Challenges with Software Verification and Validation Activities in the Space Industry

R. Feldt, R. Torkar, E. Ahmad and B. Raza
Blekinge Institute of Technology
SE-372 25 Ronneby, Sweden
Email: {rfd|rto}@bth.se

***Abstract*—Developing software for high-dependable space applications and systems is a formidable task. With new political and market pressures on the space industry to deliver more software at a lower cost, optimization of their methods and standards need to be investigated. The industry has to follow standards that strictly set quality goals and prescribes engineering processes and methods to fulfill them. The overall goal of this study is to evaluate if current use of the standards from the European Cooperation for Space Standardization (ECSS) is cost efficient and if there are ways to make the process leaner while still maintaining quality and to analyze if their verification and validation (V&V) activities can be optimized.**

This paper presents results from two industrial case studies of companies in the European space industry that are following ECSS standards in various V&V activities. The case studies reported here focus on how ECSS standards are used by the companies, how that affects their processes and, in the end, how their V&V activities can be further optimized.

***Keywords*-Case study, European Cooperation for Space Standardization, verification and validation**

I. Introduction

Software development projects for space applications and systems tend to have different dynamics than software projects in other domains. Development of software for space applications poses additional challenges due to its inherent requirement that the end product must be highly dependable. It is a formidable task to develop such systems because of the focus on reliability and dependability.

Industry has a long tradition of developing standards that strictly set quality goals and prescribes engineering processes and methods to fulfill them. The European Cooperation for Space Standardization (ECSS) has developed a single set of standards for the European space projects [1, 2]. These standards are derived from PSS-05 [3], an earlier space standard which were more prescriptive, demanded heavy documentation and favored waterfall and incremental development models. Since PSS-05 was a primary input for ECSS, activities at space industry have the legacy of PSS-05.

The quality of software is very much dependent upon the Verification and Validation Activities (VVAs) [4] taking place when developing the software. Both state-of-the-art and state-of-the-practice has proved that using combinations of different VVAs is more effective than compared to using a single VVA [5, 6]. According to [7], verification and validation of critical systems like e.g. satellites in a cost effective manner, is a challenging task and optimal VVAs are necessary to ensure quality by maximizing success in a limited budget. There is a need to optimize VVAs by understanding the overlap and variability between them, without losing quality. Defect detection completeness and successful integration of different autonomous subsystems are the major requirements for ensuring the quality of such systems. Each autonomous subsystem can be verified by different VVAs. These inter- and intra-subsystem verification processes results in a complex and multifaceted quality assurance process for the complete system.

Space industry like any other industry is evolving and constantly facing new political and market pressures. The trend has been that the traditionally high quality requirements remain but with increasing demands for lower development costs and faster delivery times. This requires an in-depth analysis of their VVAs and their approaches towards ECSS standards.

This paper presents various challenges which the space industry is facing due to the demands of ECSS, especially considering VVAs. There are three main contributions of this paper. First, it presents identified and prioritized challenges in VVAs of different space organizations. Secondly, it presents the effects of ECCS standards on VVAs and finally it discusses some proposed solutions.

The next section includes an introduction to two companies collaborating in this study, while Section III explains the design of the study. Section IV describes the results and analysis, and Section V discusses main challenges and issues. Section VI describes the challenge-cause analysis of each company, Section VII outlines the solutions based upon identified challenges and issues. Finally, Section IX concludes the study.

II. Case Companies

The study was conducted at two Swedish space companies that are developing software and hardware for the space industry. What follows is a brief introduction about each company.

A. RUAG Aerospace Sweden AB

RUAG Aerospace Sweden AB (RUAG) was formerly known as SAAB Space AB but was acquired by RUAG Aerospace in 2008. RUAG has a very long and vast experience concerning design, development and delivery of both hardware and software for computer and data handling products in space programs. The main product areas are data management systems, fault-tolerant computers and processor products, payload control computers, data processing and small mass memories. The software developed by RUAG for these computers is in the range from small BIOS software to full application software, but the main focus is on embedded and real-time software. The software development process is based on the ECSS standards, mixed with an integration-driven development approach.

B. Space Division at Swedish Space Corporation

The Space Division at the Swedish Space Corporation (SSC) develops software and hardware for space applications, such as for example the satellites Prisma, Small-Geo and Smart Olev. SSC is a system integrator and supplier for small and micro-satellites. They are also specialized in developing attitude orbit and control systems and on board data handling units. In recent years they have changed their software processes to be more agile, by using Scrum as a project management model and test driven development as an engineering model [8–10].

III. Design of the Study

A. Research Questions

In this study, we aim to answer following questions:

RQ1 What is the efficiency of current V&V activities used in space industry? By answering this question, we shall be able to identify current VVAs used in space industry and their efficiency. Defect logs and related documents will be analyzed for the purpose.

RQ2 What are the effects of the ECSS standards on the V&V processes? Companies developing space applications for European Space Agency (ESA) have to follow ECSS standards. Answer to this question will help us to understand the requirements of the ECSS standards and how it affects quality of the software and efficiency of the software development team.

RQ3 What are the challenges concerning V&V processes and their usage in the space industry?

RQ4 Is it possible to propose solutions for challenges identified in RQ3? If yes, what possible solutions are there?

B. Research Design

The experience drawn on in this research is part of a project launched at RUAG Aerospace Sweden AB (RUAG) and Swedish Space Corporation (SSC) to create more efficient VVAs, in general, and within ECSS projects, in particular. The study focuses on experiences from their ECSS projects and VVAs used in those projects. To answer the above questions the study is organized in three steps: *i*) Preliminary investigation. *ii*) Analysis and solutions identification. *iii*) Evaluation of solutions.

During the *preliminary investigation*, a web-based survey and an in-depth review of documents at SSC and RUAG was conducted to receive a certain level of understanding about VVAs and their efficiencies. The in-depth review of documents provided a description of processes and the software tools used. A literature review of state-of-the-art was carried out, simultaneously, to identify common defect types, defect detection techniques and strategies to combine the VVAs used.

According to [11, 12], a literature survey of theoretical knowledge and published practices must be complemented with industry observation to find out the commonalities of a specific problem.

Based upon the results of the web-based survey and the document analysis, a semi-structured interview was prepared and a questionnaire developed. The semi-structured interviews were performed with V&V experts at SSC and RUAG. These interviews helped in providing insight knowledge about the variations, artifacts, and complexities of the state-of-the-practice.

In the next step, *analysis and solution identification*, a connection between VVAs and defect types, effects of he ECSS standards on VVAs, issues/challenges in V&V processes and possible solutions were identified.

In the last step, an *evaluation of solutions* was performed. The proposed solutions were presented at both SSC and RUAG and feedback was collected from V&V experts using questionnaires and through informal discussions. Solutions were then further refined accordingly.

To increase the validity of the results we have used triangulation, i.e. a variety of research methods. We combined a questionnaire with semi-structured interviews and document analysis:

1) Web-Based Questionnaire: A web-based questionnaire was administered to relevant personnel at the two case companies. The questions were developed to determine the role and activities of the respondents, and their knowledge and views on ECSS in particular and on VVAs in general. A total of 37 respondents (18 at SSC and 19 at RUAG) answered the questionnaire. The answer frequency was 32.73% at SSC and 59.38% at RUAG. The low answer frequency at SSC can partly be explained by the fact that at SSC it was distributed more widely and, thus, some of the receivers might not have been in the target group.

2) Semi-Structured Interviews: Semi-structured interviews were conducted with a total number of 17 interviewees (9 at SSC and 8 at RUAG). The interviews were between 45 and 80 minutes in length. One researcher posed questions from a prepared list and the other researcher recorded the interviews. The interviews were transcribed and individually

Table I: ECSS standards.

ECSS issues	SSC	RUAG
Knowledge	2.1 (I know roughly what it is about)	2.9 () (I know its contents and how it affects software dev. activities)
Effect on software development	1.8 (Low)	2.9 (High)
Effect on software quality	2.9 (Low)	3.4 (High)
Effect on efficiency in software dev.	2.4 (Somewhat negative)	2.0 (Somewhat negative)

summarized by the two researchers. They summarized the transcriptions independently and then discussed their results until consensus was reached. The criticality level for each of the issues and challenges uncovered, were judged on a scale from general, important to critical, based on how frequently it was mentioned by different respondents and how important they judged it to be.

3) Document Analysis: Documents like e.g. software development plans, software verification and validation plans and software quality assurance plans from both companies were analyzed. Initially, these documents provided the basis for interviews and later they were complemented with the data of questionnaire and interviews.

IV. Results and Analysis

A. Web-Based Questionnaire

The questionnaire was divided into four main themes. The first theme focused on ECSS standard, the second on the effectiveness of the practiced VVAs, the third on the effort required for VVAs and, finally, the fourth on changes that could be made with respect to efforts if the companies would not need to take ECSS standards into consideration. To compare questionnaire results from both companies, a weighted average for each theme is calculated. The rest of this section presents the comparisons in tabular form.

1) Theme 1: ECSS Standards: Theme 1 of the survey is related to ECSS standards. The responses were given a weight from 1 to 5, where 1 being the lowest and 5 being the highest. Table I summarizes the results from this theme.

Table I indicates that there are small differences in the knowledge distribution of ECSS among both companies, for SSC the average is more towards ‘they know roughly what it is about’ and for RUAG it is more towards ‘they know its contents and how it affects software development activities’. SSC personnel has an opinion that the effect of ECSS on software development is low but RUAG personnel in general believes it to be high. It also shows that both companies agree that ECSS has ‘positive effects’ on the software quality but the effects on the efficiency of software development is ‘somewhat negative’.

2) Theme 2: Effectiveness of VVAs: Theme 2 of the survey is related to the effectiveness of VVAs. The responses were given a weight from 1 to 4 (1 being very ineffective and 4 being very effective). Table II summarizes the results from this theme.

Table II: Effectiveness of VVAs. (The corresponding variance is stated within parenthesis.)

VVA	SSC	RUAG
Requirement review	3.0 (0.5)	3.1 (0.1)
Design review	3.0 (0.5)	2.8 (0.4)
Code review	2.7 (0.2)	3.4 (0.4)
Unit testing	3.5 (0.7)	3.1 (0.4)
Integration testing	3.5 (0.9)	2.7 (0.9)
System testing	3.4 (0.9)	3.1 (0.3)
Validation testing	3.0 (1.0)	3.3 (0.5)
Acceptance testing	2.9 (1.1)	2.2 (0.2)

Table III: Effort required for VVAs. (The corresponding variance is stated within parenthesis.)

VVA	SSC	RUAG
Requirement review	2.6 (0.6)	2.6 (0.8)
Design review	2.6 (0.4)	2.4 (0.6)
Code review	2.6 (1.3)	2.8 (0.3)
Unit testing	3.1 (0.6)	3.2 (0.5)
Integration testing	2.8 (0.2)	2.9 (0.5)
System testing	3.3 (0.4)	3.6 (0.5)
Validation testing	3.0 (0.4)	3.8 (0.1)
Acceptance testing	2.8 (0.6)	2.7 (0.5)

For RUAG the most effective VVAs are considered to be ‘code review’ and ‘validation testing’, whereas for SSC ‘unit testing’ and ‘integration testing’ are considered to be more effective than other VVAs. The least cost effective VVA according to RUAG is ‘acceptance testing’ whereas for SSC it is ‘code review’.

3) Theme 3: Effort Required for VVAs: Theme 3 of the survey is related to the effort required for using different VVAs. The responses were given a weight from 1 to 4, where 1 being ‘very low effort’ and 4 being ‘very high effort’. Table III summarizes the results from this theme.

For RUAG, validation testing and system testing requires more effort compared to other VVAs, whereas for SSC it is system testing. Both companies believe that requirements review and design review requires less effort compared to other activities.

4) Theme 4: Change in Efforts for VVAs if ECSS is not relevant: In Theme 4 of the survey we focus on the change in effort for VVAs, if ECSS would not be a factor. The responses were given a weight from 1 to 4, where 1 being ‘would not perform the activity at all’, and 4 being ‘would put more effort on the activity’. Table IV summarizes the results from this theme.

RUAG would like to put more effort on integration testing and less effort on acceptance testing, whereas SSC would like to put more effort on unit testing and requirements review, and less on e.g. acceptance testing and unit testing. All this of course, if the companies would *not* need to take ECSS into account!

The results from the interviews and document analysis are presented in the next section as challenges and issues.

Table IV: Change in effort for VVAs, if ECSS is not a factor. (The corresponding variance is stated within parenthesis.)

VVA	SSC	RUAG
Requirement review	3.4 (0.4)	3.3 (0.4)
Design review	3.1 (0.3)	3.0 (0.5)
Code review	3.0 (0.6)	2.6 (0.2)
Unit testing	3.4 (0.3)	2.5 (0.4)
Integration testing	3.3 (0.2)	3.6 (0.4)
System testing	3.1 (0.1)	2.7 (0.5)
Validation testing	3.0 (0.0)	2.7 (0.5)
Acceptance testing	3.1 (0.1)	2.1 (0.7)

V. Challenges and Issues

A. ECSS Standards

Table V summarizes the challenges and issues regarding ECSS which were discovered during the case studies. They are sorted from more critical to less critical. The empty cells indicate that it was not an issue or challenge for that particular company. Each challenge/issue will now be presented in more detail.

Reusability and ECSS Standards: RUAG reuses document templates for ECSS between projects but have issues in reusing source code. ECSS allows for reusability but other requirements in the standards makes reusability difficult to achieve. Mainly this is because the reused part has to be fully verified and validated in the new context and sometimes this generates more work than re-implementing everything. The European space industry has traditionally been skeptical towards the reuse of source code since that was one of the main causes of the Ariane 5 mid-air explosion [13]. However, there have been recent results in clarifying this situation by proposing updates to the standards [12]. It is not clear if and how these proposed updates affect cost effectiveness. Obviously, high costs of compliance when reusing software might defeat the purpose of reuse.

Both companies agree that since space projects are similar to their previous projects they can benefit a lot from reusing artifacts from the previous projects, but they feel that improvements are needed in this regard. In case of commercial-off-the-shelf software, it is not justifiable to verify them as per the requirements of ECSS Q-ST-80C [14] because they have been used by other companies and continuously verified and validated.

Documenting Compliance Consumes QA Resources: The major problem the two companies have is the high requirement on detailed documentation and proofs of standard compliance, which takes resources away from actually performing quality assurance and verification and validation activities. This does not seem to be addressed by the ECSS standards update that is in progress. Rather the latest ECSS update, version C, is more detailed and specific on which processes and methods must be followed, how things are to be performed and requires more detailed documentation. Without going into details, it seems to be inspired by the Galileo Software Standard (GSWS) [15] which, like the ECSS, has extensive backing from the European Space Agency (ESA). RUAG considers the GSWS to be more formal and prescriptive than the ECSS, thus having a likely even higher cost for compliance.

ECSS can be misused as a marketing tool when the developing organizations focus on certain activities just to show off that they are fully compliant. This adds to the unnecessary costs as some of these activities do not affect the quality.

Interpretation Differences: Another problem with ECSS is that it can be interpreted differently by different people and organizations. Even though it creates a common understanding between customers and suppliers, this understanding is too dependent on the actual persons involved. For example, at RUAG, there have been problems when different reviewers from the customer have different interpretations of what the ECSS standard requires. A problem that is even more visible when the reviewers are changed during a project or when different reviewers review different parts of the project. This, of course, takes away resources that could have been put into increasing the quality of the software instead. The problem is even larger if we consider multiple projects. The interpretations and expectations on ECSS compliance can vary a lot between projects even if the same customer is involved.

Incremental Development and ECSS: It is difficult to work in increments when following ECSS. This is because of a legacy in the standards of a traditional, linear, waterfall process and because of the requirement on external reviews. As long as a company follow the requirement on external reviews, the customers allow an incremental development process. However, the external reviews limit the extent to which the increment-driven development can be used e.g. requirements have to be assembled early in the project for the external preliminary design review, so the incremental approach can only be used for detailed design, implementation and testing, not for the requirements. Also, if the customers require a separate detailed design review, the same limitation applies to detailed design and further constrains the use of increments. So there is an immediate mismatch with ECSS standards if a company moves away from a waterfall-like process.

Differences with Galileo Software Standards: In some projects RUAG have been following the Galileo Software Standard (GSWS) instead of ECSS. GSWS is based on ECSS and can be considered a tailored version of ECSS. However GSWS tailors parts of ECSS that previously were open to interpretation (read: decreasing flexibility). At RUAG, they consider GSWS to be stricter but more explicit and clearer than ECSS. By following GSWS, RUAG has changed their internal processes so that they are now more ECSS compliant by default. However, one important difference between the standards is that GSWS requires independent module/unit testing (there is no such requirement in ECSS), and this

Table V: Challenges and issues related to ECSS standards.

Factor	Challenge/Issue	SSC	RUAG
Reusability	Documenting quality when reusing development artifacts	Critical	Critical
Resource-intensive	Showing compliance takes resources from increasing quality	Critical	Important
Interpretation	Difference in interpretation of ECSS	Important	Critical
Increments	Limited support for (integration-driven) development in increments		Critical
Galileo standard	Differences between ECSS and Galileo		Critical
Knowledge	Distribution of ECSS knowledge in organization	Critical	
Innovation	ECSS limits innovation in processes, methods and tools	General	Important
Inflexibility	Hard to make changes/introduce new requirements during project		Important
Requirements	Documenting requirements for compliance proof	Important	
Tailoring	Unclear how to tailor ECSS	General	

of course leads to an increased work load in the respective organizations.

Knowledge Distribution: At SSC the ECSS knowledge is unevenly distributed and concentrated on few individuals. The explanation for this is that SSC have more projects where ECSS compliance is not a requirement. One side-effect to this is that it can create tensions between different projects since some projects will need more resources in order to comply with ECSS. On the other hand, one way to potentially solve these type of problems is to introduce handshaking using implementation proposals [16], a technology developed to enable synchronization between development units and departments at a large multinational company. It has proven to be a very cost effective way to enable joint understanding of complex problems in the development of safety critical systems.

Efforts on Innovation: ECSS helps the companies making sure they do not miss important aspects, but the standards make it hard to introduce new processes, methods and tools. Primarily this is because a lot of activities are done only to show compliance, which does not affect the quality of software, while still requiring much resources. Thus there is less time to consider and implement improvements. A standard, by its very nature, also restricts the introduction of unknown methods and tools. As an example from SSC, they are introducing model-driven development, with automated code creation from models, to increase productivity and quality, but it has not yet been fully accredited by ESA. At the moment, it is obviously not an efficient use of resources to having to prove code coverage and verify automatically generated code.

Tailoring of ECSS: Tailoring of ECSS, according to project needs, is very important but sometimes it is already done for RUAG by their customers since they are allowed to deviate a little from ECSS; if the customer is confident in their work, has worked with them before and they have a good relationship. In that case they are able to focus on technical issues that further improves the quality.

The inability to do tailoring of ECSS generates a lot of work and extra costs. In some cases, if a customer is less technically inclined and have less knowledge about ECSS, they are afraid to deviate from the standard and thus require a very strict interpretation of the standard.

Inflexibility and Documentation of Requirements: At RUAG they find it hard to make changes to requirements during an ECSS project. SSC, on the other hand, has successfully introduced a more agile process, even in their ECSS projects, and the company does not seem to have the same problems. However, a related problem at SSC is that they are not clear on how to document requirements and requirement changes in a way that compliance can be proven.

ECSS favors a waterfall-like development methodology. It has strict toll gates, e.g. a preliminary design review and a detailed design review, and do not allow implementation to begin before these reviews. RUAG wants to have detailed design review in small steps focusing on parts which are to be implemented and in some projects they have successfully been able to do these design reviews and implementation in parallel.

One possible way to enable some requirements flexibility, but have a rigid long-term prediction of what functionality is delivered and at what quality, is the utilization of requirements abstraction levels. Gorschek et al. [17, 18] developed and tested a model for breaking up requirements into different levels of abstraction. One aspect of this is that overall feature level requirements can be rigid and stable, while lower level requirements could be seen as black-box and irrelevant across development units. This way flexibility and change can be achieved on lower levels without heavy change request handling by central control units.

B. *VVAs in Practice*

Table VI summarizes the challenges and issues regarding VVAs in practice. Each challenge/issues will now be presented in more detail.

Unstable and Non-Testable Requirements: Both companies have issues in writing requirements. Functional and non-functional requirements are more or less mixed when they receive them from the customers. Often the customers also tend to forget some of the non-functional requirements as well.

There are three levels of requirements at both companies. Mission level or equipment level comes from the customer, which are then broken down to system level requirements. The system level requirements are then forwarded by the systems manager to the developers who further break them

Table VI: Challenges and issues related to VVAs in practice.

Factor	Challenge/Issue	SSC	RUAG
Requirements	Unstable and non-testable requirements	Critical	Critical
Testing environment	Defects in testing environment and tools	Important	Critical
Integration testing	Limited focus on integration testing of software components		Critical
Reviews and inspection	Inadequate internal formal reviews and inspection	Critical	General
Unit testing	More focus on structural coverage than black box testing		Important
Independent V&V	Test cases are not reviewed by independent developer/tester	Critical	

into implementation or unit level requirements. There are always issues in passing information from one level to another due to gaps in communication (also in this case [16] might offer a possible solution).

Defects in Testing Environments and Tools: Both companies have had problems with in-house developed tools. In some cases it requires too much effort to improve them, according to project requirements, and in other cases engineers find defects in their testing environments and tools and do not generate software problem reports (bug reports).

Limited Focus on Integration Testing of Software Components: At RUAG they wish to improve integration testing of software modules. At the moment they combine it with the validation testing of hardware. One of the reasons for this is that they mostly work with hardware drivers and in these cases it is more efficient to test the integration of software modules when performing hardware validation activities. However, at the same time they also develop application software, which requires the testing of components separately.

Inadequate Internal Formal Reviews and Inspection: RUAG consider themselves to be good at reviews and inspections, but they would like to put in even more efforts in these activities. RUAG considers it to be the most cost effective VVA. The downside is the dependence upon the engineer doing it and that the list of issues they check is increasing and is updated constantly. SSC, on the other hand, does not focus too much on reviews or inspection. They have very informal checklists and have very limited reviews and inspections.

SSC values dynamic VVAs more. This is one of the main differences in the approaches of the two case companies. But they do not have figures or measurements to actually know which activity is more effective in finding defects in a cost efficient way.

More Focus on Structural Coverage than Black Box Testing: In both companies, unit testing is mostly performed by the engineers themselves. (This also depends on the requirements from their customers.)

At RUAG, they focus much on coverage statistics. The focus of tests is more towards structural coverage and white box testing. Maybe as a consequence to this, the engineers look at what the code *does* instead of looking at what the code *should do* and test that. SSC, on the other hand, is using a more test driven development approach—the engineer writing the code will start by creating unit tests—and hence have implicitly a more black box perspective.

Test Cases are Not Reviewed Independently: At SSC, the developers are responsible for the development and testing of units and there is no independent verification of code or test cases at this stage. There are chances that defects may be missed and, hence, slip through to later stages. RUAG uses independent review of the code and in some cases they also have reviews for test cases, depending upon the requirements from customers. There are, however, cons of having complete independent verification e.g. communication issues and loss of information.

C. *Efficiency in VVAs*

Table VII summarizes issues/challenges regarding efficiency of VVAs.

Insufficient Measurements: Both SSC and RUAG have insufficient measurements in their VVAs. They do not measure the efficiency of different activities and do not even calculate the number of defects found at different levels. They need to, more explicitly, measure results of what they are doing and why they are doing it. They do not have a formal list that says this category of faults should be detected at this stage and so on. But they have more or less an implicit list for doing unit testing.

Although both companies do not have any measurements, RUAG have an opinion that more defects are found in inspection and reviews than in module testing. They spend a lot of time in unit testing and its perceived efficiency is low. However, SSC thinks that unit testing brings more value. In the end, both companies are interested in having measurements performed that are easy to use and follow up.

Faults Slip Through Different Stages: In both companies they find defects that are not local to the specific stage. The interviews and the data analysis gave at hand that this behavior most likely depended on that, in the case of SSC, the testing environment was not fully representative of the target environment, and in the case of RUAG, that they focused too much on coverage statistics during unit testing. Of course, other reasons for faults to slip through were, as is very common, tight schedules during the project.

Neither company estimates the costs of finding defects in different phases. A metric could be obtained through reporting systems or by expert judgments. If they would evaluate the costs of finding defects in different phases then an improvement potential could be determined by calculating the cost of finding defects in specific phase to the cost of

Table VII: Challenges and issues related to efficiency of VVAs.

Factor	Challenge/Issue	SSC	RUAG
Measurements	Insufficient measurements	Critical	Critical
Fault-slip-through	Fault-slip-through among different stages	Critical	Critical
Defect classification	Vague classification of defects	Important	Critical
Involving V&V experts	V&V expertise is invited to the later stages only	Critical	Important
Initial framework	Inappropriate time for initial framework	General	

finding the similar defects in other phases. Hence, a total improvement potential could be calculated [19].

Vague Classification of Defects: The classification of defects in both companies is very vague. It is based upon the severity which, in its turn, is dependent upon the person classifying defects. They use different tools and reporting systems for this. In large projects they cover the same things again and again and their processes check similar things at different stages, which increases the cost and do not improve overall quality.

There is no clearly defined strategy about what kind of defects should be captured at which stage. There is no mapping between what type of tests a phase should cover and which faults should be found when executing those tests.

V&V Experts are Involved in the Later Stages Only: V&V experts are not involved in the early stages of the project in both companies. This may result in unstable requirements and cause problems later in the project as the engineers, focusing on the unit level, do not have the full picture and often the project managers do not have a clear idea about the technical constraints. If a validation team is involved in early stages it might be easier later to perform validation and even reuse artifacts.

In short, V&V experts are considered second class engineers and are not involved actively in the early stages of the software development.

Inappropriate Time for Initial Framework: At SSC, they do not focus much on requirements review and spend less time in the requirements phase. In some cases engineers implement functionality which does not connect to a requirement or they make an experienced guess when implementing functionality.

VI. Challenge-Cause Analysis

To better understand the dependencies among challenges and to find their causes a challenge-cause analysis is performed using a Current-Reality Tree (CRT) [20]. A CRT considers multiple challenges at the same time and is very helpful in improving systems and organizational problems by identifying the root causes of those challenges. The identified challenges are called UnDesirable Effects (UDEs) and are then traced to root causes. Figure 1a represents the CRT diagram for SSC while Figure 1b represents the CRT diagram of RUAG. The boxes with grayish backgrounds are identified as UDEs.

As a support tool, when creating the CRTs, cumulative voting on the three themes was performed among 14 developers at SSC (please see a description of the $100 test in [21]). The Nemenyi-Damico-Wolfe-Dunn (joint ranking) test [22, pp. 242–244] was used to examine which particular questions differ in each theme ($p > 0.01$). The scripts for reproducing the results, using R [23], can be downloaded at [24] (each file, responding to a theme, also contains the questions asked, the vectors containing the distribution of the respondents' $100 and the results of executing the scripts).

VII. Recommendations Based on Identified Challenges

This section discusses recommendations regarding how the identified challenges can be addressed. By the help of a challenge-cause analysis we conclude that both organizations are facing problems due to three main causes: ECSS standards, faults slip through different stages and inappropriate selection of cost-effective VVAs. The rest of this section discusses the recommendations to alleviate the identified challenges.

A. ECSS Standards

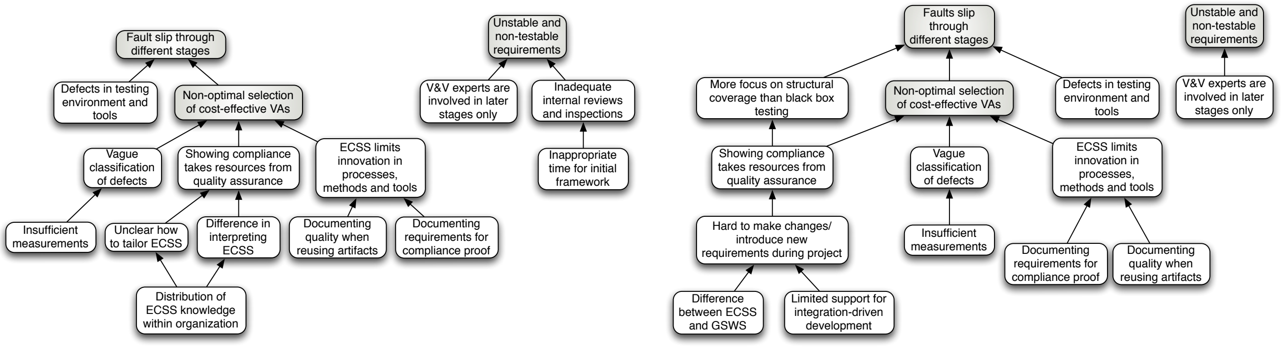
Table VIII summarizes the recommended solutions for different stakeholders in the ECSS standards. For each challenge we list solutions in three different categories based upon their relevance for: Development organization (RUAG and SSC in this case), customer (ESA or other organization stating the requirements for a development project) and ECSS (the standards body).

B. Fault-Slip-Through Measurements

Analysis of the documentation and interview data provides a sign of too little too late concerning the involvement of V&V experts. The recommendation in this case would be to: i) Early on involve V&V experts to, especially, improve the companies' requirements phase. ii) Focus more on reviews and inspections (especially in the case of SSC). iii) Have appropriate time set aside for constructing an initial framework that could make requirements more non-volatile.

Faults slipping through different stages is one of the biggest challenges for both companies and the analysis indicates that the main reasons for this are defects in the testing environment and the inappropriate selection of VVAs.

In RUAG they focus more on coverage statistics but by following a more test-driven development process they could gain an increased black-box perspective and, thus, there could be less of a chance to have faults slipping through to the later stages. ECSS on one hand wants to


Fault slip through different stages
Defects in testing environment and tools
Non-optimal selection of cost-effective VAs
Vague classification of defects
Showing compliance takes resources from quality assurance
ECSS limits innovation in processes, methods and tools
Insufficient measurements
Unclear how to tailor ECSS
Difference in interpreting ECSS
Documenting quality when reusing artifacts
Documenting requirements for compliance proof
Distribution of ECSS knowledge within organization
Unstable and non-testable requirements
V&V experts are involved in later stages only
Inadequate internal reviews and inspections
Inappropriate time for initial framework
Faults slip through different stages
More focus on structural coverage than black box testing
Non-optimal selection of cost-effective VAs
Defects in testing environment and tools
Showing compliance takes resources from quality assurance
Vague classification of defects
ECSS limits innovation in processes, methods and tools
Hard to make changes/ introduce new requirements during project
Insufficient measurements
Documenting requirements for compliance proof
Documenting quality when reusing artifacts
Difference between ECSS and GSWS
Limited support for integration-driven development
Unstable and non-testable requirements
V&V experts are involved in later stages only

(a) SSC

(b) RUAG

Figure 1: Challenge-cause analysis for SSC and RUAG respectively. (Boxes with gray background are identified as undesirable effects.)

Table VIII: Recommended solutions for different stakeholders and challenges.

Challenge	Development organization	Customer	ECSS
Reusability			Clarify reuse of artifacts and VVAs. Evaluate cost effectiveness.
Resource-intensive	Document inefficiency and overlap between activities.		
Interpretation		Designate ECSS authority for each project.	
Increments	Describe alternative processes that work and point out ECSS mismatches.	Allow process experimentation.	Allow alternative processes and consider simpler/quicker ways for process evolution to occur.
Galileo standard			Clarify relationship and motivate differences between ECSS and other common standards.
Knowledge	Ensure broad ECSS knowledge even when non-ECSS projects are run in parallel.	Ensure coherent ECSS knowledge among reviewers and in projects.	Develop lightweight ECSS training material.
Innovation	Try alternatives in non-ECSS projects first and consider ECSS when evaluating them.		Ensure the standard is primarily goal-driven. Clarify explicitly for big trends, e.g. model-driven development, how they can be incorporated.
Inflexibility			Extend to agile and other alternative processes.
Requirements	Show alternative ways to document requirements for compliance.		Evaluate alternatives for documentation.
Tailoring		Describe requirements on tailoring documents.	Clarify how to tailor and document tailoring.

improve quality and at the same time it takes away much resources from quality assurance by prescribing certain activities which does not have any positive impact on the artifacts as such, e.g. documents and proofs that certain VVAs are performed accordingly. Insufficient measurements and vague classification of defects also have an impact on faults to slip through. Both the companies are looking for simple measurements so they may evaluate the efficiency of their VVAs, the improvement potential they can have and a method by which they can have a combination of VVAs to ensure that defects are covered.

Damm et al. [19] proposed a method at Ericsson AB for faults-slip-through measurements which has three steps, in the first step a strategy is developed regarding what should be tested at which phase. This will have a direct mapping to what type of faults a certain phase should cover. In the second step, average costs of finding defects in various stages is determined; this can be obtained through the reporting system or by expert judgment. In the third step, an improvement potential is determined by calculating the difference between the costs of finding defects at the stage they were found to the cost of finding defects at the stage where they slipped through. The approach describes definitions and instructions how to apply and follow up on measurements. But the pre-requisite for applying this method is a strategy and classification of defects and measurements about the cost of finding defects at various stages.

C. Strategy for the Selection of Cost-Effective VVAs

The CRTs in Figure 1 also indicate that both companies are facing problems due to inappropriate selection of VVAs at different stages of the software development life cycle. For example, at SSC, the engineer performs unit testing of code which can cause faults to slip through to the next stage. On the other hand at RUAG, code inspection is performed by an independent engineer at unit level to ensure full structural coverage, while lacking considerably in integration testing. Both companies are facing problems in identifying the appropriate VVAs at different stages and there is a need for a strategy to select appropriate VVAs.

There are some strategies focusing on the selection of VVA combinations. Barett et al. [11] use the idea of a mapping matrix for optimizing the testing process. The matrix is filled by placing VVAs and defect types in rows and columns, respectively. If any VVA has the ability to detect a specific defect type, then the cell representing that VVA (row) and defect type (column) is ticked off. Of course, there are other models and frameworks as well; however each having its drawbacks for the participating companies in this study:

Wagner's model of quality economics presents cost vs. benefit analysis by using more detailed metrics and equations [25]. This model requires a lot of data initially and cannot be a candidate strategy for the selection of cost-effective VVAs because of lack in data at both companies.

Murnane et al. in [26], present a method for the selection of test cases by tailoring black box testing. The limitation of this method is its focus on only one aspect of a V&V process.

In the end, our investigation ended up with one candidate solution (presented next), which was introduced to the participating companies and their employees, with the result that an implementation is now planned to take place.

Wojcicki and Strooper's Model: The strategy presented by Wojcicki and Strooper [27], for the selection of cost-effective VVAs, is a candidate solution. It aims at selecting and evaluating different combinations of VVAs, in four steps, by focusing on maximizing completeness and minimizing effort thus reducing cost and enhancing efficiency. The systematic way in applying empirical information makes this strategy a competitive approach. The strategy analyzes different combination of VVAs by exploring effort and defect detection effectiveness of the VVAs, iteratively. Each iteration determines whether the particular combination produces expected results or if adjustments should be made. The model has four steps:

1) Pre-selection: Collect cost-effect information.
2) Argument 1, Maximize completeness.
3) Argument 2, Minimize effort.
4) Post-selection: Updating cost-effective information.

There are mainly three reasons for selecting this model as a candidate solution: **Maximizes completeness and minimizes effort.** Step 2 of this model requires the combinatorial selection of VVAs to ensure that all the defects are covered by VVAs. Step 3 ensures to minimize efforts by selecting the combination of VVAs which requires the minimum effort from the combinations selected in step 2. **Supports decision by empirical information.** Both SSC and RUAG do not have enough initial data in terms of defect logging and efficiency of VVAs. This model is flexible enough as it can be initiated by expert opinions, but they will have empirical data right after the first iteration. This data can also be used to determine whether the selection produced the expected results. **Scalability.** The model is flexible to be used at any V&V stage, i.e. unit, integration or system level.

VIII. Validity Threats

To increase the validity of results we have used triangulation, i.e. a variety of research methods. The results are based upon the combination of questionnaires, document analysis and semi-structured interviews. External validity is a valid threat for this research because the case companies are based in Sweden and have relatively small software divisions. They may have different perceptions about ECSS and have different issues in their V&V activities. Surveys at other companies can improve the external validity of this research.

At SSC, the survey questionnaire was sent to a broad set of employees and, hence, this might be one of the reasons why they showed less knowledge of ECSS standards. Moreover, all the interviews were recorded and chances are that respondents may be intimidated by that. In order to reduce the effect of that, before starting every interview their permission was taken and they were assured that recorded files would only be available to the researchers and only the general results and conclusions would be available to others.

IX. Conclusions

This paper describes the results of two industrial case studies at companies in the European space industry. Based on a triangulated research method using three sources of data it presents the issues and challenges that were identified. We describe the possible ways that the main stakeholders, the developing organizations, the customers and the ECSS standards organization, can work together to address these issues and challenges. We also discuss the possible ways forward to reach the goal of creating a more cost-effective verification and validation activities framework for the space industry.

Acknowledgment

This work has been supported by the Swedish National Space Board.

References

[1] European Cooperation for Space Standardization, ECSS Secretariat, ESA ESTEC, P.O. Box 299, 2200 AG Noordwijk, The Netherlands, *ECSS-S-ST-00C System—Description implementation and general requirements*, July 2008.

[2] M. Jones, U. K. Mortensen, and J. Fairclough, "The ESA software engineering standards: Past, present and future," in *Software Engineering Standards Symposium and Forum, 1997. 'Emerging International Standards'. ISESS 97., Third IEEE International*, pp. 119–126, 1997.

[3] ESA Board for Software Standardisation and Control (BSSC), European Space Agency/Agence Spatiale Européenne 8–10, rue Mario-Nikis, 75738 PARIS CEDEX,

France, *ECSS PSS-05-0—ESA Software Engineering Standards*, February 1991.

[4] S. R. Rakitin, *Software verification and validation for practitioners and managers*. Norwood, MA, USA: Artech House, Inc., 2nd ed., 2001.

[5] G. J. Myers, "A controlled experiment in program testing and code walkthroughs/inspections," *Communications of the ACM*, vol. 21, no. 9, pp. 760–768, 1978.

[6] M. Wood, M. Roper, A. Brooks, and J. Miller, "Comparing and combining software defect detection techniques: A replicated empirical study," in *ESEC '97/FSE-5: Proceedings of the 6th European SOFTWARE ENGINEERING conference held jointly with the 5th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, (New York, NY, USA), pp. 262–277, Springer-Verlag New York, Inc., 1997.

[7] G. Brat, E. Denney, D. Giannakopoulou, J. Frank, and A. Jonsson, "Verification of autonomous systems for space applications," in *Aerospace Conference, 2006 IEEE*, p. 11, 2006.

[8] D. Astels, *Test driven development: A practical guide*. Prentice Hall Professional Technical Reference, 2003.

[9] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas, "Manifesto for agile software development." http://www.agilemanifesto.org/, 2001.

[10] K. Schwaber and M. Beedle, *Agile software development with Scrum*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

[11] N. Barrett, S. Martin, and C. Dislis, "Test process optimization: Closing the gap in the defect spectrum," in *Test Conference, 1999. Proceedings. International*, pp. 124–129, 1999.

[12] M. Rodríguez, J. G. Silva, P. Rodríguez-Dapena, H. van Loon, and F. Aldea-Montero, "Reuse of existing software in space projects—Proposed approach and extensions to product assurance and software engineering standards," in *ICCBSS* (X. Franch and D. Port, eds.), vol. 3412 of *Lecture Notes in Computer Science*, pp. 258–267, Springer, 2005.

[13] M. Dowson, "The Ariane 5 software failure," *ACM SIGSOFT Software Engineering Notes*, vol. 22, no. 2, p. 84, 1997.

[14] European Cooperation for Space Standardization, ECSS Secretariat, ESA ESTEC, P.O. Box 299, 2200 AG Noordwijk, The Netherlands, *ECSS-Q-ST-80C Product assurance—Software product assurance*, March 2009.

[15] Galileo Industries, *Galileo Software Standard (GSWS)—GAL-SPE-GLI-SYST-A/0092*, 7th ed., May 2004.

[16] S. Fricker, T. Gorschek, and P. Myllyperkiö, "Handshaking between software projects and stakeholders using implementation proposals," in *REFSQ* (P. Sawyer, B. Paech, and P. Heymans, eds.), vol. 4542 of *Lecture Notes in Computer Science*, pp. 144–159, Springer, 2007.

[17] T. Gorschek and C. Wohlin, "Requirements Abstraction Model," *Requirements Engineering*, vol. 11, no. 1, pp. 79–101, 2005.

[18] T. Gorschek, P. Garre, S. B. M. Larsson, and C. Wohlin, "Industry evaluation of the Requirements Abstraction Model," *Requirements Engineering*, vol. 12, no. 3, pp. 163–190, 2007.

[19] L.-O. Damm, L. Lundberg, and C. Wohlin, "Faults-slip-through—A concept for measuring the efficiency of the test process," *Software Process: Improvement and Practice*, vol. 11, no. 1, pp. 47–59, 2006.

[20] H. W. Dettmer, *Goldratt's theory of constraints: A systems approach to continuous improvement*. ASQC Quality Press, Milwaukee, Wis., 1997.

[21] D. Leffingwell and D. Widrig, *Managing software requirements: A unified approach*. Addison-Wesley Professional, October 1999.

[22] M. Hollander and D. A. Wolfe, *Nonparametric statistical methods*. Wiley-Interscience, 2nd ed., January 1999.

[23] R Development Core Team, *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. ISBN 3-900051-07-0.

[24] R. Torkar. http://iaser.tek.bth.se/torkar/icst-package.zip, October 2009.

[25] S. Wagner, "Modelling the quality economics of defect-detection techniques," in *WoSQ '06: Proceedings of the 2006 International Workshop on Software Quality*, (New York, NY, USA), pp. 69–74, ACM, 2006.

[26] T. Murnane, K. Reed, and R. Hall, "Tailoring of black-box testing methods," in *ASWEC '06: Proceedings of the Australian Software Engineering Conference*, (Washington, DC, USA), pp. 292–299, IEEE Computer Society, 2006.

[27] M. A. Wojcicki and P. Strooper, "An iterative empirical strategy for the systematic selection of a combination of verification and validation technologies," in *WoSQ '07: Proceedings of the 5th International Workshop on Software Quality*, (Washington, DC, USA), p. 9, IEEE Computer Society, 2007.