ECSS Standard Compliant Agile Software Development

[An Industrial Case Study]

Ehsan Ahmad
Department of Computer Science and Engineering
Air University
Islamabad, Pakistan
ehsan.ahmad@mail.au.edu.pk

Bilal Raza, Robert Feldt
Blekinge Institute of Technology
SE-372 25 Ronneby, Sweden
bira07|rfd@bth.se

Tanja Nordebäck
Space Division
Swedish Space Corporation
Stockholm, Sweden
tanja.nordeback@ssc.se

ABSTRACT

Developing software for high-dependability space applications and systems is a formidable task. The industry has a long tradition of developing standards that strictly sets quality goals and prescribes engineering processes and methods to fulfill them. The ECSS standards is a recent addition, but being built on the PSS-05, it has a legacy of plan-driven software processes. With new political and market pressures on the space industry to deliver more software at a lower cost, alternative methods need to be investigated. In particular, the agile development processes studied and practiced in the Software Engineering field at large has tempting properties. This paper presents results from an industrial case study on a company in the European space industry that is using agile software development methods in ECSS projects. We discuss success factors based on detailed process and document analysis as well as empirical data from interviews and questionnaires.

Keywords

Case Study, European Cooperation for Space Standardization, Agile Paradigm

1. 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 pose additional challenges due to its inherent requirement that the end products should be highly dependable. The reliable protocols and specific space operations also differentiate them from other systems. Existing software engineering standards are not enough to cope with these challenges. The European Cooperation for Space Standardization (ECSS) has developed a set of standards for European space projects[2]. These standards are derived from PSS-05[1], an earlier space standard, which were more prescriptive, demanded heavy documentation and favored waterfall and incremental development models[3]. It also did not adequately discuss the relation and correspondence between overall space system development and its software development. Since PSS-05 was a primary input for ECSS, development activities in the space industry have the legacy of PSS-05[1, 3].

A number of factors such as increasing globalization, increased expectations on software systems in general, changes in political priorities and increasing competitiveness the space industry need to deliver software with more functionality but at lower costs. This requires new methods to be considered. Agile methodologies are gaining acceptance from a wide variety of software industries[6], however it is not clear whether agile methods can adhere to the requirements and pressures of the standards for dependable space systems. The main aim of this paper is to present the results of a study which analyze how a company in European space industry is using more agile software development methods while still adhering to ECSS standards.

The experience drawn on in this research is part of a project launched at Swedish Space Corporation (SSC) to create more efficient Verification and Validation Activities (VVAs), in general, and within ECSS projects, in particular. The study focuses on experiences from their ECSS projects and VVAs used in those projects. The Space Division at 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(TDD)as an engineering model[14, 15, 16]. So in the following, when we refer to agile methodologies, our focus is on these two agile practices. For most of their development projects they have to follow the ECSS standards. Selection and adaptation of a particular VVA at a particular stage of the project depends on having an understanding of the mapping between ECSS standards and agile practices used by SSC. The results of this study should ideally help other space companies understand if and how agile methodologies can be used in their software development.

The next section includes related work and brief introduction of ECSS standards. Section3 explains the design of the study. Section4 describes the results and analysis, and Section5 highlights the challenges and issues regarding ECSS standards compliant agile development. Section6 maps the requirements of ECSS standards and how they are met at SSC using agile practices. Section7 contains the discussion while Section8 concludes the study.

2. BACKGROUND

2.1 Related Work

Attaining an optimal balance between plan-driven and agility is an important factor for the success of dependable software projects [6]. The discussion about mapping standards like ISO 9001 and CMMI to agile ways of working started right after the inception of agile methodology. Tor et al. have summarized the work done for mapping ISO 9001 with agile software development in[7]. They pointed out that the main differences between agile development and ISO 9001 are the documents for reviews and proof of conformance. They also suggested some changes for both ISO and agile development to make them compatible with each other. In [6], Turner has discussed the components of CMMI process improvement and their relation to agile software development.

Both[6] and [7] had concluded that differences between plan-driven approaches (ISO 9001 and CMMI) and agile development are not insurmountable and agile development can produce enough documents for reviews and proof of conformance. ECSS standards are mainly based on ISO/IEC 12207[4, 5]. Although a large number of research papers like[8, 9] are aiming to reconcile ISO/IEC 12207 and agile, no explicit work has been done to discuss the issues of reconciling ECSS software standards and agile methodology according to authors' best knowledge. A study was conducted by Fátima et al.[13] that compares the management process of ECSS for software acquisition with PMBOK/DoD. But it is mainly focused on requirements of ECSS management processes (the M branch of the standard).

2.2 ECSS Standards

In 1993 the European Space Agency (ESA) along with other national space agencies and industries realized the need of a single coherent, recognized and accepted system of standards to replace the practice-based PSS-05 standard[2, 3]. The first document of this new system of standards, named European Cooperation for Space Standardization(ECSS), was introduced in 1996. The idea is that ECSS standards should continuously be created and updated to adapt to changing needs of the industry. A revision process started in 2006 and two batches of updates were released in 2008. ECSS standards divide activities into three areas: management, engineering and product assurance and has four levels:

- *Level 0*- discusses policy, architecture and objectives of ECSS
- *Level 1*- describes the strategy within management, product assurance and engineering by highlighting the requirements and interfaces of level 2 elements
- *Level 2*- explains objectives and functions of each domain. It is considered as branch-specific level
- *Level 3*- lists methods, procedures and tools to achieve the requirement of the level 2 documents. It is also known as technical domain specific level.

ECSS-E-40 and ECSS-Q-80 are related to software. ECSS-E-40 is based on ISO 12207 and allows suppliers to define their own standards, which are in compliance with or tailored to it[3, 1]. To ensure completeness and correctness, it forces suppliers for different types of review;Preliminary Design Review (PDR), Critical Design Review (CDR), Detailed Design Review (DDR)and Site Acceptance Test (SAT)etc. ECSS-E-40 is based on a recursive concept of customer-supplier relationship. A customer at one level can be a supplier for a higher level. According to clause 4 of ECSS-E-40[4], customer is responsible for defining both functional and performance requirements, interface between software components and interface between software and hardware while the supplier is supposed to maintain the interface with the customer to ensure the proper consideration of higher level system requirements. ECSS-Q-80 defines requirements for product quality assurance to ensure that the software development produces safe and reliable software[3, 5]. It focuses on identifying quality attributes, measurable quality objectives and set of metrics to verify these quality objectives.

3. DESIGN OF THE STUDY

3.1 Research Questions

We aim to answer the following research questions:

RQ1 What are the requirements of ECSS standards for software development, quality, verification and validation processes?

RQ2 What is the level of knowledge in the organization(SSC)about ECSS standards and their effects?

RQ3 What are the major concerns for dependable ECSS compliant agile software development?

3.2 Research Design

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:

3.2.1 Web-Based Questionnaire

In order to answer RQ2, a web-based questionnaire was administered to relevant personnel at the case company. 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 18 respondents answered the questionnaire and answer frequency was 32.73%. The low answer frequency can partly be explained by the fact that it was distributed more widely and, thus, some of the receivers might not have been in the target group.

3.2.2 Semi-Structured Interviews

To identify the major concerns of for dependable ECSS compliant agile software development(RQ3), semi-structured interviews were conducted with a total number of 9 interviewees. 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 summarized by the

Table 1: Knowledge and effects of ECSS standards.

ECSS issues	Responses	Weighted Average
Knowledge	18	2.1 (I know roughly what it is about)
Effect on software development	18	1.8 (Low)
Effect on software quality	15	2.9 (Low)
Effect on efficiency in software dev.	16	2.0 (Somewhat negative)

two researchers. They summarized the transcriptions independently and then discussed their results until consensus was reached.

3.2.3 Document Analysis

Documents like software development plans, software verification and validation plans and software quality assurance plans from SSC, were analyzed. Different ECSS standards like ECSS-E-40, ECSS-Q-80 were also analyzed to understand the requirements imposed by these standards(RQ1). Initially, these documents provided the basis for interviews and later they were complemented with the data of questionnaire and interviews.

4. RESULTS AND ANALYSIS

4.1 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. For understanding and analysis of questionnaire results, a weighted average for each theme is calculated.

Theme 1 of the survey is related to ECSS standards. Questions were asked about the knowledge of ECSS standards, and it's affects on *i*) software development *ii*) improving software quality *iii*) improving software development efficiency. Analysis of the responses to these questions, is one of the focal points for this position paper.

The responses were given a weight from 1 to 5, where 1 being the lowest and 5 being the highest. Table1 summarizes the results from this theme. To analyze questionnaire results, a weighted average for each issue is calculated. A significant segment of respondents (84%) said that ECSS has a positive effect on the quality of software and only 16% said that it has negative effects. Most of the respondents 84% said that the degree to which ECSS affects their software development processes is low or very low and only 16% say that it is high. It is quite interesting to find out that 50% of the respondents said that ECSS has positive effects on the efficiency of software development and the same percentage said that it has negative effects. The results from the interviews and document analysis are presented in the next section as challenges and issues.

5. CHALLENGES AND ISSUES

Table 2 summarizes the challenges uncovered as a result of document analysis and interviews. These challenges are based on the requirements of ECSS standards need to be fulfilled for ESA projects. The requirement considered as a challenge/issue was judged on how frequently it was mentioned by different respondents and how important the researchers judged it to be. Each challenge/issue will now be presented in more detail.

Software Management Process.

Supplier has to identify a suitable software life cycle process which has to be divided into phases and inputs,output and documents produced are to be defined for each phase. Discussion about the technical specifications based on the requirement baseline, must be started early in the life cycle process. A joint technical review process must be maintained throughout the whole life cycle for technical, interface and budgeting reviews, with customer cooperation.

Software Engineering Process.

Supplier is responsible for defining, establishing and documenting the requirements both functional and nonfunctional. Supplier develops man-machine interface mockup and customer evaluates it. Supplier is also responsible for transforming software requirements into high level software architecture by identifying software components and items for each component.Supplier has to develop and document a detailed design of each component by dividing it into small units so that they can be coded, compiled and tested. Each unit must be coded, tested and documented. Integration and testing must be done on both component and unit levels.

Software V&V Process.

The software validation process according to ECSS consists of validation process implementation and its activities, with respect to technical specification and requirement baseline. In order to implement the validation process, the supplier should first determine the effort required for it and then establish a process and document it. If required the supplier can have independent validation by a qualified organization. The supplier shall develop, document and develop set of test cases for software validation priority is given to the validation testing however, other validation methods such as reviews and inspection can also be used. The supplier shall also conduct test readiness reviews and critical design reviews.The software verification process according to ECSS consists of verification process implementation and its activities. In order to implement the verification process the supplier should first determine the effort required for it and then establish a process and document it. If required the supplier can have independent verification by a qualified organization.

Assurance Program Implementation.

Supplier is responsible for developing a comprehensive software product assurance plan which identifies resources and responsibilities, conduct planning, controlling and reporting product assurance, selection of development methods and tools, risk management and continuous process assessment and improvement.

Software Process Assurance.

Requirements are defined to confirm the suitability of software development process selected according to ECSS-E-40 with emphasis on dependability between hardware and soft-

Table 2: Requirements of ECSS standards considered as challenges for agile paradigm.

Factor	Requirement	Standard	Clause(s)
Software Management Process	Identify a suitable software development process and a technical review process	ECSS-E-40	5.3
Software Engineering Process	Define and document engineering process for requirements, architecture, design, coding and testing	ECSS-E-40	5.4, 5.5
Verification and Validation (V&V)Process	Determine the effort required to implement V&V process, Define the process and document it	ECSS-E-40	5.6, 5.8
Assurance Program Implementation	Develop a comprehensive software product assurance plan	ECSS-Q-80	5
Software Process Assurance	Confirm the suitability of software development process selected according to ECSS-E- 40	ECSS-Q-80	6
Software Product Quality Assurance	Identify the set of requirements to assure the quality of the final software product	ECSS-Q-80	7

ware, configuration management, and requirement for each life-cycle process.

Software Product Quality Assurance.

It includes a set of requirements to assure the quality of the final software product with focus on quality attributes and relation between them, measurable quality objectives and a set of metrics to verify quality objectives.

6. MAPPING OF ECSS STANDARDS AND AGILE PARADIGM

Table 3 summarizes success factors regarding how ECSS requirements can be addressed using agile paradigm. The rest of this section discusses each success factor in terms of agile practices being followed at SSC for ECSS compliant agile software development.

Software Management Process.

Sub-clauses 5.3.2.1 and 5.3.2.2 of ECSS-E-40 are related to the software life cycle management process itself. The initial Sprint planning meeting can be focused on the suitability of agile approaches by discussing literature, previous experiences and best practices of agile. SSC develops Software Development Plan (SDP) for each component, which is then validated by the prime contractor. Sprint planning meeting at SSC, is a platform used to discuss the inputs, outputs and documents produced for each sprint. The active participation of customer representative in communication and management practices of agile approach as described in[6], ensures that technical specifications and architectural infrastructure is according to the agreed requirements baseline. The empirical work conducted by Pikkarainen[8], a comprehensive literature review presented by Tor et al[7] has proved that different planning and review activities of agile approach can be mapped with reviews like PDR, DDR, CDR,SAT and a joint review process can easily be maintained in any kind of problem area.

Software Engineering Process.

Agile methodology, used at SSC is based on customer-supplier communication. One of the main critics of Scrum is its inability to produce a workable product during the requirements and architecture phases however, the sprint planning in Scrum and metaphor in eXtreme Programming (XP) [17] with frequent customer meetings (preferably face-to-face), enable them to understand each other and have a clear vision of the architectural infrastructure of the system[7, 8, 9]. A demonstrable layer of architectures can be delivered in the first sprint. SSC ensures to fulfill the requirements of this clause by participation of product owner (customer representative), Scrum master and Scrum team in daily Scrum meeting. Sprint planning at the start of each iteration, is used to manage the requirements and the product backlog is updated accordingly. Iterative delivery of software is useful to ensure that the architecture infrastructure actually works.

The iterative development in agile ensures the continuous application of reviews and metrics evaluation at the end of each sprint. Reviews are performed for top-level milestones. Product owner determines schedule, quality and functionally of each working unit on sprint bases. Each sprint ends with a full fledge working unit that is developed, tested, quality assured and according to standards. At SSC, the process of design is improved by planning and controlling it iteratively with the help of discussions between a self organizing team in daily stand up meeting in open office environment as explained by [6, 9]. The definition and review /justification documents mentioned in each sub-clause are maintained at the start and end of each sprint respectively to show the conformance.

Software V&V Process.

Software validation is about building the right product while software verification is about building the product right[10]. Agile methods integrate developers, testers and customers therefore the visibility of tasks and constraints is high within development teams. The reflection and post mortem analysis at the end of each sprint provide opportunities for identification and resolution of issues. Involving customers frequently validates the products and provides feedback in the process[6]. TDD produces fewer defects and is useful in verification[11]. SSC is using Scrum and TDD effectively to manage software verification and validation activities by determining the required effort efficiently. Verification experts at SSC are somewhat skeptical about the independent validation because of their bad experience, however, they also believe that independent validation is required at some stage but there is a chance of too much loss of information in having it completely independent. There were communication issues and delays in their projects due to complete independent validation.

An important characteristic of Scrum is frequent reviews. Team progress is reviewed as frequently as environmental complexity and risk dictates. The vital point of agile development is close monitoring and reviews of the process to discover problems early. These reviews are intended to discover non-conformity with the requirements[7]. The close

Table 3: Agile strategies for ECSS challenges in relation with SSC practices.

Factor	Agile Strategies	SSC Practices
Software Management Process	Management models like Scrum, Agile Modeling, DSDM	Develop a Software Development Plan for following Scrum for each project
Software Engineering Process	Techniques like Test Driven Development (TDD),Continuous Integration, Pair Programming	Sprint planning is done at the start of each sprint and the use TDD for development
Software Verification and Validation (V&V)Process	Agile paradigm forces to have visible goals and frequent reviews	Frequent review meetings in Scrum and TDD ensures process verification and minimizes product defects
Assurance Program Implementation	sprint planning meetings in Scrum	Practices like sprint planning meetings,daily stand-up meetings and sprint post-mortem analysis helps to assure this implementation
Software Process Assurance	Suggests daily stand-up meetings, sprint retrospectives in Scrum	Process assurance is achieved through daily stand-up meetings and frequent sprint retrospectives
Software Product Quality Assurance	Sprint retrospectives in Scrum	Frequent sprint retrospectives are used to identify success factors and failures

relationship between the customer and supplier increases the level of validation [6]. Design and development are reviewed between iterations jointly by the development team and customer. High-level design review is conducted at the start of the planning game and low-level design reviews are performed between iterations. Keeping track of design or development changes is also catered by these reviews[7].

At SSC, unit level development and testing is done by the same staff members, however at integration level, extra resources which are independent from the development are used. The product owner and all interested stakeholders are invited to attend the sprint review meetings, at the end of each sprint. The product owner determines which items on the product backlog have been completed in the sprint, and discusses with the Scrum team and stakeholders how best to reprioritize the product backlog to the next sprint.

Assurance Program Implementation.

Agile methodology starts with a planning phase and the issues like sprint length, number of sprints, documents to be produced, and quality measures etc. can easily be discussed and documented. One of the main reasons of introducing agile approaches at SSC was to cope with the problems caused by changes in requirements and schedules (requirements of sub-clause 5.5 of ECSS-Q-80). But these changes are normally known as short terms risks and the agile approaches focus on functional behavior of the system with no extra focus on long term risks[10]. Each sprint can be assessed at the end, in sprint review meeting. Lesson learnt session in the form of post iteration workshops, reflection workshops, and agile assessment are used to answer the questions related to assessment and improvement of working process on continuous bases[6, 8].

Software Process Assurance and Product Quality Assurance.

One of the major claims to agile process quality assurance is non-documented evolving process. But this evolving nature of agile process with the help of Scrum master and continuous assessment actually leads to a better quality assured process. Need for product quality assurance was a some agile approaches. Like XP enforces functional, acceptance and unit tests to ensure that the software product is according the quality standards accepted to the customer[9].

SSC uses common Scrum metrics like estimated team velocity, actual team velocity, sprint Burndown chart, product Burndown chart to measure the progress of team and quality of the process. TDD is used to fulfill single fault tolerant requirement of the dependable system. Process assessment and improvement is performed at Sprint retrospectives. The way the team worked together in the sprint is assessed. Positive ways of working together are identified and encouraged as future practice. Things that could work better are identified and strategies for improvement are defined and agreed upon.

Another important feedback cycle at SSC is daily stand-up meetings (Daily Scrum meeting). These meetings are very short and focused on a quick status update and discovering problems as early as possible (”done”, ”to-do”, and ”issues”). According to experience, some of typical team issues (e.g. one engineer can not proceed because he depended on a task from another, tester does not get info how to test new features, two engineers are working on solving the same problem etc) can be avoided if the team communicated them earlier. Scrum makes a lot of issues visible that exist within the team, makes visible the good and the bad, and giving the team the choice of elevating itself to a higher level. By having every member of the team see every day what every other team member was doing, the team's productivity, morale, adaptability, accountability and collaboration will significantly increase.

7. DISCUSSION

ECSS does not restrict suppliers to use any specific software development methodology. Its software engineering standards are mainly based on ISO/IEC 12207 [4, 5]. Requirements of engineering and management processes of both standards are almost the same. Studies like[6] and [9] have explained that ISO/IEC 12207 and agile are not contradictory to each other. Our findings support these studies, based on a study of a company in the high-dependability space industry.

The recursive customer-supplier model described in clause 4.2 of ECSS-E-40 highlights that ECSS focuses on strong customer-supplier collaboration. It engages customers in two ways: firstly by involving in defining functions and performance requirements with supplier (role as a customer) and secondly by interacting with customer to ensure proper actions against high level software requirements (role as a supplier). Agile is an advanced form of iterative development and primarily focuses on customer satisfaction by rapid and continuous delivery of working software[15]. Thus the recursive customer-supplier model of ECSS can be mapped easily with Scrum. Active participation of product owner in scrum planning, stand-up and sprint review meetings ensures that customer is aware of the status and quality of software. Product owner not only defines the goals of each sprint but also ensures that the most valuable component of product is produced first and product backlog is prioritized accordingly.

Standards like ECSS should keep on updated, depending upon the requirements from industry. They should leave space for companies for their innovation in solving issues and helping them staying active and adapting to new situations and requirements. If a methodology is working fine for a company and if they can provide evidence that it does not negatively affect the quality of the product standards need to be permissive. For any standard it is important that it stays current; the revision processes for ECSS that are now taking place are thus of high importance. They need to listen to practitioners within the industry when doing this work.

The companies should have more knowledge about ECSS standards especially about the tailoring according to their projects' needs. A company can be following the standard without knowing that they do. Isolating developers and testers from the burden of documentation and administrative tasks can be useful and might lead to increased quality. It has also been suggested by Theunissen et al. in[9]. SSC has initiated this by allocating a resource who makes strategies to make their processes in compliance with ECSS standards and keep track of its requirements.

8. CONCLUSION

This paper describes an industrial case study of a company in the European space industry that are using agile methods in the development of highly dependable software applications in projects that needs to adhere to the strict ECSS standards. Our results show that there is no strict contradiction in using agile methodologies in this way. There are also a number of areas in which ECSS maps very well to agile thinking and methodologies. Our detailed results outlines a number of success factors in making agile ECSS development work. It is also important that ESA and ECSS continue to update the standards and that they listen to practitioners in this process to ensure the standards stay current.

9. REFERENCES

[1] 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.

[2] 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.

[3] 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.

[4] European Cooperation for Space Standardization, ECSS Secretariat, ESA ESTEC, P.O. Box 299, 2200 AG Noordwijk, The Netherlands, *ECSS-E-ST-40C Space Engineering—Software*, March 2009.

[5] 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.

[6] R. Turner,"Agile Development: Good Process or Bad Attitude? ," in *Product Focused Software Process Improvement*, vol. 2559 of *Lecture Notes in Computer Science*, pp. 134–144, Springer, 2002.

[7] S. Tor,and H. G.Kjetil,"The Application of ISO 9001 to Agile Software Development ," in *PROFES'08*, vol. 5089 of *Lecture Notes in Computer Science*, pp. 371–385, Springer, 2008.

[8] M. Pikkarainen, "Mapping Agile Software Development onto ISO 12207," `http://www.scientificcommons.org/42262403`, February 2006. in *CiteSeerX - Scientific Literature Digital Library and Search Engine*, USA.

[9] W. H. M. Theunissen, D. G. Kourie, and B. W. Watson, "Standards and agile software development," in *SAICSIT '03: Proceedings of the 2003 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology*, (Republic of South Africa), pp. 178–188, 2003.

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

[11] L. Williams, E. M. Maximilien, and M. Vouk, "Test-Driven Development as a Defect-Reduction Practice," in *ISSRE '03: Proceedings of the 14th International Symposium on Software Reliability Engineering*, (Washington, DC, USA), p. 34, IEEE Computer Society, 2003.

[12] 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.

[13] M. de. Fátima, R. Arias, C. M. Hirata, E. T. Yano and B. M. Sakugawa "A Comparative Study between PMBoK/DoD and ECSS/Management Process for Software Acquisition," in *DASIA 2005: Data Systems in Aerospace*, ESA Special Publication, 2005.

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

[15] 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.

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

[17] K. Beck, and C. Andres, *Extreme Programming Explained: Embrace Change.* Addison-Wesley Professional, 2004.