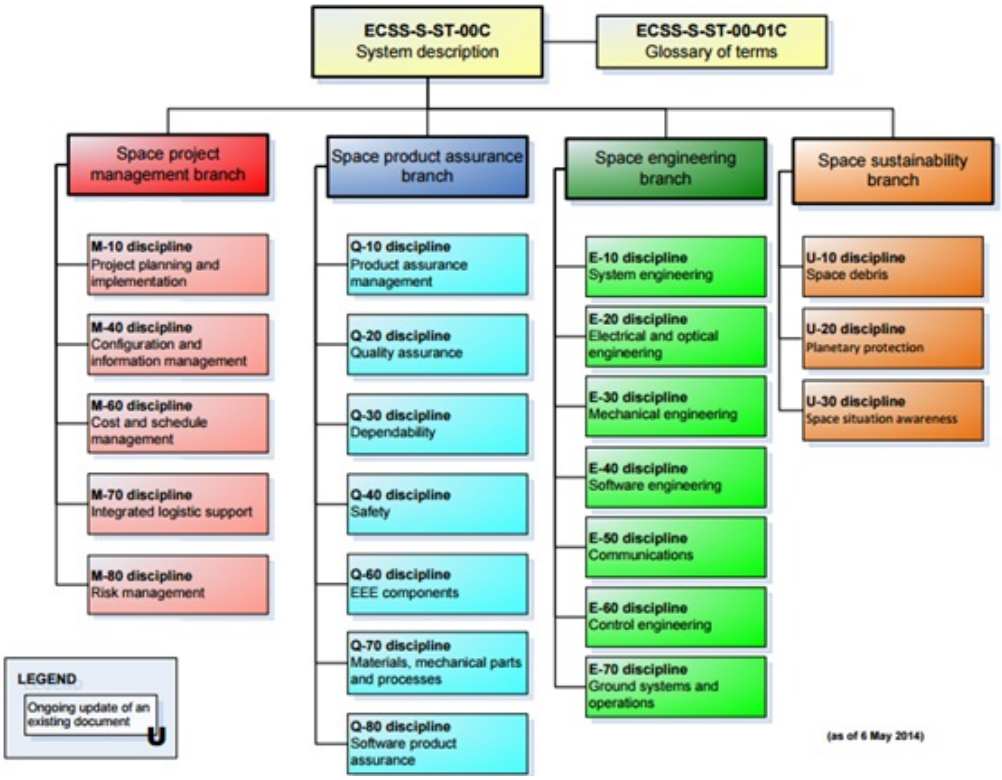


Figure 3.3: ECSS-standards.[20]

Figure 3.3 shows that ECSS-standards are divided into four main branches: Space project management; Space product assurance; Space engineering and Space sustainability. Each of these branches define its own disciplines and the space product assurance branch is divided into seven disciplines: Product assurance management; Quality assurance; Dependability; Safety; Electrical, electronic and electromechanical components; Materials, mechanical parts and processes; and Software product assurance. [20] The structure of product assurance standards are shown in Figure 3.4.

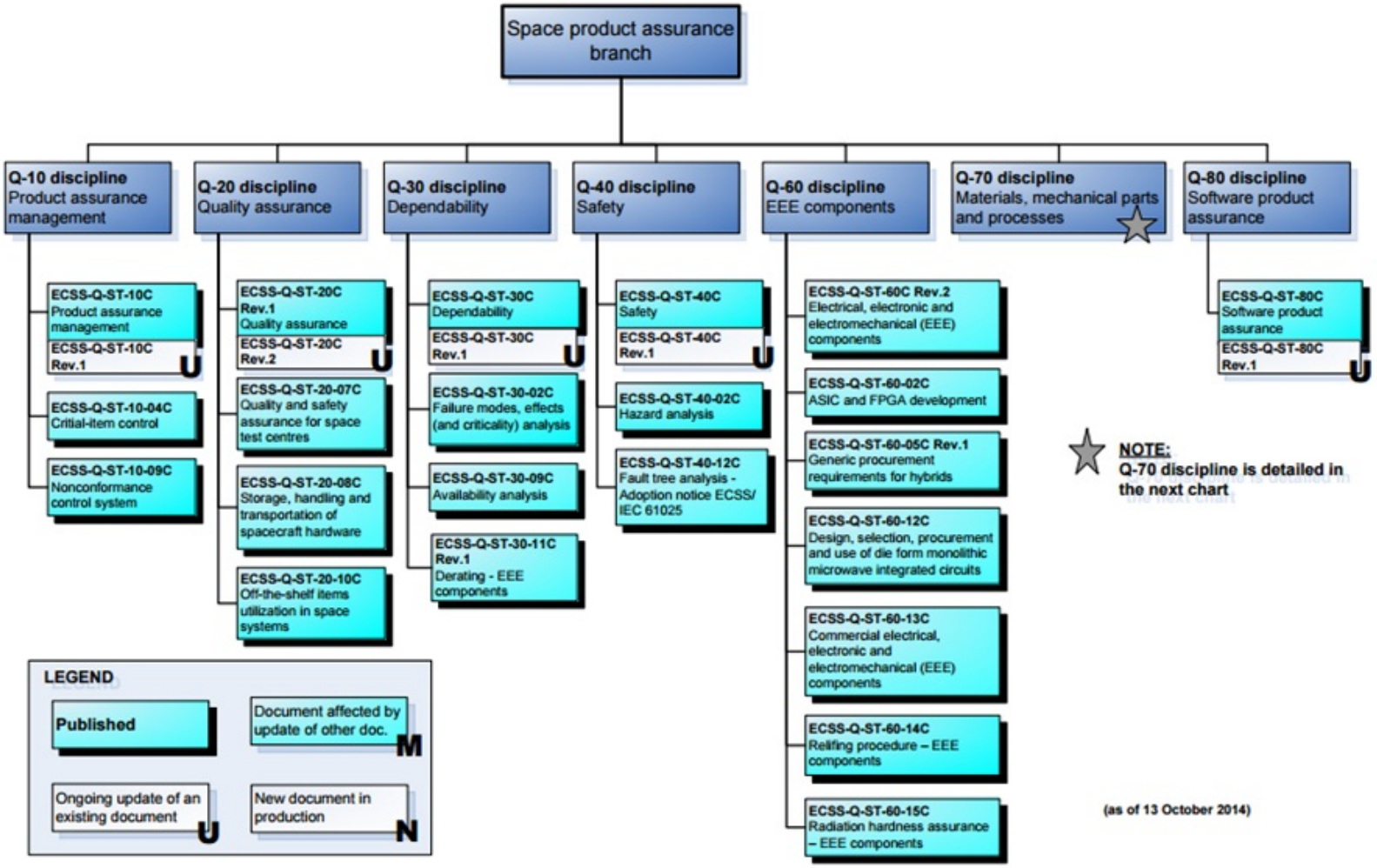


Figure 3.4: Space product assurance standards.[20]

The structure of product assurance standards is presented in Figure 3.4 and it can be seen that all seven disciplines define their own standards. Each standard is an independent document and focuses on different aspect of product assurance. Each product assurance discipline has its own standard code such as Product assurance Q10, Quality assurance Q20, Dependability Q30, Safety Q40, Electrical, electronic and electromechanical components Q60, Materials, mechanical parts and processes Q70 and Software product assurance Q80.

Product assurance disciplines that are presented in following subsections include guidelines, principles and tasks that should be regarded and performed during the project in specific phases. Product assurance standards do not offer clear process views because the content tells what to do and not how to do it. For this reason understanding of the content of product assurance standards can be difficult for the first time. In this thesis the content of these disciplines is presented briefly and if these standards are utilized in development environment it is necessary to delve into these standards in their original form.

3.2.1 Product Assurance Management

Product assurance management is represented in standard ECSS-Q-ST-10C and it defines product assurance program planning and implementation requirements. These requirements define different responsibilities such as planning; documentation; quality record; audit; and risk management responsibilities. The management of product assurance is included into the management of the project and it takes highest priority from the management of the organization. [21]

The main task of a supplier in product assurance planning is to identify personnel whose responsibility is to take care of the implementation and performance of product assurance management. Supplier also selects one person person to act as a *PA manager*. The PA manager has access to the highest management level and major tasks of PA manager are to communicate with project manager and ensure that schedule and contractual requirements are met and product assurance activities are performed. PA manager also communicates with the customer and makes sure that customer knows all product assurance matters.

Output of product assurance management planning part is called product assurance plan which is prepared, maintained and implemented by supplier and this plan should also be delivered to the customer. [21]

In product assurance program implementation the main focus is in product assurance disciplines and it is important that these disciplines respond to the contractual requirements. The activities of product assurance disciplines should be completed on schedule and should follow the task definitions that are presented in product assurance program plan. In product assurance program implementation part, PA manager has responsibilities such as ensure that all necessary documents are done and ensure that the products of the supplier are high-class. In product assurance management domain the risk management and critical item controlling make it possible that also the management level is updated of all possible unexpected situations and risks. As a result it is easier to coordinate risk management and critical item controlling with management functions. [21].

Product assurance management is in major role during the lifecycle of a project and it should be adopted already at the beginning of the project.

3.2.2 Quality Assurance

The main purpose of the quality assurance is to make sure that the project covers all mission definitions and the product works safely. Quality assurance principles and requirements are presented in standard ECSS-Q-ST-20C. Quality assurance should be adopted already at the beginning of the project and it is in progress throughout the project. This standard includes different principles for the design; verification; procurement; manufacturing, assembly and integration; testing; acceptance and delivery; and ground support equipment (GSE).[22]

The main purpose of quality assurance is to ensure that all activities are performed with high quality with these principles. In some organizations, quality management is supported with quality management system.

Design principle ensures that the product fulfils the requirements of specified quality level and design rules follow the project techniques. Design should be repeatable and reproducible because then it is possible to exploit the characteristics and performance of the product in other space system models and in serial production. Quality assurance also ensures that the verification actions are performed in a accepted way. [22]

Material are in central role in space projects and quality assurance standards offer principles to ensure that the component and material suppliers are identified, certified and trustworthy. The importance of these issue is highlighted in a situations where the old supplier can not deliver its components or materials anymore. In a situation of this kind, the responsibility of the quality assurance discipline is to offer guidelines and rules which confirm that also the new component and material supplier fulfils all quality requirements.[22]

The performance and planning of all manufacturing, assembly and integration activities should ensure that deliverable product is acceptable and all activities are performed in line with project schedule. Also all the testing work should be performed in accordance with project requirements and all testing activities should be documented properly.[22]

In delivery phase, it is important to ensure with documents and demonstrations that the product is built in accordance with the project requirements and works safely.

3.2.3 Dependability

Dependability assurance program and requirements are defined in standard ECSS-Q-ST-30C. Dependability assurance program is iterative and it is in use throughout the life cycle project. Risk management is in a major role in dependability assurance program and as said in Section 2.3 it consists of, for example, risk identification and risk elimination activities. Risk management in space system development domain will be presented later in Section 3.2.8.

Other important tasks of the dependability program are *dependability analyses* and creation of *dependability critical item list*. Dependability assurance program should also

ensure that the dependability assurance requirements are met and the risk minimization actions are running during the project. [23]

Dependability analyses should be implement throughout the project lifecycle. The identification and classification of all potential failure models could be performed for example using FMECA. Dependability critical item list is identified with dependability analyisis and it includes all single-point failures that have severity classification of catastrophic, critical or major. These classifications are presented in Table 3.1. This list also includes all items with a criticality number of six or greater, all items classified as catastrophic and products that are difficult or impossible to verify and check after integration. Documentation of all these items shall include justification for retention and approval of the customer. [23]

The main content of dependability risk assessment and control is to take care that all technical risk are identified and that these risks are categorized by their severity level of consequences. This dependability standard classifies severity levels of consequences to four levels according to Table 3.1 [23] Qualification of failure causes and origins is important. When risks are identified and severity levels, causes and origins are known, it is easier to develop actions to eliminate or minimize risks and their consequences.

Severity	Level	Dependability
Catastrophic	1	Failures propagation
Critical	2	Loss of mission
Major	3	Major mission degradation
Minor or Negligible	4	Minor mission degradation or any other effect

Table 3.1: Severity levels. [23]

Dependability assurance concerns the whole project lifecycle except Phase 0 where the tasks of dependability assurance are not usually needed. Phase A typically consists of dependability policy development and preliminary assessments and identification activities

Also preliminary dependability analyses are performed because the criticality aspect of each design option should be identified. In Phase B, preliminary design is supported by trade off studies and risk scenarios. Also failure effect severity classification and tentative dependability critical item list are created. The actions and performance that helps to eliminate and minimize risks are defined and criticality classifications of different products and systems are provided. One dependability task in Phase B is to support project Phases C and D, therefore in Phase B tasks of detailed design are planned and dependability plan is prepared as a part of product assurance plan. [23]

Phases C and D include more detailed design than earlier phases and for that reason dependability analyses and risk assessment actions are more detailed. Risk reduction actions are defined and dependability critical item list is updated in these phases. Risk reductions shall verify and tentative dependability critical item list created in Phase B should refine in a more detailed form. [23]

In Phase E, flight readiness review is supported and the impact of design evolution is considered. Dependability data is also collected during the flight missions. Phase F is the last phase and it evaluates the closure system operations of the systems and the consequences of them. [23]

3.2.4 Safety

Standard ECSS-Q-ST-40C qualifies safety technical requirements and program. Safety standards focus on protection of ground personnel and flight, associated payloads, launch vehicle, the general public, ground supported equipment, associated segments, public and private property and environment. The main purpose of safety assurance is to ensure that all safety risks in development, design, production and operational side are identified, minimized and controlled. Safety policy is supported using a safety program which contains risk analysis. [24]

Risk assessment and controlling is running throughout the project and it includes tasks such as: safety requirement allocation, identification of safety risks and hazards, conse-

quence severity evaluation and categorization, and safety risk and hazard reduction. [24]

In Phase 0, it is important to support the identification of safety risk sources. Requirement analysis and similar space mission analyses are essential because it is crucial to ensure that all safety requirements are clear and old similar space missions may offer information that helps in avoiding same mistakes in a new project.

In feasibility phase, the hazard analysis of the system design is in important role because system level safety critical functions requirements shall be identified in order that possible improvement propositions could be done. [24]

The identification of safety requirement and safety critical functions continues also in Phase B. In this preliminary definition phase this identification information is updated and also failure tolerance requirements are defined. Earlier phases support Phases C and D which consist of detailed definitions, qualification testing and production. Hazard analyses are detailed and identification information is updated again. These phases also ensure that technical safety requirements are updated and include the results of the safety analyses. Also the verification of these safety requirements shall be implemented during Phases C and D. [24]

The main purpose of Phase E is to ensure that every safety issue is controlled, maintained and within acceptable risks in active operations. Phase F includes tasks that analyze disposal operations and causes of these disposal operations.

Safety is one of the most important aspects in the product assurance domain and hazard and safety critical function identification and updating of the identification status is important in all phases. Failure and hazard analyses can be supported using methods FMECA and FTA presented in Section 2.4.

3.2.5 Electrical, Electronic and Electromechanical Components

Electrical, electronic and electromechanical (EEE) component standard, ECSS-Q-ST-60C, defines the most important issues that the requirements of electrical, electronic and electromechanical components should include. These issues are: component program management; component selection, evaluation and approval; procurement; handling and storage; component quality assurance; specific components; and documentation. [25]

Component programme should be implemented in context of electrical, electronic and electromechanical components. This programme ensures that all project requirements that are defined by the customer and supplier are in compliance with standard ECSS-Q-ST-60C. Supplier should prepare *component control plan (CCP)* and submit it to the customer. The main purpose if this list is to define all activities which ensure that the management process of the component plan is in line with the project objectives.[25]

Each component should be selected, evaluated and approved. In general, supplier should take care that project requirements, design requirements, production requirements and operational requisitions are met in the selection process. Together these requirements and requisitions ensure for example that quality level of components is high, components policy and budget are met and packaging and storage of the components are performed correctly. [25]

In selection process, data of old similar space mission can be exploited but still the supplier should ensure current data and application of each component. Already in design phase, qualified components should be used. There are also few obstacles in components selection and for example components that consists of magnesium, lithium or radioactive materials should not be used because these components can cause safety hazards.

The reliability of each selected component should be tested using for example stress tests. With tests of this kind it is possible to ensure that selected components are long-lasting which is an important property in space systems because the lifecycle of these systems is very long. All selected components should also be evaluated. In evaluation phase, sup-

plier should prepare evaluation plan and this plan should submit to the customer. Supplier evaluates all selected components and makes sure that quality, dependability, and functional and environmental properties correspond to the project requirements and needs. The results of component evaluation are collected to the evaluation report and this report is submitted to the customer. [25]

Component approval is possible after the selection and evaluation phases. The approval decision of the component should be based on all information which concerns the performance of a component. In approval phase, it is necessary to ensure that the component fulfils the dependability and quality assurance requirements. [25]

When a space product company orders electrical, electronic and electromechanical components from a supplier, customers should define component requirements within boundaries of electrical, electronic and electromechanical component standard. This is crucial because only then the supplier can deliver suitable components. The responsibility of the supplier is to ensure that the requirements of the customer are met. [25]

3.2.6 Materials, Mechanical parts and Processes

Materials, mechanical parts and processes (MMPP) discipline is presented in standard ECSS-Q-ST-70C. The structure of the standard is divided into four parts: General requirements, material control, mechanical part control and process control. General requirements mention that supplier should prepare, maintain and implement material, mechanical parts and processes plan and nominate a *MMPP manager*. The responsibility of manager is for example ensure that all materials, mechanical pats and processes that are used in manufacturing process of a spacecraft satisfy all functional requirements and restrictions of the project. The MMPP manager also obtains the status of material validation, mechanical part qualification and process verification. [26]

Material, mechanical part and process control parts are quite similar. Selection of materials, mechanical parts and processes is the first activity of MMPP controlling. The main flow of each controlling part is presented further in Figure 3.5.

Each controlling process starts with selection phase. Materials, mechanical parts and processes should be selected and they should fulfil mission needs and requirements. All materials and mechanical parts that are chosen should be tested in conditions that resemble final operating environment of the product. Criticality analyses should also be performed for all materials, mechanical parts and processes. If the criticality analysis identifies something important, such as unknown properties, the supplier should perform evaluation phase. If the evaluation phase is not needed then the controlling process moves to the next phase and this means that material control moves to the validation phase, mechanical part control to the qualification phase and process control to the verification phase. [26]

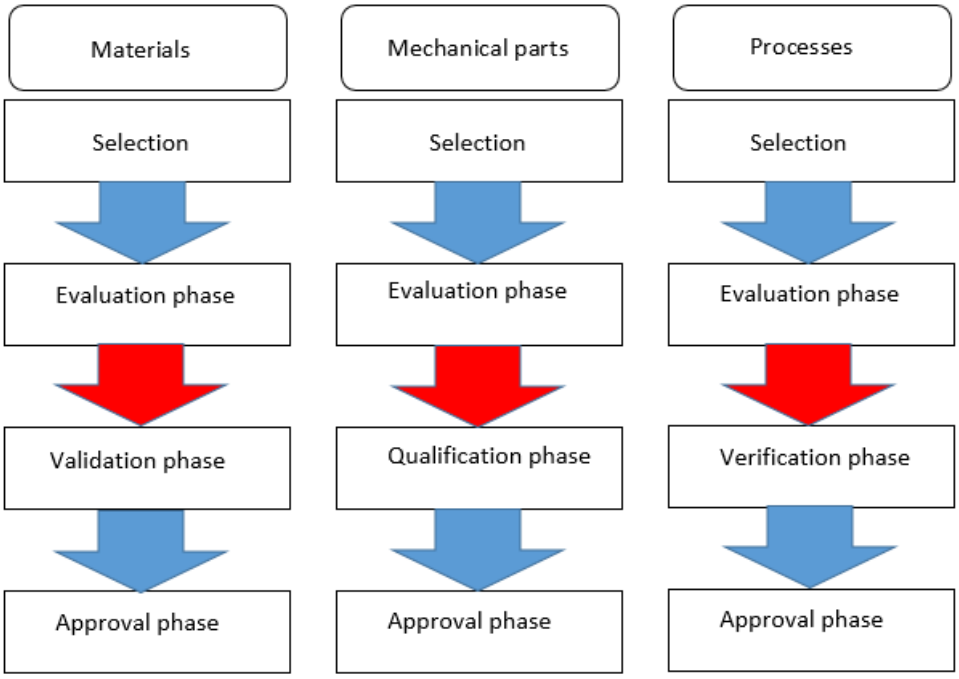


Figure 3.5: Material, mechanical part and process control flows.

Validation, qualification and verification phases ensure that materials, mechanical parts and processes fulfil the mission requirements. The final phase of the control process is evaluation phase and for example if the material does not fulfil the evaluation and validation requirements the material does not receive approval identification. [26]

As in electrical, electronic and electromechanical component standard also this standard involves several documentation requirements. The supplier should prepare and maintain documents such as *declare material list (DML)*, *declared mechanical parts list (DMPL)* and *declared processes list (DPL)*. These lists include all materials, mechanical parts and processes that are used in product manufacturing. These lists make it easier to determine the suitability of materials, mechanical parts and processes for specific application or product. [26]

Material, mechanical parts and processes actions start in Phase A with policy definition and product assurance task planning. In Phase B, requirements are defined and identified and all new needed items are also identified in this phase.

Phases C and D include identification of critical items and preliminary lists. Also tests are performed and test results are reviewed. Phase E is the final phase that includes Materials, Mechanical parts and processes activities. In this last phase, series manufacturing and identification of operational phase anomalies are supported.

3.2.7 Software Product Assurance

Standard ECSS-Q-ST-80C defines software product assurance requirements that should be followed in software product development and maintenance in space systems. The main purpose of this standard is to ensure that developed software satisfies customer requirements and this means that software should be developed to perform safely and properly. [27]

This standard is divided into three parts: software product assurance programme implementation, software process assurance and software product quality assurance. In software product assurance programme implementation part, the supplier should ensure that tasks and responsibilities are defined. Also all internal and external interfaces should be defined and documented. In software product assurance context there is a person in charge called *software product assurance manager/engineer*. One person should choose to this

role whose responsibilities are to report the project manager. He/she also has permission to maintain software product assurance programme according to software product assurance requirements. [27]

Software process assurance focuses on project life-cycle. According to the standard [27] project life-cycle related issues such as: phases, input and output of each phase, responsibilities and the role of customer in each milestone review should be defined and documented as a part of software assurance plan.

Process assurance part also includes guidelines for software analyses and criticality categorization. All software should be analysed and categorized according to its criticality level. Safety and dependability properties of critical software can be defined using analysis methods such as *Software failure modes, effects and criticality analysis (SFMECA).* This analysis method is the same method as FMECA but in software domain. Its main purpose is to identify failures caused by software.

In space systems product assurance is in a major role because it is necessary to ensure that both software and hardware are safe. In addition hardware-software interaction should work without any problems which means that software should react correctly to a hardware failure. Just for this particular purpose software product assurance standard defines analysis method called *Hardware-software interaction analysis (HSIA).*

3.2.8 Other standards

Product assurance standards define principles for different activities and as mentioned earlier those standards mostly explain what to do but not how to do it. Chapter 2 presents two different activities that support product assurance process: risk management and verification. In ECSS-standard system, verification standard is located in the space engineering branch and risk management in the space project management branch.

Verification

Verification is closely connected to the quality context and for that reason in an important part of product assurance process. In space development, verification activities are presented in standard ECSS-E-ST-10-02C. Verification consists of planning, execution, reporting, control and closeout activities which should be performed with high quality. [28] Verification process and activities are presented in Figure 3.6.

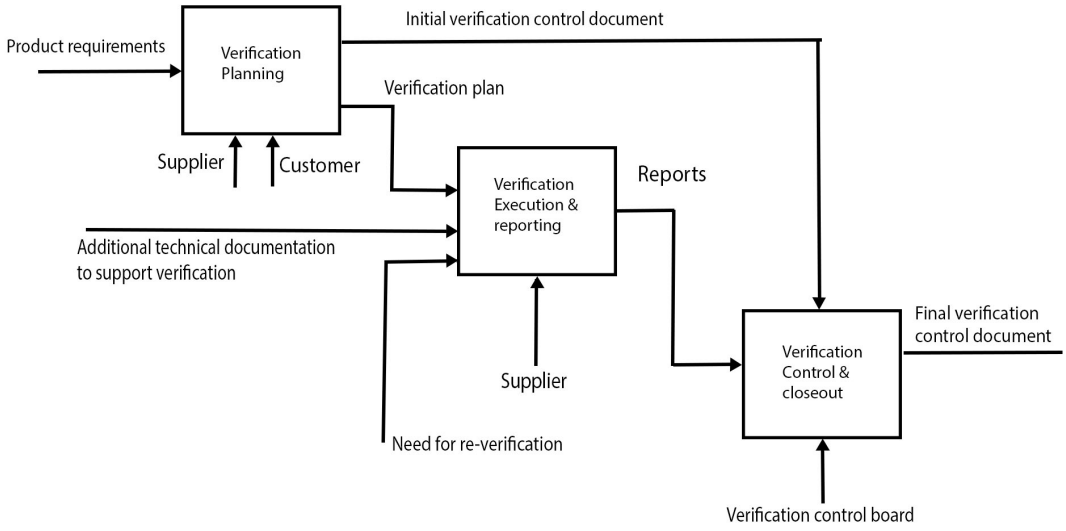


Figure 3.6: Verification process.[28]

Figure 3.6 shows that the verification process starts with a planning phase. In this phase the input is product requirements and output is verification plan. Execution and reporting phase starts when the planning phase is over and verification plan is ready. The conclusions of this second phase are documented in report form and these reports are input of the control and closeout phase. The results of this final phase are included in final verification control document which ensures that product is verified according to the requirements. This document should also be confirmed by customer.[28]

Verification is performed using one or more verification methods which are testing, analysis, inspection and review of design. All safety critical systems should be verified using testing methods and software and hardware components should be tested in their target

environment so that it would be possible to know how these components act in their destination environment.[28]

Risk Management

Risk management is introduced in Chapter 2. In product assurance context safety and dependability areas are important and risk management is an essential part of them. In space technology risk management activities are presented in standard ECSS-M-ST-80C. Figure 3.7 presents the process of risk management in more detail.[29]

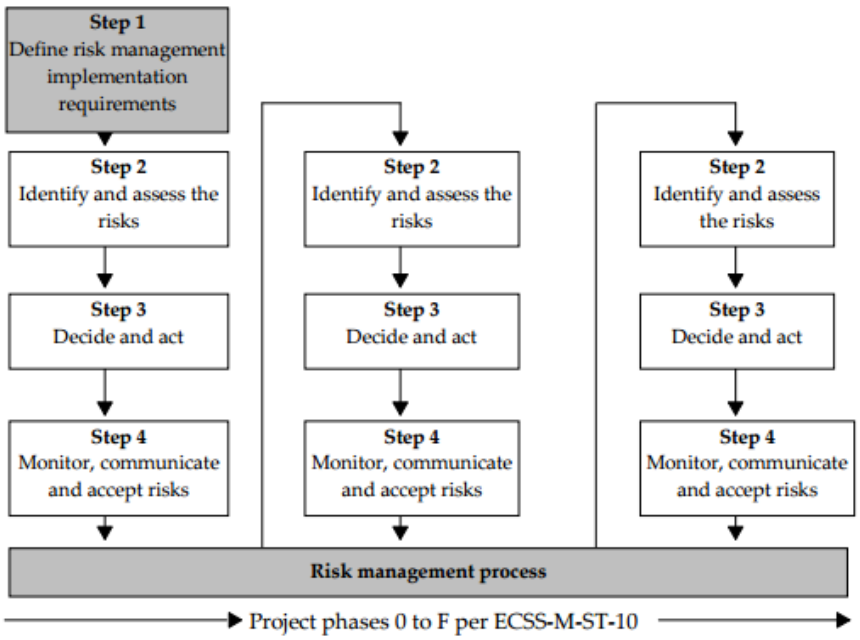


Figure 3.7: Risk management process.[29]

The structure of risk management in space domain is also quite iterative and risk management includes similar tasks listed in Chapter 2. Risk management standard specifies nine different actions that are performed continuously:

1. Define risk management policy
2. Prepare risk management plan
3. Identify risk scenarios

4. Asses the risks
5. Decide if the risk is accepted
6. Reduce the risk
7. Recommend acceptance
8. Monitor and communicate the risk
9. Submit risk acceptance.

In addition to safety risk, also cost and schedule risks are attempted to be avoided in this standard. Space projects are very expensive itself but problems might increase the cost even more. For that reason it is important to identify possible problems early and mitigate these risk before they occur. Risk can appear at any time so it is important to implement risk management continuously throughout the project.

Chapter 4

Agile Development

Agile development methods are popular in the software development domain. For that reason agile development is better known as agile software development. Agile emphasizes a fast response to change, transparency of the process, communication and customer collaboration. The pursuit of these properties is supported with agile principles, values and multiple methods.

Agile follows *iterative and incremental development (IID)*. *Iterative development* is a development model where the lifecycle of the project is divided into similar size fragments called *iterations*. The typical length of an iteration is between one to six weeks and usually activities, such as design, requirement analysis, programming and testing, are re-done in each iteration. If a system grows incrementally, it means that new features are added to the system in addition to the previous functionalities. Development of this kind is known as *incremental development*. [30]

The structure of agile development process is presented in Figure 4.1.

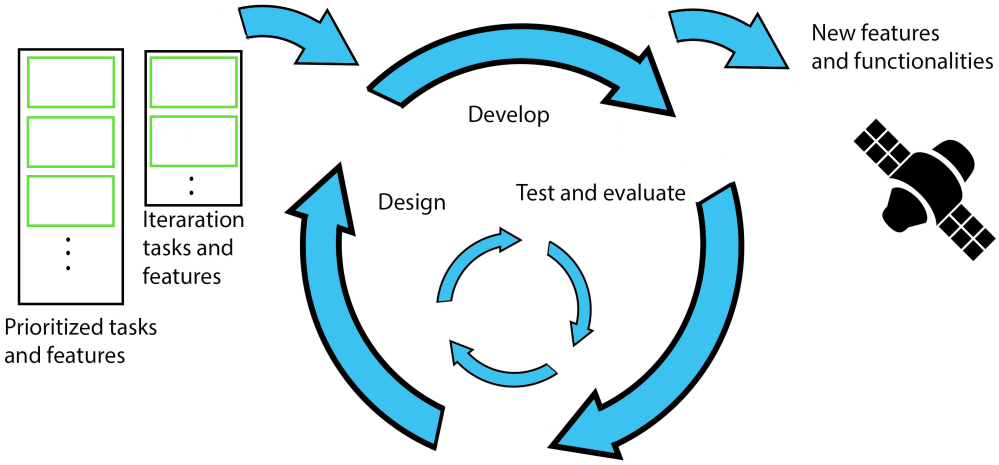


Figure 4.1: Structure of agile development process.(Adapted from [31])

The lifecycle of the product is cyclic. In Figure 4.1, big cycle is one iteration and smaller cycle represents a day. A *prioritized feature list* includes all tasks and requirements that should be completed before the project is over. Before every iteration, the features of the next iteration are selected from prioritized feature list and added to the *iteration feature list*. These tasks are performed during the next iteration and after every iteration there is a working product with new features and functionality. Results of the iteration should be presented to the customer.

4.1 Agile Values and Principles

Agile software development was introduced in 2001 when *"Manifesto for Agile Software Development"* was signed by 17 software engineers who were interested in agile and iterative methodologies. This agile manifesto defines four values and twelve principles which support iterative and transparent development.[30]

The four values of the Agile Manifesto are: [32]

1. ***Individuals and interactions*** *over processes and tools.*
2. ***Working software*** *over comprehensive documentation.*
3. ***Customer collaboration*** *over contract negotiation.*

4. ***Responding to change*** *over following a plan.*

It is also noted that while the items on the right have value too, the items on the left are valued more.

Agile embraces change and fast reaction to the changing customer requirements is a common feature of agile development methods. Customer collaboration, early delivery and iterative project structure ensure that after every iteration the customer can give vital feedback and report if requirements have changed. In agile, customer satisfaction is one key element and for that reason everything that gives value to the customer is important. Working software is a concrete product where the customer can see progress and where feedback of the product is a way for the customer to express satisfaction or dissatisfaction. That is the reason why the working software is a measurement of progress in agile.[30]

Agile methods do not forget individuals. Each member increases the value of the project and product with their own expertise. In programming work, which is a human activity, it is essential that developers are satisfied and working environment is efficient and productive. In agile, this kind of environment is a common project room where all team members work together. Working in this fashion advances communication because face–to –face communications are more common [30]

In addition to the common work space, the development team is a *self–organized team* which consists of independent engineers with cross-functional skills. The team divides all work tasks and this partition is based on interests and knowledge of each member [30]. For that reason members of the team are more motivated because their working task focuses on the targets of their interest. However, the team rotates these tasks very often because it increases the expertise level of the team.

The twelve principles are: [32]

1. *Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.*
2. *Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.*
3. *Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.*
4. *Business people and developers must work together daily throughout the project.*
5. *Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.*
6. *The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.*
7. *Working software is the primary measure of progress.*
8. *Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.*
9. *Continuous attention to technical excellence and good design enhances agility.*
10. *Simplicity –the art of maximizing the amount of work not done–is essential.*
11. *The best architectures, requirements, and designs emerge from self-organizing teams*
12. *At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.*

4.1.1 Agile vs. Waterfall

Waterfall is a quite old development method and already known in 1960's and widely promoted in 1970's. Nowadays agile is compared with the waterfall method quite often. [30]

The waterfall structure is presented in Figure 4.2. The structure reminds stairs and the basic idea is that the steps presenting project phases, lead down, meaning that the first phase is on the top and the final phase at the bottom. In addition, the waterfall model is a sequential approach which means that next stage starts after the previous is completely done and for example design phase cannot start before the analyze phase is completely done. [31]

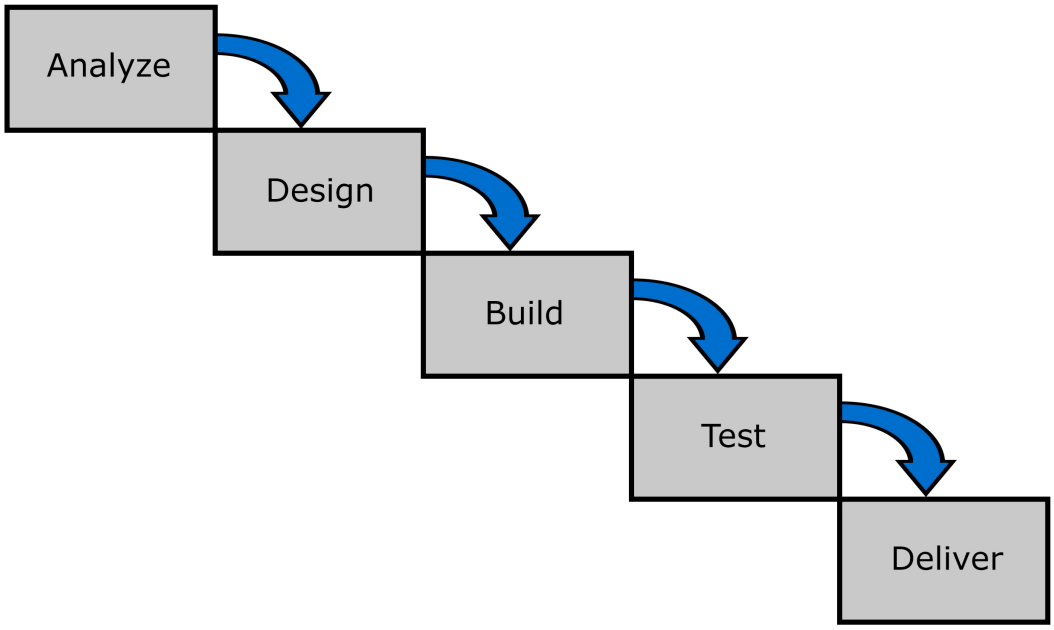


Figure 4.2: Structure of waterfall method.(Adapted from [31])

All methods as well as waterfall method, contain pros and cons and article [33] lists a few of them. The pros of the waterfall method are:

1. Clear requirement definition allready at the beginning of the project.
2. Specific period time of each phase.

3. Detailed documentation ensures quality.

4. Ease of implementation.

In the waterfall method, all customer requirements should be specified at the beginning of the project. It is contrary to the principles of waterfall if the next phase is started before these requirements are specified. However, this strict periodic time ensures that all phases are completely done and documentation in each phase ensures the quality of development. [33]

The waterfall is quite a simple and linear method so the implementation is easy. Because the process is simple also the amount of implementation resources is minimal.[33]

The cons of waterfall are:

1. Problems are not solved completely during each phase.

2. Responding to changing customer requirements is difficult.

In later project phases, problems accumulate easily if they are not solved in earlier phases. Fixing of emerged problems and failures becomes harder and more expensive in late phases and it may cause project failure. Also responding to the changing customer requirements is hard or even impossible. All requirements are specified already at the beginning of the process so all new requirements will not be implemented into this actual development process. [33]

Agile and waterfall methods are different. Agile offers an iterative development environment to the project, where the process consists of repeatable iterations. The waterfall method consists of stages and progress happens one stage at the time. Therefore, it can be stated that agile offers more time to ponder and understand requirements than the waterfall method.

Customers are more satisfied in agile projects than in waterfall projects. In agile, the customer is presented in every iteration and the customer is able to provide feedback after

every iteration. For that reason the customer can change the requirements easily and the development team can respond to these changes in an efficient way.[30]

Agile offers higher quality and productivity with lower risk while the waterfall method offer lower productivity and inferior quality with higher risk. The main reason for this difference is that agile methods require early testing while in the waterfall all testing work is performed at the end of the project. Agile tries to solve all problems early, it does not push them forward. [30]

Agile tries to offer transparency and flexibility to the development process. All unnecessary documentation is replaced with face-to-face conversations so all misunderstandings and mistakes are solved early with team members. Flexibility that agile offers might be one of the key reasons why the popularity of agile is increasing nowadays.

4.2 Agile Methods

Agile development covers many different methods. This thesis presents the following methods: *Scrum*, *Lean and Kanban*, and *Extreme Programming*. All of these methods follow agile values and principles. Even if all those methods follow the same values and principles each of them promote agility in their own way. The most popular of these methods is Scrum which is based on regular ceremonies. Lean and Kanban are based on waste elimination. Extreme programming is known for practises such as pair programming which aims to increase the quality and efficiency of a project. Each method is presented in their own subsection.

4.2.1 Scrum

Term *Scrum* is mentioned already in 1986 in a study ”The New New Product Development Game” written by Takeuchi and Nonaka . In this study Scrum was introduced as a method where products were created using overlapping phases and team was introduced as a small group with cross-functional skill set. [31][34] Nowadays Scrum is a well-known agile method and ”The Scrum Guide” presents the guidelines of Scrum. [35]

The structure of Scrum is presented in Figure 4.3. In Scrum iterations are called *sprints* and traditional length of one sprint is two to four weeks.

In Scrum, the prioritized feature list is called a *product backlog*. The name is different but the purpose is the same: the product backlog is a list of tasks and customer requirements that should be completed before the project is over and the product is ready. Before each sprint, the tasks and responsibilities that are planned to be done in the next sprint are moved to the *sprint backlog* which is the equivalent with the iteration feature list.[35][36]

As in Figure 4.1 also in Figure 4.3 the big cycle represents a sprint and smaller represents a day. According to agile principles, also in Scrum, the result of each sprint is potential deliverable product increment.

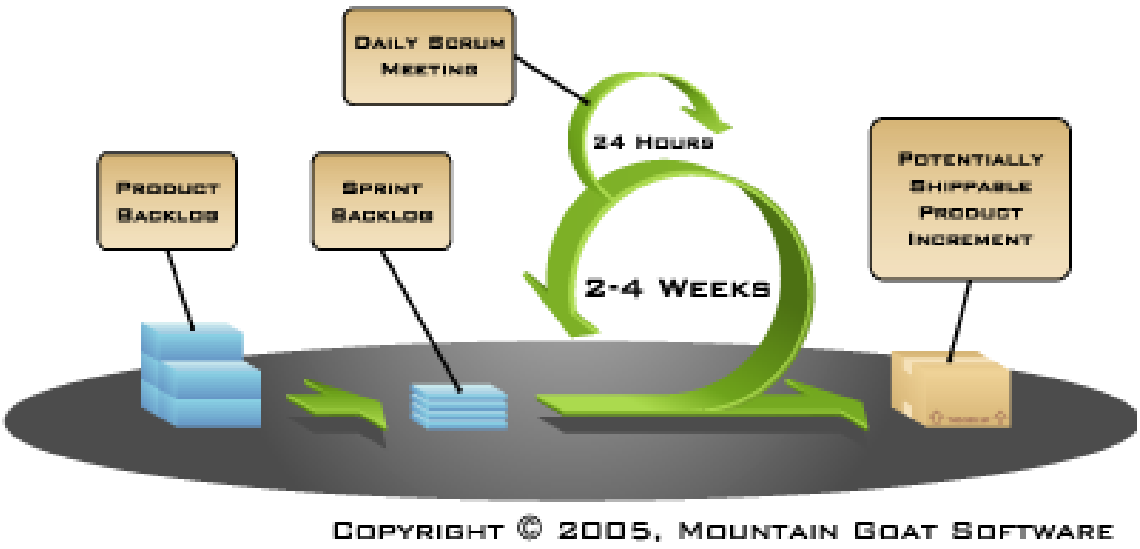


Figure 4.3: Scrum method.[37]

Scrum involves three different roles: *Scrum team; Product owner* and *Scrum master.* In Scrum, the scrum team consists of a product owner, Scrum master and development team. Each of these roles involves different duties and responsibilities which are presented next. [35][36]

The major responsibility of the product owner is to update product backlog. The product

owner should also ensure that the project vision is clear to everybody and there should only be one product owner who makes final decisions about the direction of product development and the priority of the developed features. [35][36]

The Scrum master protects the development team and takes care that there are no interruptions that disturb the working of development team. Responsibilities of the Scrum master are to ensure that the development team focuses on their sprint commitments and the Scrum master can also help the product owner to target the product backlog if the product owner does not know how to do it. The Scrum master is also facilitator of the meetings and ensures that person in the development team understands the project goal and works together as a team. [35][36]

The development team of Scrum is a self–organized team which consists of team members with different fields of expertise. Requirements of the product come from the customer but the technical implementation is the responsibility of the development team. This means that the members keep a brainstorming sessions together and decide what the best technical solutions are. [35][36]

The Scrum includes several regular meetings such as *Daily stand-up, planning, review* and *retrospective* meetings. In a planning meeting, the product owner and the development team discuss about the highest priority requirements and items. After that, they choose the requirements that are supposed to be done in the next sprint. Those requirements are then moved to the sprint backlog. [35][36]

In a daily stand-up meeting the development team members make their progress transparent to others. This meeting should only take 15 minutes and during this time each member tells what they have done since yesterday and what they are going to do today. In addition, they should also inform if they have met any obstacles during the working process. However, solving these obstacles are left outside the meeting. [35][36]

Next meeting is called review which is held at the end of a sprint and it is a great oppor-

tunity for the development team to get feedback from the *stakeholders* who consist of all relevant people with interest for the project such as customer and product owner. In the review meetings Development team; product owner; Scrum master and interested stakeholders are present. The main goal of this meeting is to discuss the product and changes that might needed to be done. In addition, the development team should also demonstrate their sprint accomplishments to the customer and stakeholders. [35][36]

The last meeting is retrospective. The purpose of retrospective for the development team is to discuss their working in the previous sprint. They go through all things that worked well and those that did not work so well, and after that the development team thinks what changes should be made in their working habits for the next sprint. [35][36]

In addition to all roles and meetings, Scrum uses tools that ease the process management. Burndown chart is a good way to follow progress status because in this chart work progress is presented as a function of time so it shows directly if the project is not on schedule.

4.2.2 Lean and Kanban

The key principle of *Lean software development* is waste elimination and continuous improvement. The elimination of waste is efficient only if the value chain is made visible. Lean was first introduced in 1950's in Japanese manufacturing industry and later Tom and Mary Poppendieck introduced Lean in a software development domain. Lean involves seven principles: [38][39]

1. *Eliminate waste*
2. *Build quality in*
3. *Create knowledge*
4. *Defer commitment*
5. *Deliver fast*

6. *Respect people*

7. *Optimize the whole*

Waste is everything that does not increase or produce value to the customer. [39] In Lean, features should not be built only for good measure because features of this kind are waste. Quality should be a part of the product and developers should learn something new all the time because continuous improvement increases the value to the customer.

In Lean, fast reaction to change is crucial, so restrictive decisions should be made as late as possible because then those restrictions do not limit future development work. Fast delivery is also a major issue in Lean and there are only a few things that should be done at the same time. The team can affect their work so a reciprocal respect is important because it makes achieving goals easier. The benefits of final product and customer satisfaction are good measurements and help optimize the whole.[38][39][40]

Kanban approach is a well-known tool of Lean and it is a method that helps project teams to modify their actions and performance in line with the Lean principles. Kanban was introduced already in the 1950s in Japanese manufacturing industry and in 2004, Kanban was introduced in software development domain by David J. Anderson. The most significant goal of Kanban is to understand the value chain. Workflow visualization helps to understand the drifting of the value in the system. Kanban consists of five principles [38][39]:

1. *Visualize the workflow*

2. *Limit work in Progress*

3. *Measure and manage flow*

4. *Make process policies explicit*

5. *Improve collaboratively (using models and the scientific method)*

Word Kanban is Japanese and means ”signboard”. In Kanban one good way to visualize the workflow is a *Kanban board*. The simplest Kanban board consists of three columns

such as: To Do; Doing and Done. These columns represent the process and phases of the work. [40]

4.2.3 Extreme Programming

Extreme programming (XP) is a lightweight methodology in a software development domain. As other agile methods, also XP emphasizes teamwork, customer satisfaction and flexible response to change.[30]

XP is a very team oriented method. Development team should consist of ten or fewer engineers and customers should work with the team as much as possible. Fast and efficient production is one key characteristic of XP. For that reason iterations are short and the traditional length of the iteration is one to three weeks. Continuous testing and integration are in a key role in XP and unit tests and acceptance tests should be written for all codes and features.[30]

These main objectives are supported with different practises. Kent Beck and Cynthia Anders introduce many different practises in their book [41] but traditionally XP is well known for its twelve programmer relevant practices as follows. [30] [40][41]

1. *Planning game*

   Planning game defines the scope of the next release. The length of this session varies from a half day to a day. In this session, customer writes *story cards* also known as *paper index cards*. To this card, customer describes features briefly and after that the development team evaluate the content of the cards. Finally, the customer decides which features are selected for the next iteration.

2. *Small, frequent releases*

   Fast delivery with small fragments increases customer satisfaction. The customer can give feedback more often and small fragments ensure that the customer can more easily maintain an overall picture of the software system.

3. *Metaphors*

The system is build using a common language called metaphor. System metaphors describe the architectural themes which help in design communication.

4. *Design*

   In XP, the design should be kept simple to allow future changes. Sometimes future changes are necessary and simple design helps to avoid the integration of general software components that are not required in the early phase of a project.

5. *Testing*

   Continuous testing and unit tests ensure that the software is working and the quality of software is high.

6. *Frequent refactoring*

   The software will be kept as simple as possible and improvements are made by frequent refactoring.

7. *Pair programming*

   Each code is programmed by two programmers who work on the same computer and rotate the input devices. Pairs may also change because different programming tasks need different expertise. Pair programming emphasises quality because two programmers on the same computer can handle software bugs and problems in a more efficient way than one programmer.

8. *Team code ownership*

   The whole team owns the code so each programmer pair is able to make necessary changes.

9. *Continuous integration*

   All code is continuously integrated and all unit tests and acceptance tests should be run.

10. *Sustainable pace*

    The amount of working hours should be constant, about 40 hours every week. Especially, overtime should be avoided. Common rules help the team to achieve the best possible result.

11. *Whole team together*

    Programmers and customers should work together in the same project room. The main purpose is that at least one customer works either fulltime or less with the team. These customers are experts who make decisions regarding their requirements and priorities.

12. *Coding standards*

    In addition to team code ownership, continuous refactoring and changing programming partners, team members should use similar programming style.

In addition to these twelve practises XP defines five core values [40][41]:

1. *Communication*
2. *Simplicity*
3. *Feedback*
4. *Courage*
5. *Respect*

Frequent communication with customer and team mates help to keep the design simple and clear. Many problems in the project are originated from poor communication. Increasing amount of communication allows the team members to know what other members are doing and what is the updated state of the project. Communication level can be increased using pair programming practice because two programmer on the same computer are in continuous social interaction. Also close customer collaboration increases the communication between customer and development team. [30][41]

Simplicity means that XP aims for a situation where the simplest thing that could work is done. Changes are easier to make in future if the previous structure is simple. It is unnecessary to develop complex features in early phases if there is no certainty that all this complexity is needed. One good example of simplicity are simple story cards as mentioned in the context of planning game. Because these cards include a brief description of

tasks and features formal artifacts are avoided.[30][41]

Feedback gives essential information about the system process state. Daily testing is one good example that gives essential feedback about the product.[41] Early delivery and fast reaction to the changing requirements increase the customer satisfaction and customer is able to give feedback more often.

In XP, the team accepts the actual state of the product and team encourages making changes if necessary. Responding to changes is easier if the team respects the vision of the customer and customer respects the expertise of the team. [40][41]

4.3 Agile in Embedded Systems Development

Space systems are critical embedded systems that consist of software, hardware and mechanical parts. Traditionally, embedded systems are application specific which means that the system performs predefined tasks. Agile is traditionally used in software engineering. However, the utilization of agile methods has increased also in other development fields such as in embedded system development. Still, the applicability of agile development might be a challenge in an embedded domain. The main reason is that the embedded development includes characteristics that are not included in a software domain.[42]

Article [42] examines agile principles in the embedded system development domain and offers an alternative way to interpret agile principles. The article defines four key restrictions that make agile embedded development challenging.

C1 Need for system level documentation Especially in highly standardized technology industries the need for documentation is high. The main purpose of documents is to ensure that all standard guidelines are observed.

C2 Hardware-software interdependences In embedded systems, software, hardware and mechanics development are dependent of each other. However the development

process of hardware and mechanics is realized in a longer time cycles because the tasks of the mechanical and hardware components are hard or even impossible to divide into smaller subtasks. This is a challenge for iterative development.

C3 Heterogeneous teams with different skillsets. Embedded system development employs experts in many fields. The main challenge is that these experts have different skillset and terminology. Experts with very different backgrounds may not fully understand each other and this may cause misunderstandings which can cause mistakes in a development process.

C4 Inflexibility due to real-time functionality. In embedded systems, software should perform functions in real-time, in a certain time window and in a predictable way to avoid major failure. This correlates to design work because synchronisation between different parts of embedded systems has to be maintained to ensure system functionality. In such embedded systems, speed and power consumption are emphasized over modular design and readable code.

Because space systems are embedded system these previous challenges and especially challenges C1, C2 and C3 offer good starting point to the creation process of the survey questions that are presented later in Section 5.2.2.

Chapter 5

Survey: Product Assurance in Agile Space Systems Development

In this thesis, the survey part is based on report ”SPACE-2000, Development of PA/QA for space instruments” [1] which was made in 1996 by AL Safety design. SPACE-2000 was an international space technology program and the main purpose of the report was to clarify how Finnish space companies were implementing product and quality assurance into their projects. The report consists of survey part and company visit part. The survey investigated possible potential problems and development needs regarding the practises of the space product and quality assurance. The purpose of visiting companies was to collect more detailed information about the product assurance processes of companies. In addition to the results of the survey and customer visits, the report also includes theoretical introduction that briefly presents product assurance disciplines.

Now the purpose is to update the previous survey and clarify how committed the Finnish space companies are to following product assurance disciplines and how clear product assurance standards are for respondents. In this updated survey, agile is as a small supplement and the purpose is to find out how do these developers feel about can agile provide assistance to the product assurance processes.

5.1 Research Method

Product assurance in agile space systems development survey started in September 10th 2015 and ended September 25th 2015. The survey was held as a web survey using Webropol service. The link of the survey was sent to the respondents by a contact person of Tekes. In addition, the survey link was sent to the partners of AgiSpacES project.

5.2 Background of Survey Questions

The survey platform is attached as an appendix at the end of this thesis. The survey consists of about forty questions which form two main sections: Product assurance and product assurance and agile methods. The survey begins with three questions that collect organizational and work background information as follows:

- Name of the organization, Question 1.
- Responsibility of the respondent in organization, Question 2.
- Work experience of respondent, Question 3.

The results of the survey are anonymous so the identification of an individual respondent is impossible even if they answered to first three questions.

5.2.1 The First Section: Product Assurance

The actual survey questions start after the third question with heading product assurance. The first section examines the performance of product assurance activities inside the organization where the respondent works. This part includes similar questions than the previous survey in 1996 and the survey is allocated to the people who presumably work with space product assurance. In addition to this assumption, Question 4 examines how often the respondents work with product assurance.

Even if the respondents work with product assurance quite often it might be possible that following the product assurance standards is hard for them and for that reason the Question 5 examines how difficult or easy it is to follow the standard during the project.

Product Assurance Disciplines

Product assurance disciplines presented in Chapter 3 offer guidelines for successful product assurance. Because these disciplines are relevant in space technology they are included in the survey questions as follows:

- Knowledge of each discipline, Question 6.
- Importance of each discipline, Question 7.
- Implementation of each discipline inside the organization, Question 8.
- The amount of product assurance tasks, Question 9.
- Improvement need of disciplines, Question 10.

In Questions 6, 7 and 8, the respondents can evaluate each of seven product assurance disciplines and the main purpose is to find the most important and well known disciplines and outline how the organizations implement these disciplines.

The amount of work tasks might be quite high in a space project and because product assurance is a relevant part of a space project the load of the product assurance task is researched in Question 9. Each product assurance discipline defines its own tasks and the performance of these disciplines might need some improvements. In Question 10, the respondents are able to name at maximum two this kind of disciplines.

Schedule, Cost and Resource needs

Space project consists of different elements that affect cost, schedule and resource needs of the project. For that reason it might be necessary to give some extra attention to evaluation process of these needs. Questions 12, 13, 14 and 15 examine the cost, schedule and resource needs issues as follows:

- Ease of evaluation of needs, Question 12.
- Major challenges of the evaluation process, Question 13.
- The need of a tool that eases the evaluation process of needs, Question 14.
- Possible properties of tool, Question 15.

Only those respondents that answered "No" to the Question 12 are able to answer also in Question 13. In Question 13, those respondents list challenges that make the evaluation process harder. The main purpose of Question 14 is to examine if the respondents feel that there is a need for a tool that might ease this evaluation process. In Question 15 there is an optional possibility to list possible properties of the tool.

Analysis Methods and Risk Management

Failure analysis methods that are presented in Chapter 2 are essential part of the quality system of the product. Question 16 asks that do respondents use failure analysis methods in their organization and if they do use, are they able to name what method they use. Risk management can be supported with these failure analysis methods and is also an important part of the product assurance process and should be implemented as a part of the development process. The implementation of risk management is examined in Question 17 and respondents are also able to justify their answer if the think that risk management is not implemented as a part of process.

Space products consist of hardware and software components and the collaboration of these components should be seamless. Product assurance standards present the HSIA

method that is mentioned in Chapter 3. The purpose of this analysis method is to support the integration process of software and hardware. However it might be possible that this integration process is not supported enough and this issue is observed in Question 18.

Product Assurance Requirements

Several survey questions are seeking information which proves that the product assurance process contains problems that can be supported with agile methods. Good examples of questions of this kind are Questions 19 and 20 which address to the changes and updating need of product assurance requirements during the project.

Product Assurance Management

The management of product assurance is in a major role in space product assurance and especially PA manager is in an important role. In Question 21 the importance of this role is studied and in Question 22, the most important tasks and responsibilities of the PA manager are listed.

5.2.2 The Second Section: Product Assurance and Agile Methods

The product assurance and agile methods section starts with preliminary Questions 23, 24 and 25 which examine the following issues:

- Familiarity of agile methods, Question 23.
- Knowledge of agile methods, Question 24.
- Usage of agile methods in organization, Question 25.

The rest of the questions are designed to support the viewpoints and statements of following subsections.

Project Lifecycle

The research of space product assurance and agile begins with lifecycle division. Because agile based on iteration and space system technology itself offers very complex development environment due to precise specifications it is necessary to suggest the division of the space product lifecyle to the respondents at the beginning of the second section of the survey. The purpose is that the lifecycle is divided according to following Figure 5.1.

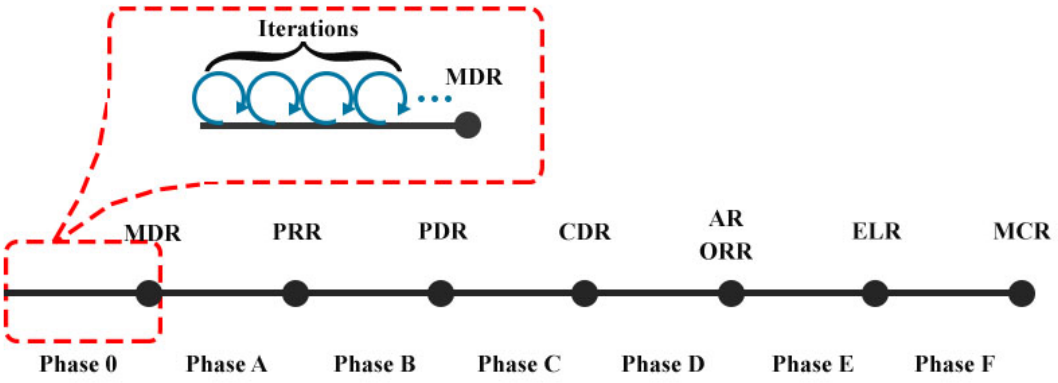


Figure 5.1: Agile development between milestones.

In a space project, development life-cycle is divided into seven phases and space system development slightly reminds the waterfall model because the earlier phase should completely be done before the start of the next phase. The main purpose is to show that it is possible to apply agile development into space system development. Agile development can be utilized between milestones as shown in Figure 5.1 and the opinion of the respondents about division of this kind is asked in Question 26.

In space development, requirement definitions are made at the beginning of the project life-cycle. In agile, all strict specifications are tried to be made as late as possible. Even if standards require definitions already at the beginning of the project life-cycle, it is important to remember to keep all definitions as minimal as possible but in accordance with requirements.

In Section 3.1 it is mentioned that requirements are updated at the beginning of each phase. If the life-cycle of the product lasts three years, then the time between two milestones is few months which is quite long time. In system development where the customer and technical requirements act in a key role, it would be necessary to update requirements more often and for example after every iteration. Question 27 considers this kind of requirement updating and the main point is to examine how much it would affect to the safety and reliability properties of the final product. Definition related problems are also observed in article [19] which examines agile software development and ECSS-standards in a Swedish Space Corporation. In this article early definition problem is solved with the sprint planning meeting. In this organization sprint planning meeting is a platform which is used for planning and discussion about documentation, inputs and outputs that are produced in each sprint. Also the presence of the customer and increased customer feedback ensures that all specifications and technical design meet the requirements.

Risk Management and Verification

Risk management and verification are presented in Chapters 2 and 3. Risk management is an iterative activity also in the space development domain so according to standards it is possible to implement risk management as a part of iterative space system development. Even if risk management is an iterative activity itself, it might be important to ask how much iterative developments support the risk management. In Question 28, the respondents are able to give their opinion about this issue.

Verification has been included in the survey because it is tightly bound to the quality aspects of a product. The verification process reminds the waterfall model as shown in Figure 3.6. Question 29 considers verification and the meaning of this question is to examine that if it is possible to improve the performance of verification and testing activities if these activities are performed in each iteration.

Challenges of efficient verification of space systems are also examined in an article [43]. The article introduces challenges such as missing customer requirements and limited per-

formance of verification experts in early phases. These challenges support the idea that it might be necessary to support verification process of space products with agile. Agile practises that would solve these challenges are presented further in Section 6.2.2.

Process Management

Challenge 2 presented in Chapter 4 mentions that lifecycle of hardware components is longer because in hardware development, tasks are harder to divide into smaller subtasks. In space technology, hardware process is highly standardized and all electrical, electronic and electromechanical components and materials, mechanical parts and processes should be approved. Sometimes there might be problems in material and component delivery and delays might occur. Also in C3 in page 50, it is mentioned that in embedded development team members have different skills because software and hardware developers have different education and work history. Different backgrounds might cause misunderstandings within the team because they do not have common terminology. In agile, the rotation of tasks is very typical but in embedded development it might be complex because the educational backgrounds of hardware and software developments are different. For that reason the interest in the work of others is essential [40] and updating of the overall picture is emphasized.

Agile can offer good practises to help developers to understanding the general view of the project better and produce more compatible hardware and software components. Kanban board is useful tool to perceiving the overall view. Other tool that can be used to update the status of the process is burn-down chart presented in Chapter 4 which express the progress as a function of time. Both of these tools are presented in Question 30 and the main purpose is to examine that would these tools ease the evaluation process of schedule and resource needs of the product assurance tasks.

Question 32 concentrates on understanding the overall view. The main aspect in this question is to examine that would the better understanding of the overall view increase the quality, safety and reliability of the product.

The project would be easier to manage if the amount of the simultaneous tasks is low. In this survey the number of simultaneous working tasks is examined in product assurance manner and the main purpose of Question 33 is to figure out that do the respondents feel that it is easier to focus on product assurance tasks if the number of simultaneous working tasks is lower.

Status updating can also be handled with meetings and for example in daily stand-up meeting all team members receive the updated overall view of the project status and potential problems come to light. For example if there is a problem in software and it needs several changes then also hardware developers know that they have to check if there is a need to make some changes to their hardware component. Without this kind of communication between hardware and software development teams it is possible that hardware and software components do not work together safely. If this problem is not noticed until in the late phase of the project it might cost a lot of money or cause harm to the environment or humans.

The usefulness of the daily stand-up meeting is asked in Question 37. This question observes that is it possible to support the product assurance process by organizing this kind of status meeting every day. Other meeting that is presented in process management perspective is the review meeting. In Question 38, the review meeting is presented as an opportunity to increase the reliability and safety of the product and prepare the project for the milestones.

Documentation and Customer Collaboration

Space system development is highly standardised and space standards offer guidelines and activities for the successful system development process. There are several activities that can be supported with agile and documentation, mentioned in C1, is a good example of this kind of activity.

Documentation is a huge part of space system development and especially in the product assurance process. In the selection and evaluation processes of components and materials documentation is a key element between the supplier and customer. The needs and requirements of the customer are essential if development of safe and reliable products is desired. For that reason outcome of the project would be better if the customer and supplier also communicate in other ways. Agile allows that the amount of customer collaboration increases and it ensures that the customer collaboration is tighter and for example face-to-face conversations are more common. Agile methods such Scrum and XP present good activities, such as review meeting and whole team together practise, that support customer collaboration.

Customer collaboration is also a good way to eliminate unnecessary documentation such as emails that are produced by customer-supplier communication. Still space standards define a huge number of documents that should be prepared. This documentation activity can be supported using backlog. If all or almost all product assurance activities are recorded to the backlog it might ease documentation process because it is easy to check from backlog which tasks are already done and can be included in documentation.

The topics customer collaboration and documentation are included in the survey in Questions 31, 35 and 36 as follows:

- Product backlog would ease the documentation process, Question 31.
- Affects of the customer collaboration to the product assurance, Question 35.
- Optional comments about customer collaboration, Question 36.

Software Development

Chapter 4 presents the XP method which tries to improve quality itself. Its twelve practises provide a good platform for a high-quality software product. For example, pair programming is a good practise to program better quality code in a more efficient way and

common metaphor ensures that all developers use the same vocabulary in design work. Simple design ease future development work and allows changes. Testing work in each iteration ensures that the product is working and meets the requirements.

The survey examines that how much the practises of XP would increase the quality of the software product in Question 34.

Product Assurance Activities and Agile

Final two survey Questions 39 and 40 are conclusion questions and these question examine that there are product assurance activities that can be supported with agile methods and are there product assurance activities that can suffer from agile methods. Respondents were also able to give additional feedback after the survey in Question 41.

Chapter 6

The Results of The Survey

The number of respondents in this qualitative survey is quite small because space technology is small industry in Finland. Developers from five different space organization responded to the survey and the total number of responds is eight. The number of respondents vary between different organizations. In one organization, the number of the respondents were greater then one and for that reason answers of this organization are joined to represent the average response of the organization.

6.1 The Results of The First Section

Commonly respondents feel that they work with product assurance a few times a month and following the product assurance standards is quite difficult during the project. The reason why following the standards is quite hard can be explained by the fact that these developers work with product assurance too rarely. Few times a month may not offer enough practice that developers would routinely monitor with standards.

6.1.1 Product Assurance Disciplines

The results of the Question 6, 7 and 8 are listed below in Table 6.1. In this list the results of each column, knowledge, importance and implementation, are ranked in order where the discipline which got the best result on average is the topmost. For example the quality assurance is the best known discipline and for that reason in level Rank 1. In some cases,

more than one discipline got the same support from the respondent so for that reason there can be more than one discipline in the same ranking level.

Rank	**Knowledge**	**Importance**	**Implementation**
1	Quality Assurance	EEE Components	PA Management
2	Materials, Mechanical Parts and Processes; EEE Components; PA Management	Materials, Mechanical Parts and Processes; Quality Assurance, PA Management; SW Product Assurance	EEE Components; Quality Assurance
3	Dependability	Safety	Materials, Mechanical Parts and Processes, and Safety
4	SW Product Assurance	Dependability	SW Product Assurance
5	Safety		Dependability

Table 6.1: The results of the Questions 6, 7 and 8.

The knowledge of the product assurance disciplines is examined in Question 6. The best known discipline is quality assurance and disciplines of electrical, electronic and electromechanical components; materials, mechanical parts and processes; and product assurance management are also known quite well. The Question 7 focuses on the importance of each discipline and the electrical, electronic and electromechanical components discipline has been considered the most important. The result of the Question 8 shows that almost all of these disciplines are implemented always or often throughout the project. The three best ranked disciplines are product assurance management; electrical, electronic and electromechanical components; and quality assurance.

The conclusions of these three discipline questions is that the knowledge of quality assurance; materials, mechanical parts and processes; and electrical, electronic and electromechanical components is good and these disciplines are also perceived as important. The knowledge and importance of safety and dependability are quite low which is odd because these disciplines emphasize risk management which is one cornerstone of product assurance. One reason why the order of the disciplines is this might be that disciplines such as materials, mechanical parts and processes; and electrical, electronic and electromechanical components include more functional tasks and for example dependability discipline includes iterative background processes that are running all the time. These processes do not necessarily receive as much attention and in developers perspective this might lower the importance of these disciplines.

Even if the majority of these disciplines are mostly implemented always or often in organizations, the amount of the product assurance tasks is estimated to be sufficient and no one had thought that it would be too high. Question 10 examines the improvement need of disciplines. According to the survey, product assurance management and quality assurance are two disciplines that need improvements the most. This is interesting because the result of Question 6 and 8 shows that product assurance management and quality assurance are well known disciplines and implemented throughout the project almost always. The respondents are aware of the content of these disciplines and implement them but still for some reason they think that there is a need to improve the performance of discipline activities.

In Question 11, the respondents had an opportunity to give some improvement examples concerning the previous question. The respondents think that product and quality assurance training would improve the performance of discipline activities.

6.1.2 Schedule, Cost and Resource Needs

Questions 12, 13, 14 and 15 focus on schedule, cost and resource need evaluation. Figure 6.1 shows that the majority of respondents think that schedule, cost and resource need evaluation is easy and there is no need for a tool that would ease the evaluation process.

The minority that answered that the evaluation of the schedule, cost and resource need is not easy in Question 12, were able to list major challenges of evaluation process in Question 13. The major challenges are missing and changing product assurance requirements and time-consuming character of product assurance. Latter mentioned challenge is a problem if it is not fully taken into account in planning phase where the proposal schedule, cost and resource need evaluation is done.

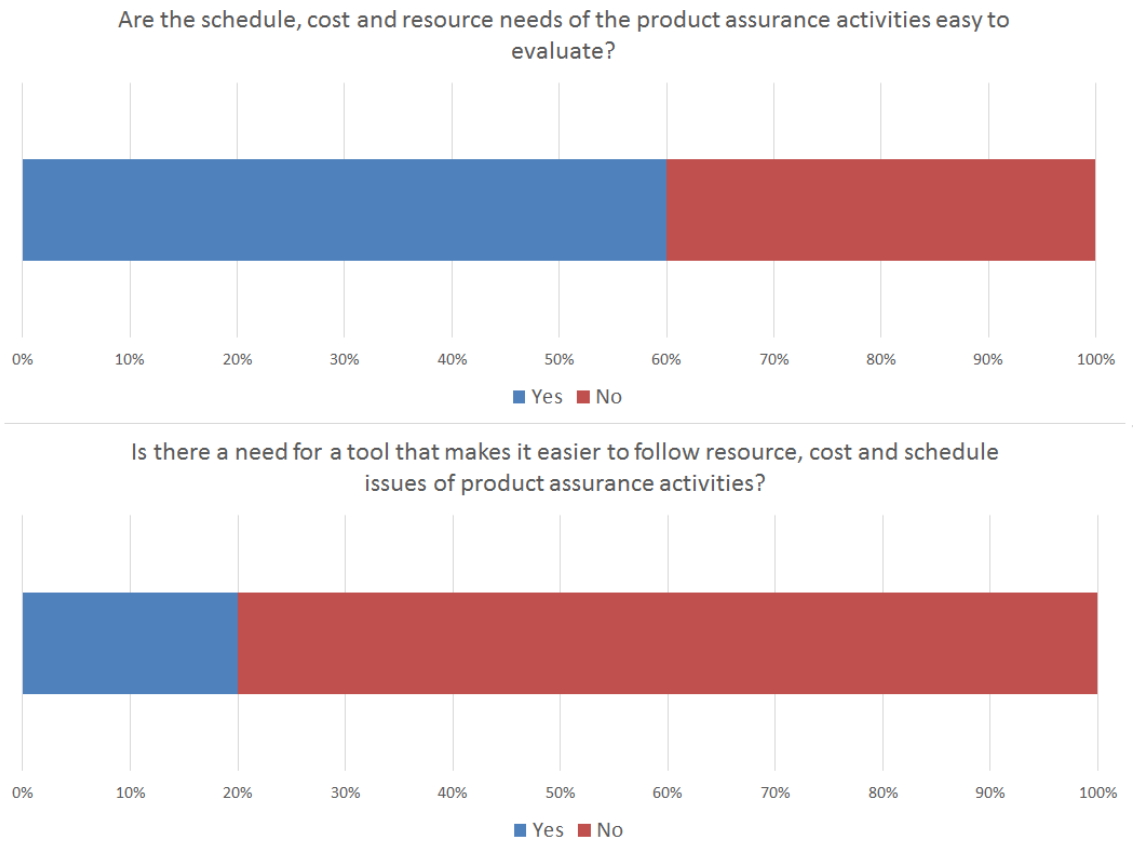


Figure 6.1: The results of the Question 12 and 14.

Those respondents who answered in Question 14 that there is a need for a tool that would ease the evaluation process of the schedule, cost and resource need were given to list some properties of a possible tool in Question 15. However this Question 15 did not receive any answers.

6.1.3 Analysis Methods and Risk Management

The most of the organizations who responded to the survey use a failure analysis method as shown in Figure 6.2. The most popular methods are FMEA and FMECA which are also presented in this thesis and less used methods are WCA and PSA but these methods are not presented in this thesis. Figure 6.2 also shows that three out of five organizations think that risk management is implemented as a part of process.

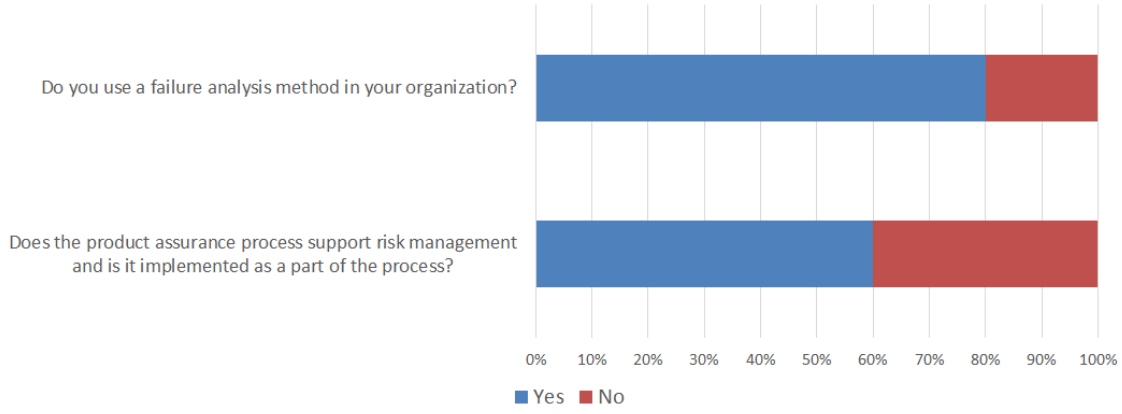


Figure 6.2: The results of the Questions 16 and 17.

Even if the hardware-software integration is supported with HSIA method, the result of Question 18 in Figure 6.3 shows that this integration process is not supported enough. The majority of the respondents think that it is supported moderately or slightly.

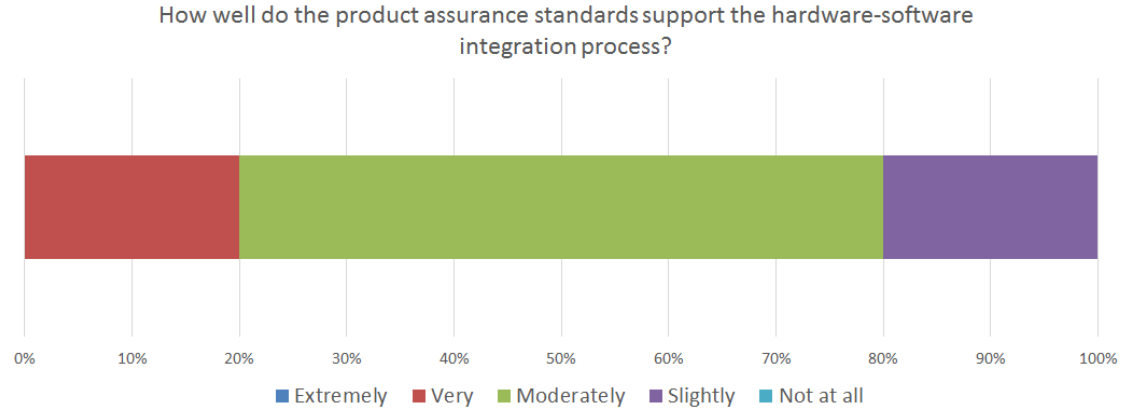


Figure 6.3: The results of the Question 18.

6.1.4 Product Assurance Requirements

The results of the Questions 19 and 20 indicate that product assurance requirements change sometimes or rarely. The majority of the respondents also feel that there is no need to update product assurance requirements more often during the project.

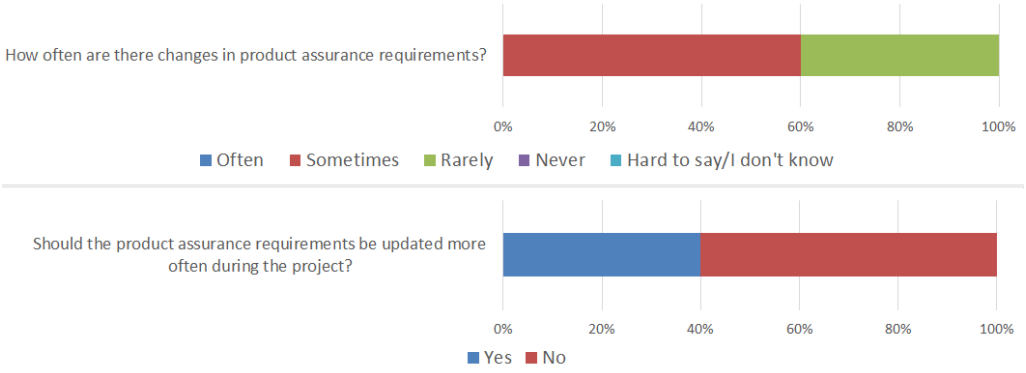


Figure 6.4: The results of the Questions 19 and 20.

One reason why the respondents feel that there is no need to update product assurance requirements more often might be the lifecycle structure of space projects. Space standards guide to update requirements after milestones and the time between two milestone might be months so the respondents are used to long intervals of requirements updates.

6.1.5 Product Assurance Management

Figure 6.5 shows that the opinion about the importance of PA manager is divided. 40% of respondents think that this role is very important and 40% think that it is slightly important. The reason why the opinion respondents differ might be a caused by different emphasis of the role inside the organization. The role might include more responsibilities in different organizations so the visibility of the role might vary. If this responsibility is not visible then the developers might embrace it as less important.

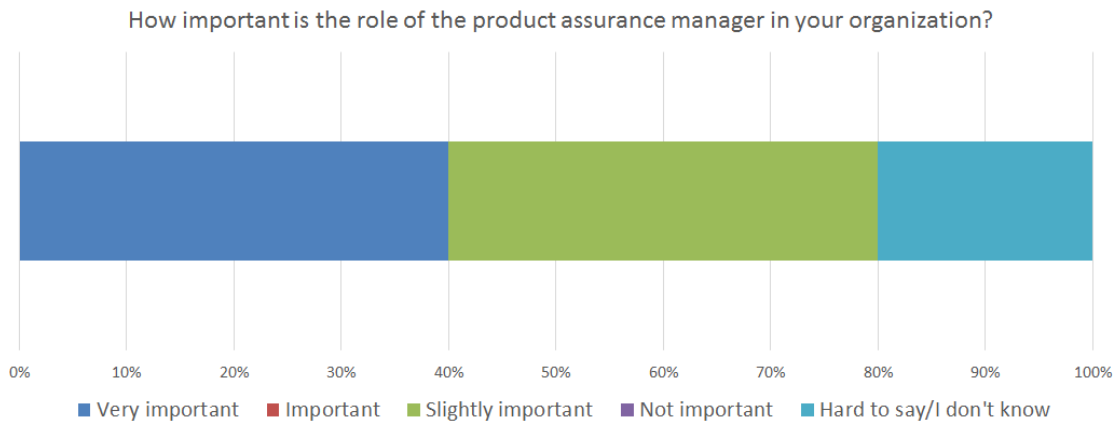


Figure 6.5: The results of the Question 21.

Even if the majority of the respondents feel that this role is slightly important they have found a lot of task and responsibilities that belong to PA manager. Next list includes a few of these tasks and responsibilities.

- Management and practical implementation of requirements.
- Ensuring and controlling quality.
- Supporting project management from product assurance point of view.
- Making sure that the final product will be compliant with requirements

6.2 The Results of The Second Section

The majority of the respondents are familiar or somewhat familiar with agile methods and only one respondent is not familiar with these methods. The result of Question 24 are listed below. In this list, the best known method is the topmost and the least known is the lowest.

- Scrum
- Lean
- XP

- Kanban

The result of Question 25 in Figure 6.6 shows that three out of five organization use agile methods to some extent and the rest two organization use a little or not at all.

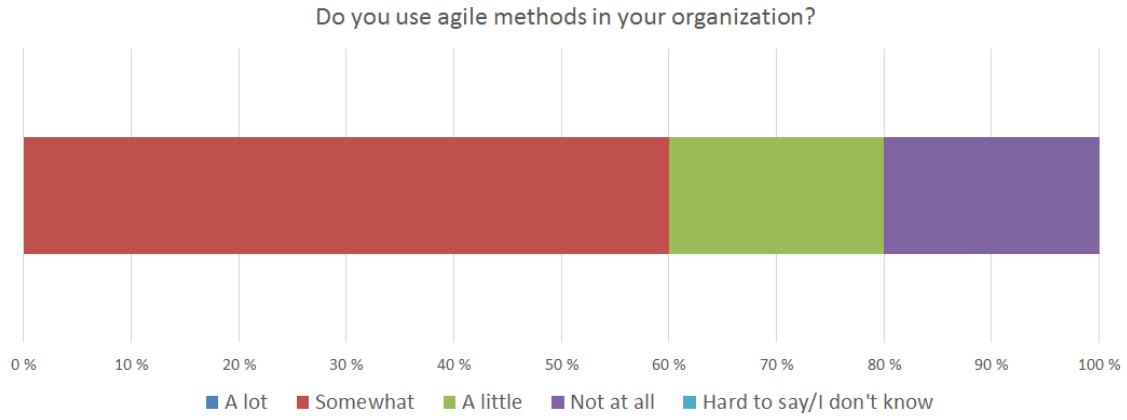


Figure 6.6: The results of the Question 25.

6.2.1 Project Lifecycle

The division of lifecycle is seen as well as possible but also hard. The result of Question 26 is presented in Figure 6.7 and it can be seen that 40% of respondents have seen this division as possible and the rest 60% of the respondents feel that it is hard. The reason why the division has not been seen as easy can be explained with the result of Question 25. Figure 6.6 shows that organization that responded to this survey use agile methods somewhat or less in their organization. For that reason the division of this kind might seem difficult to implement because it is not used in practice. Also standards prefer waterfall method quite strongly and because of that the respondents might experience iterative space system development as a challenge.

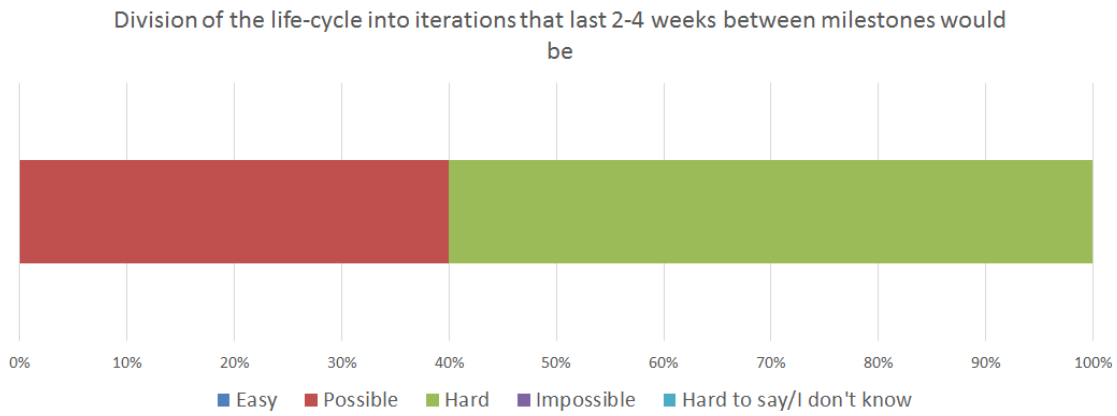


Figure 6.7: The results of the Question 26.

Questions 19 and 20 in the first section examine the amount of requirement changes and the updating need of the requirements. The results show that changes occur sometimes and there is not a greater need to update requirements more often. However, in Question 27 examined that if the requirements are updated after every iteration it would increase the safety and reliability of the product and Figure 6.8 shows that majority of the respondents think that safety and reliability would increase at least somewhat.

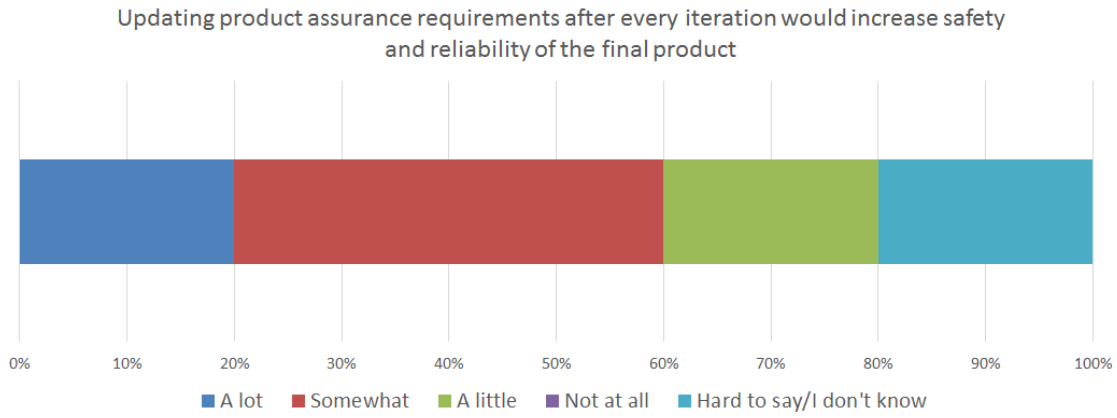


Figure 6.8: The results of the Question 27.

6.2.2 Risk Management and Verification

The result of Question 28 is broadly in line with expectations and is presented in Figure 6.9. Because the risk management is an iterative activity itself it is natural to expect

that also respondents think that iterative development support risk management. All the organizations that responded think that iterative development model can support risk management to some degree. Still it seems that the iterative risk management does not receive the full confidence of respondents. The respondents of one organization feel that iterative development would support risk management a lot and in other organizations respondents think that it would support somewhat or less.

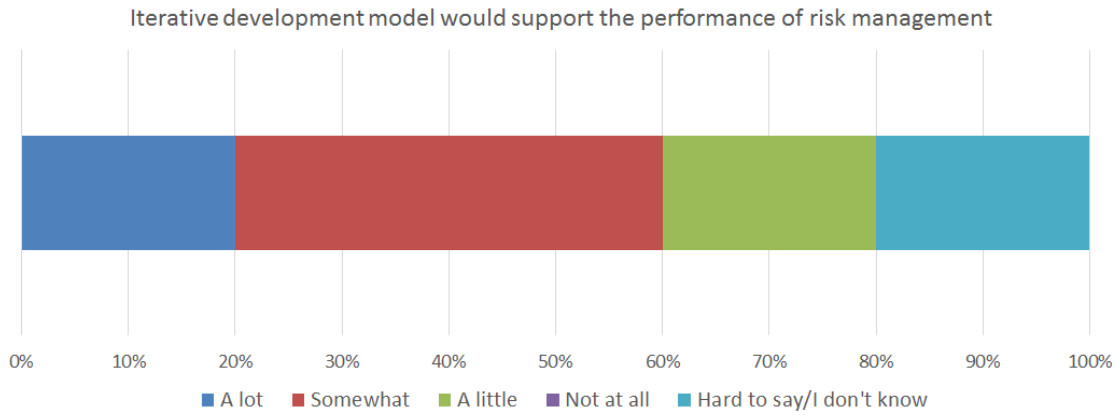


Figure 6.9: The results of the Question 28.

Even if there is no consensus between the answers, it can be concluded that the risk management and its activities do not comply with iterative and agile development because respondents does not think that it is unable to support risk management with iterative development. Iterative risk management can thus be presented in connection with agile as follows.

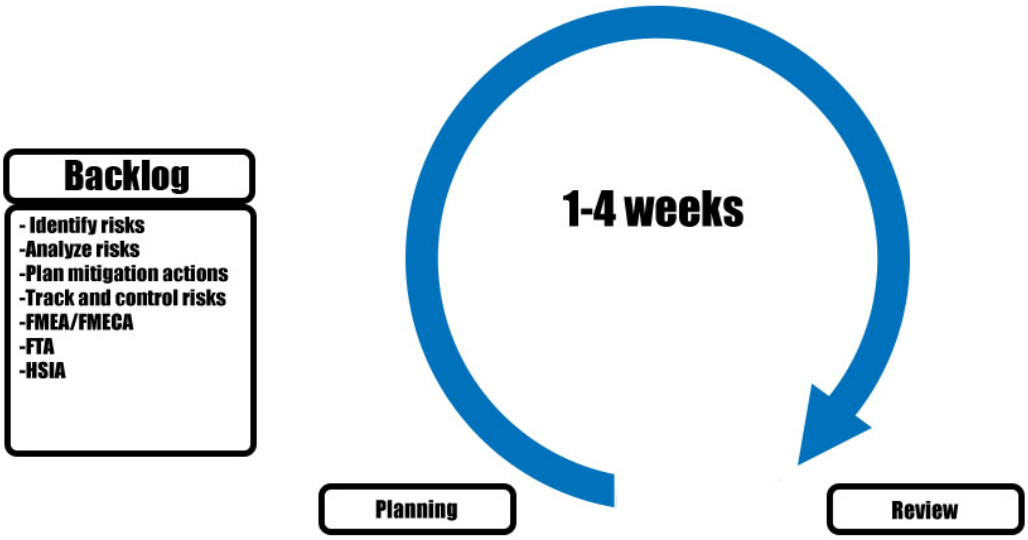


Figure 6.10: Risk management in iterative development.

In Figure 6.10 all risk management task are included to the backlog and in planning meeting these task are prioritized. Risk management tasks such as identification, analyzing and mitigation actions are performed. In addition to these tasks also analysis tasks, such as FMEA, FTA and HSIA that are used to support risk management and software-hardware integration can be included to the backlog.

Because the risk management can be supported with iterative development it would be necessary to think that agile can offer also other opportunities to risk management. In context of Scrum, burndown chart was introduced as a chart that made it possible to monitor progress of project graphically. In risk management context, this chart can be changed for risk burndown chart where the risk exposure is presented as a function of sprints as shown in Figure 6.11.

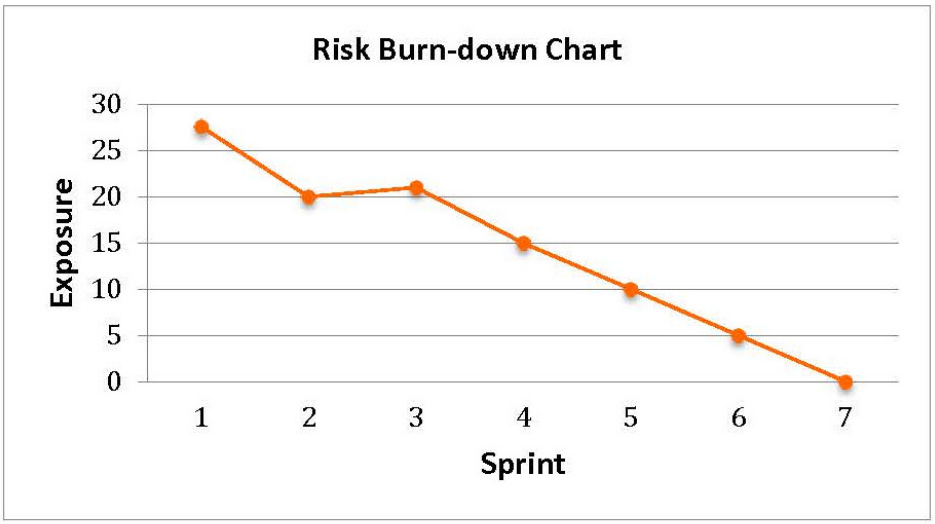


Figure 6.11: Risk burndown chart.[44]

Even though the verification emphasises waterfall method, the majority of respondents think that it can be supported with iterative development. However if the results of Questions 28 and 29 (Figures 6.9 and 6.12) are compared it can be noted that the respondents think that risk management can be supported with iterative development more efficiently than verification.

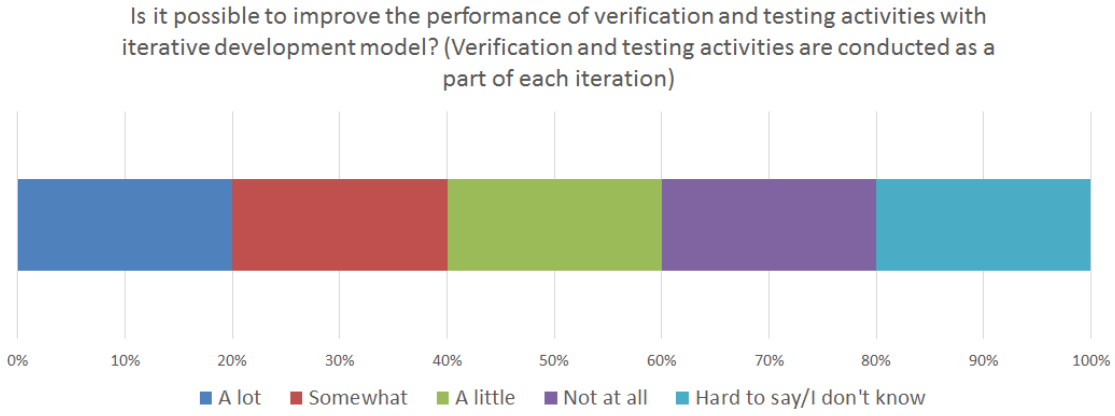


Figure 6.12: The result of the Question 29.

Few challenges that affect to the effectiveness of the verification are introduced in Section 5.2.2. Because the respondents think that it is possible to support verification with iterative development it might be relevant to figure out how agile development would solve these challenges. Missing customer requirements might cause misunderstandings between the customer and supplier. In space system development the requirements are in key element and without clear requirements it might be possible that the product is unreliable.

Agile and especially Scrum offers different types of ceremonies and review meeting is a candidate to solve the challenge of missing customer requirements. After every iteration customer and supplier updates the requirements and thus ensures that requirements are clear. For this reason misunderstandings do not occur and developers can be confident with that the system is reliable.

Another challenge introduced in Section 5.2.2 concerns verification experts. The problem is that these experts are invited into the process only in late phase of the project. Working of these experts would be efficient if they took part in the verification process already in planning phase. The amount of verification tasks might be quite low in early phases so the working with verification every day and full-time might be a challenge. However it might be necessary that verification experts take part for the development process by attending meetings. Daily stand-up meeting and review meetings are good opportunities to verification experts to express verification related problems and challenges.

6.2.3 Process Management

Earlier in Questions 12 ad 14 respondents answered that schedule, cost and resource needs are easy to evaluate and there is no need for a tool that would make this evaluation easier. However in Question 30, Kanban board and burn down chart are presented as tools of this kind. Now the Figure 6.13 shows that the majority of the respondents which have an opinion about this issue think that these tools can make it somewhat easier to evaluate schedule, cost and resource needs. It is interesting that respondents do not feel need for tools but when they are offered one they think that it might be potential and useful. Maybe the need of process management tools is not questioned and things are done in

the same way as before because new ways of doing things in a more efficient way are not recognized.

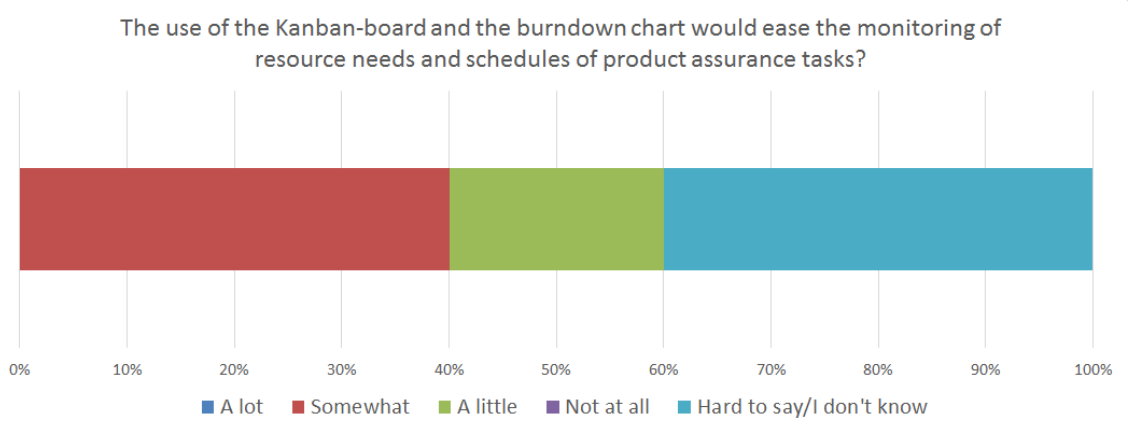


Figure 6.13: The results of the Question 30.

Kanban board also supports the understanding of the overall view. The result of Question 32 shows that better understanding of overall view of the project is seen as an opportunity to increase the quality, safety and reliability properties of the product. Figure 6.14 shows that the majority of the respondents think that better understanding would increase at least somewhat the above-mentioned properties of the product.

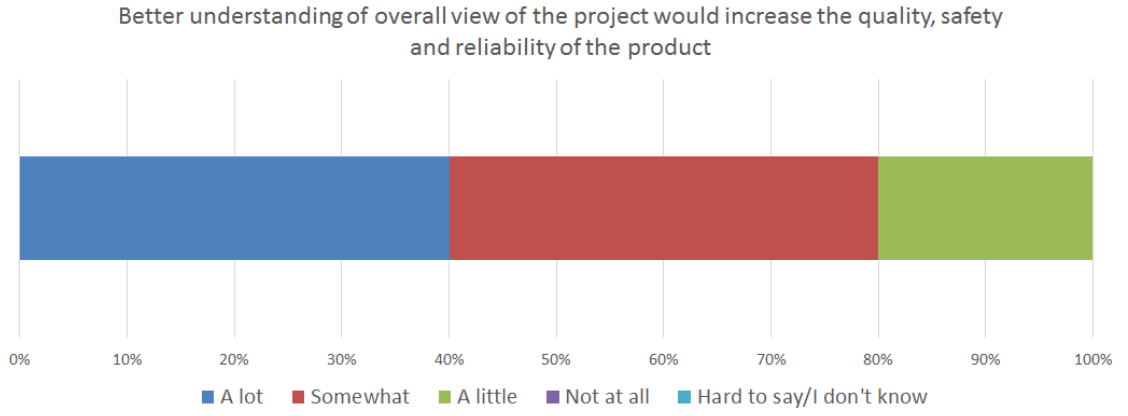


Figure 6.14: The results of the Question 32.

The results of Question 33 were unanimous. All respondents think that if the amount of simultaneous working task is lower it is easier to focus on product assurance tasks. Few justifications include good aspects why the elimination of simultaneous working tasks is

important in product assurance domain. One aspect was that if there is less simultaneous working tasks then there is more time to focus on doing things right in the first place. Another point of view was that smaller batches are easier to verify. Workflow visualization is one of the Kanban principles and according to result of Question 33, in space system development this principle should be followed during the project lifecycle because it make it easier to focus on product assurance tasks and this way increase the safety and reliability of the product.

Daily stand-up meeting has been seen useful and the respondents think that it might support the product assurance. As mush as 40% of respondents think that daily stand-up meeting can support product assurance process a lot as seen in Figure 6.15.

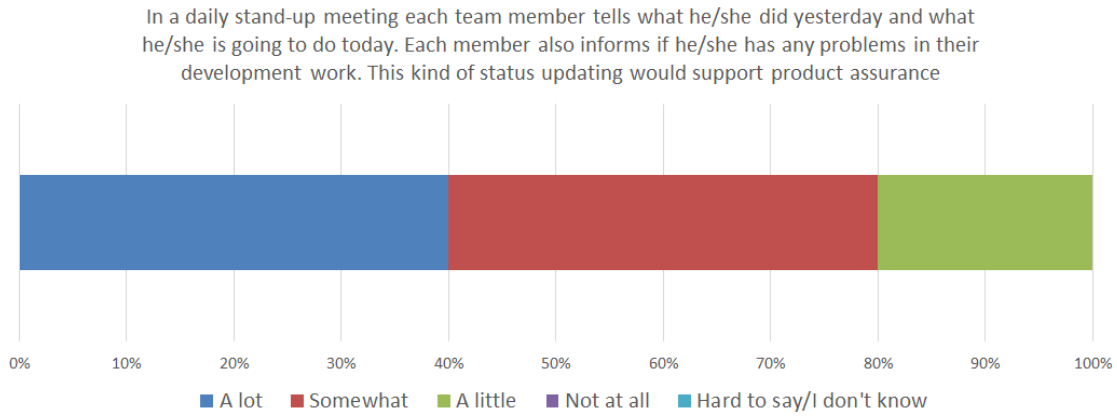


Figure 6.15: The results of the Question 37.

In Question 38, the review meeting is presented as an opportunity that prepares a project to the milestones and increases the safety and reliability properties of the product. Figure 6.16 shows that the majority of the respondents think that review meeting can support safety and reliability properties of the product and prepares the project to the milestones.

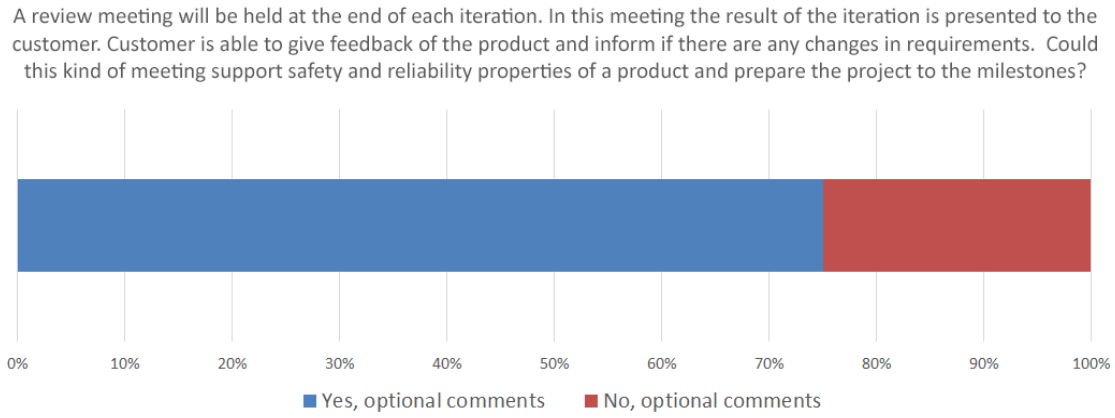


Figure 6.16: The results of the Question 38.

6.2.4 Documentation and Customer Collaboration

In documentation context, product backlog is considered to be useful tool that would ease the documentation process of completed tasks and requirements. All respondents think that it would ease that process a lot or somewhat as shown in Figure 6.17.

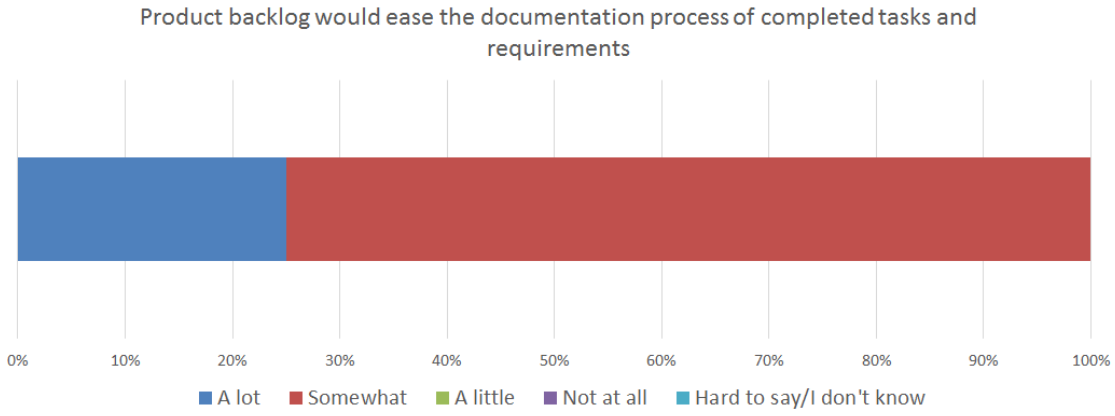


Figure 6.17: The results of the Question 31.

Because the product backlog seems so useful it can be assumed that also other tools that can be connected to the documentation would be potential. Tools can also be used to avoid unnecessary documentation not only to support documentation process. For example sometimes unnecessary documentation is prepared inside the team because it is a way to communicate with other team members and inform PA manager. In Section 6.2.3

mentions that Kanban board would make it easier to follow the process. This tool would also support the working of PA manager because one glance with the board allows the PA manager and other team members to know what tasks are already done and what other team members are doing. This decreases the workload of PA manager and the status of the project is updated consistently. Still it is important to remember that this board only works if everybody updates it regularly.

Customer collaboration is the topic of Questions 35 and 36. The majority of the respondents feel that customer collaboration would affect positively to the product assurance as noted from Figure 6.18. Even if customer collaboration is felt as positive matter in product assurance context, the optional comments of Question 36 includes aspects that make efficient customer collaboration challenging. Especially working in the same space is perceived to be challenging and it is hard for the respondents to imagine that how developers, acceptance testers and customer could get into the same space. In space development it is obvious that for example it is hard for the customer to work with other development team in the same space because the customer can be located in a completely different country. Even if it is hard for development team and customer to work in the same space, they need to update the status in other ways.

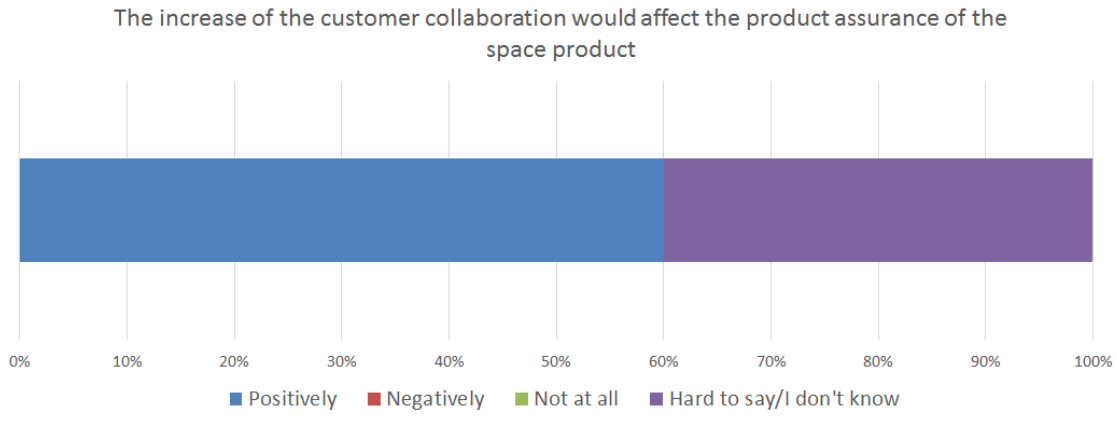


Figure 6.18: The results of the Question 35.

In Chapter 5 it is said that unnecessary documentation such as email should be avoided but sometimes, in space system development, this kind of communication is necessary

because the customer is somewhere else. However, this kind of communication should be kept as minimal as possible which might be a problem. If the topic of the email is complex, the amount of emails might increase too high. In this kind of situations a phone call might be better option and also the misunderstandings that might cause safety and reliability loss of the product would be more likely to be clarified immediately.

6.2.5 Software Development

Agile development and especially XP offers practises that support the quality of the software product. In Figure 6.19, the result of Question 34 shows that the respondents do not know how to deal with these practises.

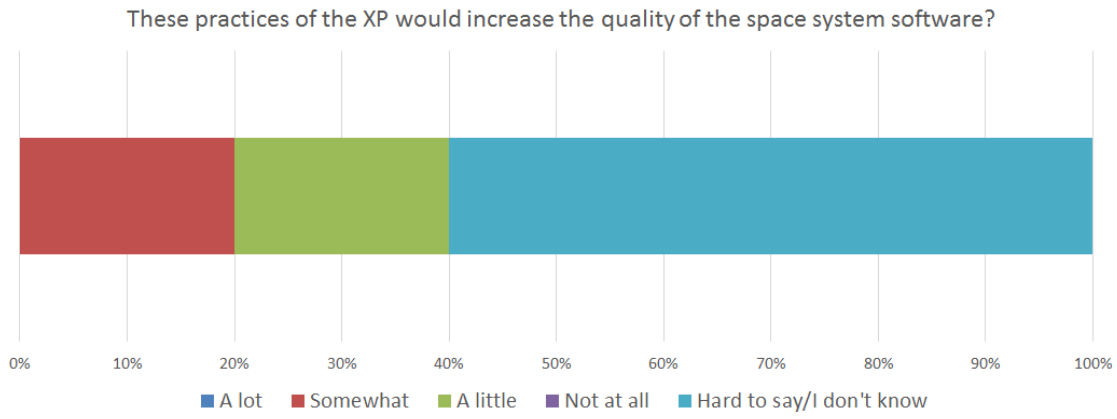


Figure 6.19: The results of the Question 34.

One reason might be that this agile method is not familiar enough to the respondents for them to evaluate the impact of these practises to the software.

6.2.6 Product Assurance Activities and Agile

The majority of the respondents think that there are product assurance activities that can be supported with agile development and there is no activities that can suffer from agile development. The respondents felt that one possible issue that can suffer from agile development is the cost of acceptance tests because the test is performed in every iteration and might cause the test costs to explode. Additional feedback part also tells the respondents feel that hardware testing and agile is a challenging combination.

6.3 The Conclusions of The Survey

6.3.1 Product Assurance

The main purpose of this survey is to answer two Research Questions, RQ1 and RQ2, which are presented in Chapter 1. The first Research Question searches answer to the question how well space product assurance is performed in Finnish space organizations generally and the second Research Question searches answer to the question how space product assurance can be supported with agile.

In space organizations, functional disciplines such as electrical, electronic and electromechanical components; materials, mechanical parts and processes; product assurance management; and quality assurance are well known and quite well implemented disciplines. The reason why these methods are well known and implemented quite well can be explained with the fact that these disciplines include functional tasks that are dependent of project phases. Even if the respondents know these methods quite well they are a little bit uncertain and they might need some additional practice to compliance with product assurance disciplines.

The amount of product assurance tasks is sufficient but a higher amount could offer additional practice. If the respondents would work with product assurance task more often, the product assurance disciplines would become more familiar to the developers and they would work more routinely with them.

Schedule, cost and resource need evaluation are quite easy and there is no need for a tool that would make this evaluation easier. However it is hard to evaluate the schedule, cost and resource need if there are missing requirements in design phase because if these requirements appear later they might affect to the budget and schedule negatively.

Risk management is implemented as a part of the project and most of the organizations use failure analysis method. The most popular analysis methods are FMEA and FMECA. Even if the failure analysis methods are used, the respondents think that hardware-software

integration process is not supported enough by standards.

Product assurance requirements change sometimes but the majority of the respondents still think that there is no need to update these requirements more often. The lifecycle structure of space system development might be one reason because respondents are used to long intervals of requirements updates and for that reason they feel that there is no need to update requirements more often.

Product assurance management is one of the seven disciplines and it defines the role of PA manager. The importance of this role varies in different organizations. The reason for this might be the different emphasis of the role inside the organizations. Efficient and visible use of the role makes it more important because the developers see the benefits of the role better.

As an answer to RQ1 product assurance and its activities are performed in Finnish space organizations quite well and respondents do not observe major needs for improvement. However, the second section of the survey shows that several product assurance parts can be performed in a more efficient way if they are supported with agile methods.

6.3.2 Product Assurance and Agile Methods

Respondents think that there is no need to update requirements more often during the project but it would be beneficial to update requirements after every iteration. The frequent updating of requirements supports the safety and reliability of the final product because then it can be assured that manufactured system is built according to correct requirements.

Process management can be supported with agile. Understanding the overall view is important if reliable and safety products are desired. Because the hardware-software integration is supported poorly by the standards, it is the responsibility of the development team to ensure that hardware and software product are reliable and they work together safely. For this reason it would be essential for software developers to understand the

hardware development process better and vice versa. This kind of overall view and status updating can be supported with tools such as Kanban board and meetings such as daily stand-ups.

Risk management and verification are essential parts of space product assurance. Even if the risk management is an iterative activity itself the performance of it can be supported with agile development. If all risk management tasks are recorded into the product backlog it would ease the monitoring of risk management tasks between the iterations. Risk management and verification can both benefit from iterative structure of the project and different meetings during the iteration. If risk management and verification activities are taken into account during each iteration then the states of these two are public and these activities are connected to the project already at the beginning of the project.

Also PA manager would benefit from different meetings and especially daily stand-up meetings. One responsibility of the PA manager is to take care that product assurance tasks are completed on schedule so this kind of status updating meeting would help PA manager to perform all responsibilities more efficiently. If the PA manager takes part in meetings the communication with development team and customer is more efficient which also can result in product assurance activities being performed more efficiently and the need for requirement changes are identified earlier.

As an answer to RQ2 product assurance can be supported with agile practises and even individual agile practises can have a positive effect to product assurance.

Chapter 7

Summary

Product assurance is an essential part of reliable system development. Product assurance process can be supported with risk management and with different failure analysis methods. Product assurance is emphasised in development process of mission critical systems and in this thesis mission critical systems are space systems.

All space system are complex systems that consist of different parts and components. Because space systems are long-lasting and difficult to repair, is it necessary that all components and parts work properly right from the beginning. The development process of space systems is highly standardized by ECSS-standards. These standards define four different branches: Space project management, space product assurance, space engineering and space sustainability. In this thesis, the main focus is in space product assurance branch.

Space product assurance branch defines seven disciplines: Product assurance management; quality assurance; dependability; safety; electrical, electronic, and electromechanical components; material, mechanical parts and processes; and software product assurance. In addition to these disciplines, also verification and risk management are presented in this thesis because they act in a key role in product assurance domain.

Agile development is a development model that emphasizes transparency of the process and fast response to changes. Agile development defines different methods and agile

methods presented in this thesis are Scrum, Lean and Kanban, and Extreme programming. Scrum is a well known method that is based on regular ceremonies such as daily stand-up, review and retrospective meetings. Scrum also defines different roles such as Scrum master and product owner. Lean and Kanban emphasize waste elimination and in these methods everything that does not produce value to the customer is waste. The waste elimination is enhanced for example with workflow visualization. Extreme programming defines different practises that are programmer relevant and advance the quality of the software products.

Agile development differs a lot from traditional waterfall method. In waterfall method, all requirements and specifications are defined already at the beginning of the project. In agile, the development process consists of iterations and for that reason the development process is more flexible and transparent than in waterfall model.

Space systems development process differs from agile development a lot. Space systems development process reminds the waterfall model and standards also emphasizes waterfall. Because the space systems development process is complex and it is necessary to ensure that the final product is safe, agile development would offer practises that support space product assurance.

In this thesis the main goal is to search answers to two research question: RQ1 how well product assurance is performed in Finnish space organization generally? and RQ2 how space product assurance can be supported with agile development?

The answers of these two research questions are examined with the help of a survey which consists of about forty questions that were sent to the experts of space project development.

The survey is divided into two parts that support RQ1 and RQ2. The first part of the survey examines product assurance and especially the performance of product assurance activities. The second part proposes different agile practises that would support the prod-

uct assurance process.

The survey got eight answers from five different organizations. In these organizations, product assurance disciplines such as quality assurance; material, mechanical parts and processes; product assurance management; and electrical, electronic and electromechanical components are known and implemented quite well. Respondents think that there is no need to improve the performance of product assurance discipline and activities. For example there is no need to update product assurance requirements more often and schedule, cost and resource need evaluation seem easy.

However the results of the second section of the survey show that product assurance activities can be supported with agile development. If product assurance requirements are updated after every iteration it would increase the safety and reliability properties of the product. Tools such as Kanban board and burndown chart seem useful and would support the evaluation process of schedule, cost and resource needs.

Also other agile tools and practises would support product assurance. Daily stand-up meeting and Kanban board would ease status updating and review meeting would prepare the project for the milestones. Product backlog would support the documentation process because if all tasks are recorded to the backlog it is easy to check which tasks are done and when.

Broadly respondents think that product assurance activities could be supported with iterative development and there are no activities that would suffer from iterative development except hardware testing.

The results of the survey provide answers for RQ1 and RQ2. Product assurance is performed in Finnish space organizations quite well and there is no need to improve the performance of product assurance. However survey suggests that product assurance could be supported with agile and even individual agile practises can have positive effects to product assurance process.

Bibliography

[1] Al Safety Design Oy. *SPACE-2000, Development of PA/QA for space instruments*. Report version 1.1, 1996.

[2] Dev G Raheja and Michael Alloco. *Assurance technologies principles and practices: A product, process and system safety perspective*. John Wiley & sons, second edition, 2006.

[3] National Aeronautics and Space Administration. *NASA technical standard NASA-GB-8719.13, Software Safety Guidebook*, 31 March 2004.

[4] European Cooperation on Space Standardization. *ECSS-Q-HB-80-03A, Space Product Assurance Handbook: Software dependability and safety*, 26 January 2012. <http://www.ecss.nl/>

[5] Jussi Pekka Kasurinen. *Ohjelmistotestauksen käsikirja*. Docento Oy, 2013.

[6] Tuula Kyllönen. *Ohjelmistojen testauksen kehittäminen ja parantaminen*. Joensuun yliopisto, 2008.

[7] David Alberico, Johan Bozarth, Michael Brown, Janet Gill, Steven Mattens and Arch McKinlay VI. *Software System Safety Handbook -A Technical and Managerial Team Approach*, 1999. <http://www.system-safety.org/Documents/Software_System_Safety_Handbook.pdf>.

[8] C. Warren Axelrod. *Applying Lessons from Safety-Critical Systems to Security-Critical Software*. Systems, Applications and Technology Conference (LISAT), pages 1-6, 2011 IEEE Long Island.

[9] Joseph Berk. *System failure analysis.* A S M International Material Park, USA 2009.

[10] William Marshall. *Avoiding Failure, Tolerant Faults*. <http://sg.rs-online.com/web/generalDisplay.html?id=promotions/discover_article/discover_article1#.VUmtUPntlBd>. Accessed April 15, 2015.

[11] Jeevan Perera and Jerry Holsomback. *An Integrated Risk Management Tool and Process*. Aerospace Conference, pages 129-136, 2005 IEEE. 2004

[12] Pat L. Clemens. *Falut tree analysis*. Fourth edition, 1993. <http://rischioatmosfereesplosive.studiomarigo.it/profiles/marigo2/images/file/1736612536.pdf>.

[13] Pentti Haapanen and Atte Helminen. *Failure Mode and Effects Analysis of Software-based Automation Systems*, August 2002. <http://www.julkari.fi/bitstream/handle/10024/124480/stuk-yto-tr190.pdf?sequence=1>.

[14] Ebru Nihal Cetin. *FMECA Applications and Lessons Learnt*. Reliability and Maintainability Symposium (RAMS), pages 26-29, 2015.

[15] European Cooperation on Space Standardization. *ECSS-S-ST-00C, ECSS system-Description, implementation and general requirements*, 31 July 2008. <http://www.ecss.nl/>

[16] Malcolm Macdonald, Viorel Badescu. *International Handbook of Space Technology*. Springer, 2014

[17] European Cooperation on Space Standardization. *ECSS-M-ST-10C Rev. 1, Space project management -project planning and implementation*, 6 March 2009. <http://www.ecss.nl/>

[18] European Cooperation on Space Standardization. *ECSS-E-ST-10C, System engineering -System engineering general requirements*, 6 March 2009. <http://www.ecss.nl/>

[19] Ehsan Ahmad, Bilal Raza, Robert Feldt and Tanja Nordebäck. *ECSS Standard Compliant Agile Software Development*, 2010. <http://robertfeldt.net/publications/Ahmad_NSEC2010_Agile_ECSS_Camera_Ready.pdf>.

[20] European Cooperation on Space Standardization. *ECSS Standards*. <http://www.ecss.nl/>

[21] European Cooperation on Space Standardization. *ECSS Standard ECSS-Q-ST-10C, Space product Assurance -Space Product Assurance Management*, 15 November 2008. <http://www.ecss.nl/>

[22] European Cooperation on Space Standardization. *ECSS standard ECSS-Q-ST-20C, Space product assurance -Quality assurance*, 1 March 2013. <http://www.ecss.nl/>

[23] European Cooperation on Space Standardization. *ECSS standard ECSS-Q-ST-30-C, Space product assurance -Dependability*, 6 March 2009. <http://www.ecss.nl/>

[24] European Cooperation on Space Standardization. *ECSS standard ECSS-Q-ST-40C, Space product assurance -Safety*, 6 March 2009. <http://www.ecss.nl/>

[25] European Cooperation on Space Standardization. *ECSS standard ECSS-Q-ST-60C, Space product assurance -EEEE components*, 21 October 2013. <http://www.ecss.nl/>

[26] European Cooperation on Space Standardization. *ECSS standard ECSS-Q-ST-70C, Space product assurance -Materials; Mechanical parts and processes*, 15 October 2014. <http://www.ecss.nl/>

[27] European Cooperation on Space Standardization. *ECSS standard ECSS-Q-ST-80C, Space product assurance -Software product assurance*, 6 March 2009. <http://www.ecss.nl/>

[28] European Cooperation on Space Standardization. *ECSS standard ECSS-E-ST-10-02C, Space engineering -Verification*, 6 March 2009. <http://www.ecss.nl/>

[29] European Cooperation on Space Standardization. *ECSS standard ECSS-M-ST-80C, Space project management -Risk management*, 31 July 2008. <http://www.ecss.nl/>

[30] Craig Larman. *Agile and Iterative Development: A Manager's Guide*. Addison-Wesley, 2004.

[31] Barbee Davis. *Agile Practices for Waterfall Projects: Shifting Processes for Competitive Advantage*. J. Ross Publishing Inc., October 2012.

[32] Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, and Dave Thomas *Manifesto for Agile Software Development*, 2001. <http://agilemanifesto.org/>. Accessed May 1, 2015.

[33] Balaji Sundramurthy and Murugaiyan Sundara Rajan. *WATEERFALLVs V-MODEL Vs AGILE: A COMPARATIVE STUDY ON SDLC*. International Journal of Information Technology and Business Management, Vol. 2 No. 1, June 2012.

[34] Hirotaka Takeuchi and Ikujiro Nonaka. *The new new product development game.* Harvard Business Review, 1986.

[35] Ken Schwaber and Jeff Sutherland. *The Scrum guide*, 2013. <http://www.scrumguides.org/>. Accessed June 10, 2015.

[36] Stacia Viscardi. *Professional Scrum Master's Handbook*. Produkt Publishing, 2013.

[37] Mountain Goat. *Scrum images* <http://www.mountaingoatsoftware.com/agile/scrum/images>. Accessed June 10, 2015.

[38] Mary Poppendieck and Tom Poppendieck. *Implementing Lean Software Development*. Addison-Wesley, 2006

[39] Muhammad Ovasis Ahmad, Jouni Markkanen and Markku Ovio. *Kanban in software development: A systematic literature review*. Software Engineering and Advanced Applications (SEAA), 39th EUROMICRO Conference on, pages 9-16, 2013.

[40] Teijo Lehtonen, Seppo Tuomivaara, Ville Rantala, Marja Känsälä, Tero Jokela, Kaisa Könnölä, Matti Kaisti, Samuli Suomi, Minna Isomäki and Marko Ylitova. *Sulautettujen järjestelmien ketterä käsikirja*. Painosalama Oy, 2014.

[41] Kent Beck and Cynthia Anders. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, second edition, 2005.

[42] Matti Kaisti, Tapio Mujunen, Tuomas Mäkilä, Ville Rantala, and Teijo Lehtonen. *Agile Principles in the Embedded System Development*. In XP 2014 Rome - 15th International Conference on Agile Software Development, 2014.

[43] Robert Feldt, Richard Torkar, Ehsan Ahmad and Bilal Raza. *Challenges with Software Verification and Validation Activities in the Space Industry*. Software Testing, Verification and Validation (ICST), 2010 Third International Conference on, pages 225 - 234, 2010.

[44] Satheesh Thekku Veethil. *Risk management in agile*. <https://www.scrumalliance.org/community/articles/2013/2013-may/risk-management-in-agile>. Accessed August 16, 2015.

Appendix A

Survey Platform

1. Organization/company where you are working *

(This information is NOT connected to your other answers)

_____

_____

_____

2. What is your main responsibility in this organization? *

◯ Design

◯ Software development

◯ Electronics development

◯ Mechanical development

◯ Testing

◯ Management

◯ Other_____

3. How many years have you worked with space technology/space system development? *

◯ <5 years

◯ 5-10 years

◯ >10 years

Space Product Assurance

4. How often do you work with product assurance? *

◯ Daily

◯ Weekly

◯ A few times a month

◯ Rarely

◯ Never

5. Following the standards during the project is *

◯ Easy

◯ Quite easy

◯ Quite hard

◯ Hard

◯ Hard to say/I don't know

6. How well do you know the following product assurance disciplines? *

	Extremely	Very	Moderately	Slightly	Not at all
Product assurance management	◯	◯	◯	◯	◯
Quality assurance	◯	◯	◯	◯	◯
Dependability	◯	◯	◯	◯	◯
Safety	◯	◯	◯	◯	◯
EEE components	◯	◯	◯	◯	◯

Materials, Mechanical parts and processes	○	○	○	○	○
Software product assurance	○	○	○	○	○

7. How important are the following product assurance disciplines? *

	Very important	Important	Slightly important	Not important	Hard to say/I don't know
Product assurance management	○	○	○	○	○
Quality assurance	○	○	○	○	○
Dependability	○	○	○	○	○
Safety	○	○	○	○	○
EEE components	○	○	○	○	○
Materials, Mechanical parts and processes	○	○	○	○	○
Software product assurance	○	○	○	○	○

8. In your organization, following the product assurance disciplines are implemented throughout the project *

	Always	Often	Sometimes	Rarely	Never
Product assurance management	○	○	○	○	○
Quality assurance	○	○	○	○	○
Dependability	○	○	○	○	○
Safety	○	○	○	○	○
EEE components	○	○	○	○	○
Materials, Mechanical parts and processes	○	○	○	○	○
Software product assurance	○	○	○	○	○

9. The amount of product assurance tasks is *

- ◯ Too high
- ◯ Sufficient
- ◯ Moderate
- ◯ Too low
- ◯ Hard to say/I don't know

10. Performance of the following product assurance discipline activities should be improved in your organization *

Select max 2.

- ☐ Product assurance management
- ☐ Quality assurance
- ☐ Dependability
- ☐ Safety
- ☐ EEE components
- ☐ Materials, Mechanical parts and processes
- ☐ Software product assurance
- ☐ None of the above

11. Please give some improvement examples (optional)

_____

_____

_____

12. Are the schedule, cost and resource needs of the product assurance activities easy to evaluate? *

- ◯ Yes
- ◯ No

If you answered "No" to the question 12.

13. What are the major challenges in evaluating these needs?

_____

_____

_____

14. Is there a need for a tool that makes it easier to follow resource, cost and schedule issues of product assurance activities? *

◯ Yes

◯ No

If you answered "Yes" to the question 14.

15. What kind of properties should that tool have?

_____

_____

_____

16. Do you use a failure analysis method in your organization? *

◯ Yes, which_____

◯ No

17. Does the product assurance process support risk management and is it implemented as a part of the process? *

◯ Yes

◯ No, justify_____

18. How well do the product assurance standards support the hardware-software integration process? *

◯ Extremely

◯ Very

◯ Moderately

◯ Slightly

◯ Not at all

19. How often are there changes in product assurance requirements? *

◯ Often

◯ Sometimes

◯ Rarely

◯ Never

◯ Hard to say/I don't know

20. Should the product assurance requirements be updated more often during the project? *

◯ Yes, justify_____

◯ No, justify_____

21. How importance is the role of the product assurance manager in your organization? *

◯ Very important

◯ Important

◯ Slightly important

◯ Not important

◯ Hard to say/I don't know

22. What are the most important tasks and responsibilities of the product assurance manager? *

_____

_____

_____

Product Assurance and Agile Methods

Agile development emphasizes customer collaboration, communication and fast response to changes. Agile development methods divide the life-cycle of the project into short fixed length iterations which typically last 2-4 weeks. The different agile methods, such as Scrum, Extreme Programming, Kanban and Lean offer different activities, roles and tools that support agile principles and values.

23. How familiar are you with agile development? *

- ◯ Very familiar
- ◯ Familiar
- ◯ Somewhat familiar
- ◯ Not at all familiar
- ◯ Hard to say/I don't know

24. How well do you know the following agile methods? *

	Extremely	Very	Moderately	Slightly	Not at all
Lean	◯	◯	◯	◯	◯
Kanban	◯	◯	◯	◯	◯
Scrum	◯	◯	◯	◯	◯
Extreme programming	◯	◯	◯	◯	◯

25. Do you use agile methods in your organization? *

- ◯ A lot
- ◯ Somewhat
- ◯ A little
- ◯ Not at all
- ◯ Hard to say/I don't know

Space Product Development Process and Agile Development

European Cooperation for Space Standardization divides space system development process into seven phases which are separated with milestones. Agile development divides life-cycle into short fixed length iterations that last 2-4 weeks. Although space system development is a highly standardized industry, iterations of agile development could be implemented between milestones.

26. Division of the life-cycle into iterations that last 2-4 weeks between the milestones would be *

- ◯ Easy
- ◯ Possible
- ◯ Hard
- ◯ Impossible
- ◯ Hard to say/I don't know

27. Updating product assurance requirements after every iteration would increase safety and reliability of the final product *

- ◯ A lot
- ◯ Somewhat
- ◯ A little
- ◯ Not at all
- ◯ Hard to say/I don't know

28. Iterative development model would support the performance of risk management *

- ◯ A lot
- ◯ Somewhat
- ◯ A little
- ◯ Not at all
- ◯ Hard to say/I don't know

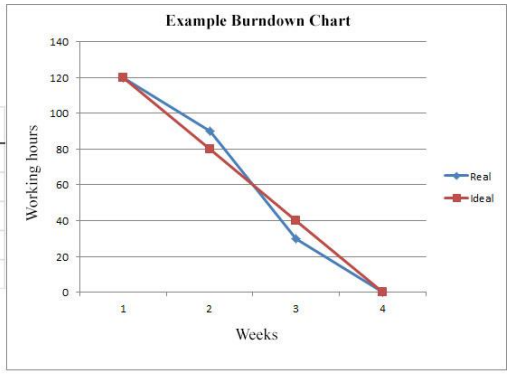
29. Is it possible to improve the performance of verification and testing activities with iterative development model? (Verification and testing activities are conducted as a part of each iteration) *

- ◯ A lot
- ◯ Somewhat
- ◯ A little
- ◯ Not at all
- ◯ Hard to say/I don't know

Agile development methods offer tools that make it easier to follow the progress of an iteration. One good example of this kind of tool is a Kanban-board where all tasks are included on a board that consists of three columns: "to do", "doing" and "done". When the board is up to date only one view tells the status of the iteration. Another good example of an agile tool is a burndown chart which is a graphical tool that expresses the progress as a function of time.

To do	Doing	Done
Task 5	Task 3	Task 1
Task 6	Task 4	Task 2
Task 7		

Kanban board



Burndown chart

30. The use of the Kanban-board and the burndown chart would ease the monitoring of resource needs and schedules of product assurance tasks? *

- ◯ A lot
- ◯ Somewhat
- ◯ A little
- ◯ Not at all
- ◯ Hard to say/I don't know

Product backlog is a prioritized and changing list of tasks and customer requirements that should be completed during the project.

31. Product backlog would ease the documentation process of completed tasks and requirements *

- ◯ A lot
- ◯ Somewhat
- ◯ A little
- ◯ Not at all
- ◯ Hard to say/I don't know

32. Better understanding of overall view of the project would increase the quality, safety and reliability of the product *

- ◯ A lot
- ◯ Somewhat
- ◯ A little
- ◯ Not at all
- ◯ Hard to say/I don't know

33. Would it be easier to focus on product assurance tasks if the amount of simultaneous working tasks was lower? *

- ◯ Yes, why_____
- ◯ No, why_____

Extreme programming (XP) is an agile method that offers practices. For example, the design is kept simple and all software is programmed in pairs. The testing includes unit testing, i.e. software is divided into small units which functionalities are confirmed with tests, and acceptance testing i.e. test that ensures the contractual correctness of the product . Also the whole team, including the customer, works in same space.

34. These practices of the XP would increase the quality of the space system software? *

- ◯ A lot
- ◯ Somewhat
- ◯ A little
- ◯ Not at all
- ◯ Hard to say/I don't know

35. The increase of the customer collaboration would affect the product assurance of the space product *

◯ Positively

◯ Negatively

◯ Not at all

◯ Hard to say/I don't know

36. Optional comments about customer collaboration

_____

_____

_____

Scrum is a well known agile method that includes different ceremonies such as planning, daily stand-up, review and retrospective meetings.

37. A daily stand-up meeting each team member tells what he/she did yesterday and what he/she is going to do today. Each member also informs if he/she has any problems in their development work. This kind of status updating would support product assurance process *

◯ A lot

◯ Somewhat

◯ A little

◯ Not at all

◯ Hard to say/I don't know

38. A review meeting will be held at the end of each iteration. In this meeting the result of the iteration is presented to the customer. Customer is able to give feedback of the product and inform if there are any changes in requirements. Could this kind of meeting support safety and reliability properties of a product and prepare the project to the milestones? *

◯ Yes, optional comments_____

◯ No, optional comments_____

39. Are there product assurance activities that cannot be supported with agile development? *

◯ Yes, what_____

◯ No

40. Are there product assurance activities that would suffer from use of agile development? *

◯ Yes, what _____

◯ No

41. Free comments and feedback

_____

_____

_____