Agile Software Development Methodologies for Safety Critical Systems

UNIVERSITÀ
DEGLI STUDI
FIRENZE

UNIVERSITÀ
DEGLI STUDI
DI PERUGIA

INdAM
Istituto Nazionale
di Alta Matematica

Università di Firenze, Università di Perugia, INdAM consorziate nel CIAFM

DOTTORATO DI RICERCA
IN MATEMATICA, INFORMATICA, STATISTICA

CURRICULUM IN INFORMATICA
CICLO XXXIV

Sede amministrativa Università degli Studi di Firenze
Coordinatore Prof. Matteo Focardi

Agile Software Development Methodologies for Safety Critical Systems

Settore Scientifico Disciplinare INF/01

Dottorando:
Hafiza Maria Maqsood
H. Maria Maqsood.

Tutore
Prof. Andrea Bondavalli
Andrea Bondavalli

Coordinatore
Prof. Matteo Focardi
Matteo Focardi

Anni 2018/2021

Declaration of Authorship

I, Hafiza Maria Maqsood, declare that this thesis titled, Agile Software Development Methodologies for Safety Critical Systems and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: H. Maria Maqsood.

Date: 27\04\2022

Acknowledgements

First, I thank my advisor Prof.Andrea Bondavalli, for being the kindest and most helpful guide I could have asked for. His door was always open, and the hours we spent discussing safety critical systems, agile (and everything else) are countless. Regardless how busy he was, he managed to carve out some time to answer my questions and hear my thoughts. I owe him all my academic results: it is not much, but I hope more will come in the future. I thank Eduardo Guerra for always taking out time to give his valuable suggestions on my work and for his precious teachings and insight. I would like to thank Fatima Mattiello for having invited me in INPE, Brazil for a month. She shared her work and treated me as an equal from the moment I entered the lab, although the difference in our understanding and knowledge was immense.

I thank Paolo Lollini, Andrea Ceccarelli, Tommasso Zoppi, Mohammad Gharib and Mirko Staderini for welcoming me in the Safety Critical Systems' family in Firenze. I thank all the people I've worked with during my PhD, as well as those that were kind enough to share their knowledge with me. Amongst them, Stefano Pietropaolli for his inspring lessons on cyber forensics, Paolo Lollini for amazing lessons on Quanitative Analysis of Systems, Andrea Ceccareli for giving me oppurtunity to program robots and Prof Andrea Bondavalli for teaching the concept of critical systems from scratch to the latest work being done in today's world.. . .

Dedicated to my Husband......

Abstract

The aim of my research is to discover possible application of agile for development of safety critical systems. Initially I have performed a detailed and methodological systematic literature review. It highlights the main hurdles for application of agile development methodologies for development of safety-critical systems. It also provides a comprehensive view of current state of literature regarding this topic. After successful completion of literature review I was able to list down major contradictions among agile approaches and safety-critical systems' traditional development approaches. Further I have also figured out possible directions of solutions.

Based on literature review I conclude that when security practices are included in any agile development process model they have reverberations on agility of that model to an extent where it cannot be called agile any more. To address this issue I have proposed a method to calculate the effect on agility of process model after inclusion of security practices. Secondly, I found through systematic literature review that Agile methods cannot be applied to development of safety-critical systems in their original form. Certain amendments or changes are required to find a middle ground where agile is adapted for such systems while respecting the safety standards. I worked further on this idea and proposed an approach which is hybrid model based on agile principles and safety critical systems' development standards. I have worked on different stages of software development life cycle and proposed an approach for all phases of Requirements, phases of Testing and Communication strategies among teams.

Contents

List of Figures

List of Tables

List of Abbreviations

SCS	**S**afety **C**ritical **S**ystems
XP	**X**treme **P**rograming
SCRUM	**S**ystematic **C**ustomer **R**esolution **U**nraveling **M**eeting
DSDM	**D**ynamic **S**ystem **D**evelopment **M**ethod

Chapter 1

Introduction

1.1 Background

Safety-critical systems are defined as those systems whose failure can cause harm (Avizienis et al., 2004). The system is considered safety-critical if its failure can lead to unacceptable circumstances such as loss of human lives or damage to the environment (Avizienis et al., 2004). Development of these systems in an agile way can be very beneficial in terms of time and cost. The basic principles of agile say that there should be rapid development, strong communication among all stake holders and changes should be welcome at any stage of development (Beck et al., 2001a). As there is a lot of focus on people so every individual in team should be motivated and must be given suitable environment and support to perform their jobs (Stavru, 2014). According to Jacobson (Jacobson, 2002), an agile team is very responsive to changes since adapting to change is what agile software development is all about. It is very important for an agile team to understand that software is developed by teamwork, and collaboration is the heart of success (Boström et al., 2006). Software engineers gathered forces and began to classify agile processes in early 2001 (Beck et al., 2001a). Agile Alliance stated the agile manifesto as (Beck et al., 2001a) “Individuals and interactions over processes and tools, Working software over comprehensive documentation, Customer collaboration over contract negotiation; Responding to change over following a plan” The agile manifesto states that it gives priority to working software over detailed documentation . It seems there is a conflict between heavy documentation and agile principles. (Cohen, Lindvall, and Costa, 2003) argues that there are issues that must be looked upon in written communication and we should not abandon documentation, instead we should use the documentation at and for appropriate points, especially for the development of safety-critical systems . The critical nature of safety-critical systems requires that if not all , maximum risks are handled during development of these systems. However, it is not enough to just perform risk analysis, certain safety standards must be incorporated during development of these systems, for this reason they must have testing strategy in place (Zimmermann et al., 2009). There is a lot of room for improvement of this mechanism to fit perfectly to the needs of safety-critical systems especially in agile perspective (Tracey, 2000). In short we can say that

- Agile stresses on cross-functional autonomous teams (Chen et al., 2015) whereas, safety-critical systems need specific experts doing specific tasks.
- There is also a gap to be filled about how organizations should arrange teams to achieve the right level of autonomy in any particular scenario, in our case for safety-critical systems (Hoda, Noble, and Marshall, 2012b).
- It is a challenging task to create cross-functional teams and keep the size of the team small (10-15) persons on team, which is a standard agile team size (Stray, Moe, and Hoda, 2018).
- (Holcombe, Ipate, and Grondoudis, 1995) states that formal verification methods and test driven development can be used together and there can be many enhanced benefits of this approach. This combination of formal verification methods with test driven development can ensure safety of systems along with reliability. (Laplante and DeFranco, 2017) argue that safety-critical systems have to comply to certain safety standards, currently there are not many techniques of testing that can fit into this scenario.

In 2004, a study commissioned by the U.S. Congress generated a list of “critical infrastructure” systems. In addition, a subsequent study conducted by the U.S. Department of Homeland Security (DHS) identified a set of “16 critical infrastructure sectors whose assets, systems, and net works,whether physical or virtual, are considered so vital to the United States that their incapacitation or destruction would have a debilitating effect on security, national economic security, national public health or safety, or any combination thereof.”

Agile has been used for development of software for years now. It has been more extensively used after the manifesto of agile was formalized (Beck et al., 2001a). However, there are still many concerns that need to be addressed for using agile in safety-critical systems. Agile manifesto was presented in 2001 (Beck et al., 2001a), and with that began the formalization of agile methodologies. It has also widened the use of agile methods in many different fields. Still there is a lot to be established for incorporating agile in development of safety-critical systems.

1.2 Research Statement/Problem

Agile methodologies are widely adapted for software development processes. However, for safety-critical systems there is still need of research and experimentation before they can be applied to the development phases in a more efficient way. The purpose of this research is to highlight the major areas of concern or points of conflict between agile and safety-critical systems. Along with that I try to address few of major problems.

First, I calculate the effect on agility of process models after including security practices. Along with that I propose solutions for possible applications of agile process models for development of safety-critical systems by combining traditional approaches and agile approaches for system development.

1.2.1 Methodology

Firstly, I address the issue of finding problems or hurdles in adaption of agile for safety-critical systems by performing a detailed analysis of literature. For this purpose I have used the methodology of Systematic Literature Review. The approach is formalized by (Kitchenham et al., 2009). First I gather the most relevant literature, then I perform the qualitative and quantitative analysis of literature to further enhance the quality of selected studies. Then I extract the relevant information from selected literature. After performing these steps of literature review I found list of major conflicts between agile methodologies and safety-critical systems' development approaches.Details are mentioned in Chapter 2

Further, I address the major problems found through systematic literature review. The major and core issue is that a process model losses its values of agility when security practices are included in it. These security practices are integral part of development for safety-critical systems. They are imposed by standards followed for development of these systems. I propose a method to quantitatively measure this effect and decide how much compromise on agility of process model is acceptable for a team looking forward to use agile for development of safety-critical systems. Details are mentioned in Chapter 3.

Another major problem elaborated in literature is a need of concretely defined processes on how to use agile for development of safety-critical systems in terms of life cycle. At what stages of development life cycle it is possible to use agile approaches and how it will be performed. This is a huge challenge, since Agile methods cannot be adapted in their original form for development of safety-critical systems. However, hybrid models can lead to successful combinations. My next contribution is along this line of work. I elaborate a method to include agile approach at different stages of software development life cycle, In particular, at requirements stage, testing stage and for communication among teams. Details are given in Chapter 4.

Chapter 2

Systematic Literature Review

2.1 Introduction

Agile has been used for development of software for years now. It has been more extensively used after the manifesto of agile was formalized (Beck et al., 2001a). However, there are still many concerns that need to be addressed for using agile in safety-critical systems. Agile manifesto was presented in 2001 (Beck et al., 2001a), and with that began the formalization of agile methodologies. It has also widened the use of agile methods in many different fields. Still there is a lot to be established for incorporating agile in development of safety-critical systems. This is the core purpose of performing this literature review, to find out gaps between agile and safety-critical systems and also to find out possible collaborations between the two. I have performed this literature review by following the guidelines of Kitchenham (Kitchenham, Charters, et al., 2007). The literature is very sporadic and for some fields (avionics, nuclear plants etc.) there is not much available. However, for understanding challenges and problems in general between agile and safety-critical systems Heeager and Nielsen (Heeager and Nielsen, 2018) have discussed areas of major concerns. There are other papers that discuss application of agile in different fields. Medical and pharmaceutical are the most discussed ones (Mc Hugh et al., 2013). However, there is no major evidence to specify problems particular to each field. Also there is not much discussed in terms of which agile methods are more used and what are the reasons. Some papers argue to use hybrid approach like (Gallina, Muram, and Ardila, 2018).

(Kitchenham, Charters, et al., 2007) has shaped the worked of literature reviews. In her point of view, Systematic literature reviews in all disciplines allow us to stand on the shoulders of giants and in computing, allow us to get off each others' feet. The advantages of systematic literature reviews are that:

- The well-defined methodology/review protocol makes it less likely that the results of the literature are biased, although it does not protect against publication bias in the primary studies.
- They can provide information about the effects of some phenomenon across a wide range of settings and empirical methods. If studies give consistent results, systematic reviews provide evidence that the phenomenon is robust and transferable. If the studies give inconsistent results, sources of variation can be studied.
- In the case of quantitative studies, it is possible to combine data using meta-analytic techniques. This increases the likelihood of detecting real effects that individual smaller studies are unable to detect. The major disadvantage of systematic literature reviews is that they require considerably more effort than traditional literature reviews. In addition, increased power for meta-analysis can also be a disadvantage, since it is possible to detect small biases as well as true effects (Kitchenham, Charters, et al., 2007).

(Kitchenham, Charters, et al., 2007) Some of the features that differentiate a systematic review from a conventional expert literature review are:

- Systematic reviews start by defining a review protocol that specifies the research question being addressed and the methods that will be used to perform the review.
- Systematic reviews are based on a defined search strategy that aims to detect as much of the relevant literature as possible.
- Systematic reviews document their search strategy so that readers can assess their rigour and the completeness and repeatability of the process (bearing in mind that searches of digital libraries are almost impossible to replicate).
- Systematic reviews require explicit inclusion and exclusion criteria to assess each potential primary study.
- Systematic reviews specify the information to be obtained from each primary study including quality criteria by which to evaluate each primary study.

- A systematic review is a prerequisite for quantitative meta-analysis (Kitchenham, Charters, et al., 2007).

2.2 Review Protocol

I have carefully established review protocol before starting systematic literature review. This review protocol is warily chosen, approved and thoroughly discussed with two other researchers working in same field. One of them is my supervisor for this work.I have chosen to follow the guidelines by (Kitchenham et al., 2009). In the light of her guidelines, I established a review protocol that will be followed further to perform this literature review. In this phase I decided in detail about the steps and protocols that will be followed during the process, these are given below

1. Develop research questions.
2. Define inclusion/exclusion criteria.
3. Locate studies from following sources

i Google Scholar

ii ACM

iii IEEE

iv SCOPUS

v Web of Science

4. Perform query refinement
5. Select studies on the basis of inclusion/exclusion criteria.
6. Assess quality of study by peer review of selected studies using quality matrix based on rigor and relevance.
7. Extract data from selected studies, using the approach of "coding studies", with the help of tool 'R'.
8. Analyze and present results also provide interpretation of results.

2.3 Methodology

I have applied the methodology in accordance with review protocol already established. I have established research questions to address these issues. In broader term empirically researched questions have the primary goal of identifying concerns about agile in different fields. I also try to identify the most used agile approaches and aspects that make them more suitable as compared to others. Then, I further investigate the prospects of hybrid approach. I have performed the review systematically, by following concrete steps (Kitchenham et al., 2009). I have later analyzed the data by using a data analysis tool called "R" (Agbo et al., 2021). studies have been analyzed for their quality before the analysis of results they present. I have used the systematic mapping approach (Kitchenham et al., 2002) and interpretive literature reviews approach.

There are many ways of performing a literature review as explained by (Templier and Paré, 2015), (Cooper, 1988), (Dybå, Kitchenham, and Jørgensen, 2005), (Petersen, Vakkalanka, and Kuzniarz, 2015). It is also common to combine multiple approaches. I have used the approach given by (Wolfswinkel, Furtmueller, and Wilderom, 2013) along with steps and guidelines of (Kitchenham, Charters, et al., 2007). Both have defined steps in quite similar way, I have used certain aspects from both approaches. (Templier and Paré, 2015) suggested that there are 4 types of literature reviews possible. Namely cumulative, narrative, developmental and aggregative. Cumulative reviews present extensive current knowledge and also draw an empirical conclusion on the base of studies knowledge. Narrative reviews present and summaries the currently available literature about any topic. Developmental review presents new concepts based on previous knowledge. Aggregative review presents the tests that verify currently present hypothesis.

I have performed a systematic literature review (Kitchenham et al., 2009) based on grounded theory (Wolfswinkel, Furtmueller, and Wilderom, 2013). Firstly, I have defined research questions, secondly I adapted a inclusion/exclusion criteria to select or exclude studies, thirdly I used multiple resources to find relevant studies, fourthly I performed query refinement, fifth step was to select studies, sixth was to assess quality of selected studies and last but not the least is coding of selected studies. In the proceedings sections I provide details of each step.

2.3.1 Research Question

I started by formalizing research questions, so that I can review the selected literature with clear notion of "what to look for" in mind. I formulated following questions.

- Which agile process model/models are most used or discussed for developing safety-critical systems?
- Which phases of SDLC are most discussed for adopting agile for development of safety-critical systems?
- Which domains of safety systems report highest adaption of agile for their systems?

To answer first question I looked into literature with this specific notion in mind. I found that the answer varies slightly from one field to another. For example there can be some different set of problems when agile is applied in medical domain (McHugh, McCaffery, and Casey, 2014), (Hajou, Batenburg, and Jansen, 2015) Or when it is applied in automotive domain (Roy et al., 2018). However, mostly the problems are common.

I observed that there is a huge trend of using Scrum and in some cases XP for development of safety-critical systems (Mc Hugh et al., 2013), (Carpenter and Dagnino, 2014), (Taliga, 2017), (Grenning, 2001b), (Rasmussen et al., 2009). I try to figure out aspects that make these process models being used more widely as compared to other agile process models.

Another strong observation was that there is huge trend of hybrid software development life cycle (Gallina, Muram, and Ardila, 2018), (Mc Hugh et al., 2013), (Axelsson et al., 2016), (Carpenter and Dagnino, 2014), where some parts of agile are integrated into other development methods. I tried to find out the prospects of using agile alone or in hybrid mode for development of safety-critical systems.

Hence I have set of empirically derived questions that I try to answer through this literature review.

2.3.2 Inclusion/Exclusion Criteria

I have chosen the following criteria for including studies in review.

Inclusion Criteria

- Primary studies and secondary studies.
- Studies that relate to agile and safety-critical systems.
- Studies that discuss any particular agile process model with safety-critical system.
- Studies that discuss issues between agile and safety.
- Studies that discuss safety and agile in broad spectrum.

The studies that fall under the following criteria were not included in this review.

Exclusion criteria

- Studies written in languages other than English.
- Short papers or posters having length of less than three pages.
- Studies that are unavailable at mentioned platforms.
- Studies that focus on safety-critical systems without discussing process models or methodology for development.

2.3.3 Locate Studies

I have used combination of search engine and different repositories to locate the relevant literature. In particular I have used following resources

- Google scholar
- ACM - Association for Computing Machinery
- IEEE - Institute of Electrical and Electronics Engineers
- Scopus
- Web of Science

2.3.4 Query Refinement

I started with basic keywords of "agile" and "safety-critical systems". However, soon I realized that I must form a proper string to include all relevant literature. After doing multiple iterations by including different keywords and using clause of "OR", "AND" I were able to comprehensively formalize the following query string.

(Agile OR agility OR Scrum OR XP OR Feature driven development OR ASD OR Crystal OR DSDM) AND (safety software OR safety-critical software OR safety-critical systems OR regulated software)

2.3.5 Select Studies

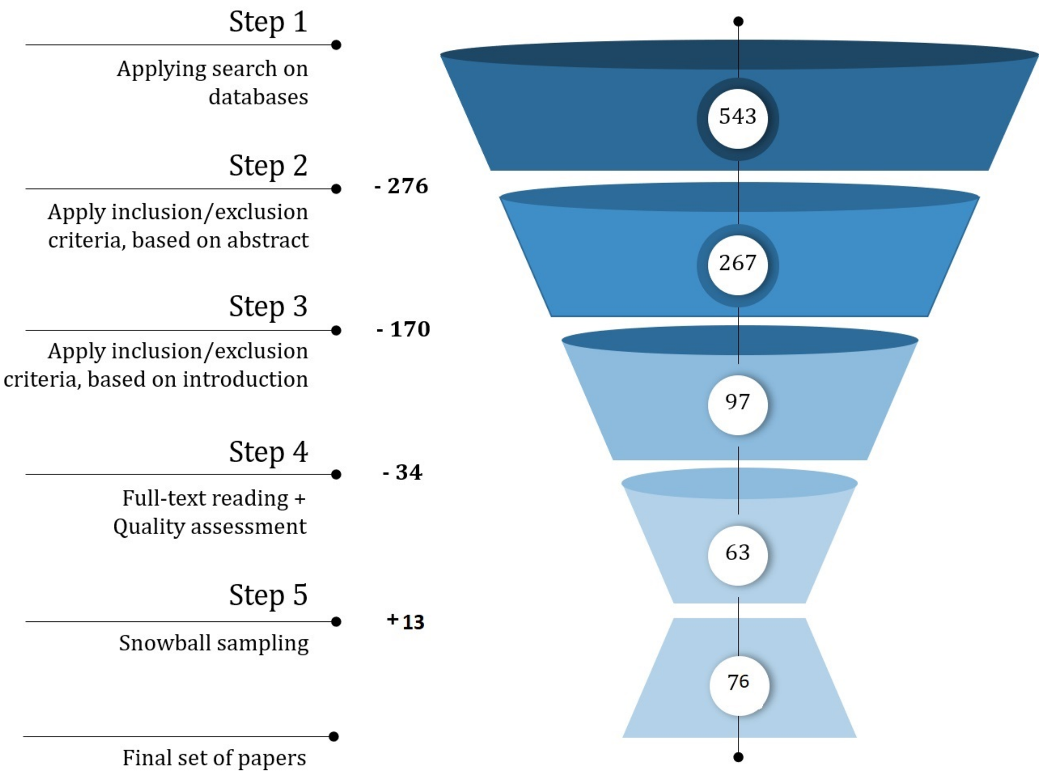

Step 1
Applying search on databases
Step 2
Apply inclusion/exclusion criteria, based on abstract
Step 3
Apply inclusion/exclusion criteria, based on introduction
Step 4
Full-text reading + Quality assessment
Step 5
Snowball sampling
Final set of papers
- 276
- 170
- 34
+13
543
267
97
63
76

FIGURE 2.1: Paper selection process.

It is to be noted that key word Scrum caters to all studies with discussion on methods derived on the basis of Scrum. More particularly, studies that mention R-Scrum, SafeScrum more suitable for safety-critical system, and LeSS and scaled agile framework (SAFe) are all included by keyword Scrum. When I started search after formalizing the query it still yielded 1000s of results. I further analyzed these results to select the most relevant ones with substantial information. In first iteration, just by looking at titles I found following number of papers to be relevant

- ACM [82 studies]
- IEEE [98]
- Scopus[27]
- Web of Science[47]

- Google Scholar [289]

I performed second round of selection on these 543 papers. I read abstracts of all these studies and shortlisted them to 267 papers, based on pre-defined inclusion/exclusion criteria. In third round, I started reading introductions of papers and it further reduced the selected number of papers to 97. As only 97 were in accordance with inclusion criteria. In fourth step, I read full text of selected papers and also applied two rubric for quality assessment. The two rubrics used for quality assessment were rigor and relevance. Further details on quality assessment are given in section 2.2.6. After completing this stage I had 63 papers. In last step, I performed one execution of backward snowballing on these papers and that made me add 13 studies to selected data. Snowballing is a technique where you look into the references of your selected studies. After going through the list of referenced papers I found 9 more papers that were relevant to my interest. Hence, at the end of this selection process I had total of 76 papers. I further performed analysis on this selected literature.

2.3.6 Assess study quality

There are different methods available to assess the quality of studies. (Bandara et al., 2015) explains the steps of performing a literature review and also includes the quality assessment of studies. Bandara et. al (Bandara et al., 2015) gives a list of questions that should be answered for all papers to examine the quality of papers.

(Sabir et al., 2019) Sabir et al. performed SLR and used the quality assessment techniques given by (Er, 2005) PICOC AND (Brereton et al., 2007) Brereton et al. They formed a checklist of questions and based their quality criteria on answers to those questions. Their checklist can be divided into four parts.Firstly they ask questions about criteria to design the study, secondly they have points about method used for setup of study. Third are questions about how they have performed the study and fourth and final are points about how they have come up with conclusions from the selected studies. (Anwar and Pfahl, 2017) Anwar et al. used a table of questions to assess quality of studies. They have assigned numbers to each question and further performed analysis on the basis of these numbers. The higher the numbering score, higher the quality of paper.

TABLE 2.1: Rubric - Rigor Calculation (Ivarsson and Gorschek, 2011)

Description of Context (C)	Study Design (S)	Validity discussed (V)
Strong (1)	Strong (1)	Strong (1)
Medium (0,5)	Medium (0,5)	Medium (0,5)
Weak (0)	Weak (0)	Weak (0)

I carefully looked into above mentioned various methods for performing quality analysis on selected set of studies. I have used well-defined inclusion/exclusion criteria from very start of the process. After selecting 76 papers, I further applied two metrics for calculating rigor and relevance of papers. These metrics are proposed by Martin Ivarsson and Tony Gorschek (Ivarsson and Gorschek, 2011).

Two authors have separately applied these metrics on selected studies. To study the co-relation between results of both authors I calculated Kendell's coefficient of concordance (Field, 2014), which served as an extra check to ensure the results of metrics are reliable.

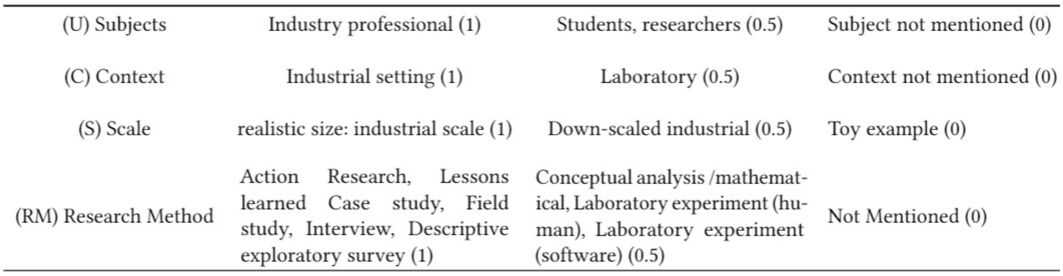
Rigor AND Relevance

First matrix given in Table 2.1 evaluates the studies for rigor. This is a scoring rubric for evaluating rigor given by Mark and Gory (Ivarsson and Gorschek, 2011). They have proposed three aspects to measure rigor. First is description of context, user should be able to understand the context of study. Second aspect is design of study. The parameters used to design the study should be well explained. For examples variables used, sampling or selection criterion etc should be clearly defined. Third aspect is validity of results. There must be comprehensive discussion about the validity of results, limitations and threats to validity etc. They further assign scores to each aspect, if the aspect is well-defined it is given a score of 1, if it is at medium level then score of 0.5 is given and for weak description 0 is given. For strong description an aspect should be clearly defined and with all required parameters in a way that is understandable by user. If an aspect has some relevant parameters but in a vague way, that is not understandable by reader then it is at medium level. If there is no discussion of an aspect then it is considered weak. This rubric is shown in Table 2.1.

Two authors have assigned scores to all studies separately and each of them calculated one final value of rigor for each paper.

I have also calculated score of relevance for all papers. In same way as I did for rigor, two authors have filled the matrix for relevance separately. Both authors calculated single final value of relevance for all papers. Papers were further scrutinized on the basis of these values. As shown in Table 2.2. Martin Ivarsson and Tony Gorschek (Ivarsson and Gorschek, 2011) have proposed matrix for evaluating relevance on the basis of four aspects. First is subjects used in evaluation in a study. If they are relevant users then value of 1 is assigned otherwise zero is assigned. Second aspect is context, if a study represents an evaluation in context of problem in industrial scenario then value of 1 is assigned otherwise in case of laboratory experiments or no context, a value of zero is assigned. Third aspect is scale, if an industrial level evaluation is performed than a value of 1 is assigned otherwise in case of down scaling or dummy example a value of zero is given.Last aspect is research method used by the study, in case of action research, interviews or any other method that has interaction with industrial setting, a value of 1 is assigned, otherwise for closed, laboratory experiments zero is assigned.

TABLE 2.2: Scoring rubric for evaluating relevance based on Ivarsson and Gorschek (Ivarsson and Gorschek, 2011)



(U) Subjects	Industry professional (1)	Students, researchers (0.5)	Subject not mentioned (0)
(C) Context	Industrial setting (1)	Laboratory (0.5)	Context not mentioned (0)
(S) Scale	realistic size: industrial scale (1)	Down-scaled industrial (0.5)	Toy example (0)
(RM) Research Method	Action Research, Lessons learned Case study, Field study, Interview, Descriptive exploratory survey (1)	Conceptual analysis /mathematical, Laboratory experiment (human), Laboratory experiment (software) (0.5)	Not Mentioned (0)

Quantification of rigor and relevance is performed by summing up individual values assigned to aspects as shown in Table 2.4. For example I have a study with code name P11. The final values of Rigor and Relevance for P11 are

P11, Rigor = C+S+V (Table2.4)
=1+1+0.5 (Table2.1)
Rigor = 2.5

P11, Relevance = U+C+S+RM (Table2.4)
=0+1+1+1 (Table2.2)
Relevance = 3

This calculation shows that P11 has rigor score of 2.5 out of 3 and relevance score of 3 out of 4. Hence it has good values of rigor and relevance and will be

TABLE 2.3: Quantification of Rigor and Relevance

Rigor= C+S+V	Context (C)	Study Design (S)	Validity (V)	Discussed
Relevance= U+C+S+RM				
Subjects,Users (U)	Context (C)	Scale (S)	Research (RM)	Method

included in selected set of studies. I excluded the studies with accumulative score of 1 or less from set of selected studies.

Coefficient of Concordance

To further assess the quality of these attained values, I calculated Kendell's coefficient of concordance (Kendall, 1948) (Legendre, 2005). Kendall's coefficient of concordance commonly known as Kendell's W is a non-parametric statistical approach. It is normalized form of the Friedman test (Marozzi, 2014). It is used for assessing agreement among raters. Kendall's W ranges from 0 (no agreement) to 1 (complete agreement). Kendall's coefficient of concordance (W) gives the degree of association of ordinal assessments made by multiple assessors on same data (Fager, 1957). I have used following formula to calculate Kendell's coefficient of concordance

$$W = 12 * S(m^2) * ((n^3 - n) - m * T))$$

Where S is the sum of squared deviations(in my case sum of squared deviations of final values of rigor OR relevance), m is the number of raters, n is the total number of objects(in my scenario, number of selected studies) and T represents tied ranks. For initial calculation I do not consider the property of tied ranks hence marking T as zero. I selected 10 random studies from selected list of studies. These 10 studies have been assigned certain values of rigor and relevance by two authors. After applying the above mentioned formula of Kendell's coefficient I got following results.

For Rigor

$$W = 0.92$$

For Relevance

$$W = 0.76$$

My set of data had tied ranks. To further clear the data calculation and obtain cleaner results I applied the same formula of Kendell's Coefficient of Concordance (Kendall, 1948) by considering the tied ranks T in data (Gearhart et al., 2013). Updated values of 'W' were

For Rigor

$$W = 0.97$$

For Relevance

$$W = 0.98$$

As 1 is considered the complete agreement and 0 represents no agreement between raters, these values represent that there is great consensus among assessors about quality of selected studies. This gives an empirical evidence for the quality of studies in terms of their relevance and rigor.

2.3.7 Coding

There are plenty of ways to organize data for analysis. There are many tools available to make the process easy and efficient (Mohammadi and Prasanna, 2003). The choice of tool can vary according to the type of data, field or the kind of analysis one wants to perform on data (Ali and Bhaskar, 2016) (Bandara et al., 2015). I have used 'R' tool for coding of data (Chambers, 2008). It is widely used by many researchers. It provides very efficient support for analysis of data. I have used it in particular for coding of data. I have used aspects of grounded theory (Wolfswinkel, Furtmueller, and Wilderom, 2013) technique. I have followed its steps of coding specifically. Coding the data refers to assigning keywords to data that best represent that paper. I have followed three steps of assigning codes to data.Open codes, Axial codes and Selective codes 2.2.

Open codes refer to the bird's eye view of data. I have read papers time and again to get the basic idea of each paper clearly. Then I created one line summary of each paper, writing the essence of paper with major keywords.

Open, selective and axial coding can be done in back and forth manner. This is the strategy I have used to ensure that I extract all required information from studies. After performing the step of open coding I moved to axial coding. Here, the idea is to read what you have extracted till now and narrow it down to more concrete keywords. I read the extracted summaries and extracted keywords from summaries. At this stage I had a relatively huge set of keywords extracted from all selected studies.

The last step in coding is to inter-relate the codes and find common grounds. All the steps of coding are done by moving back and forth multiple times. I established some high-level categories and linked all codes to these categories. This leads me to interesting findings regarding different aspects of agile and safety-critical systems.

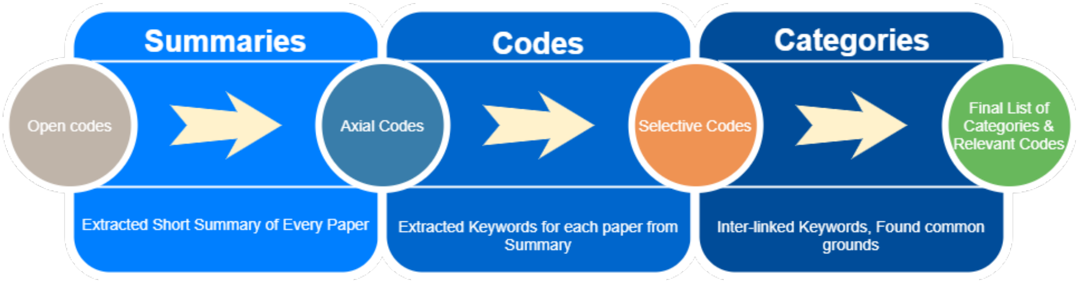

Summaries
Codes
Categories
Open codes
Axial Codes
Selective Codes
Final List of Categories & Relevant Codes
Extracted Short Summary of Every Paper
Extracted Keywords for each paper from Summary
Inter-linked Keywords, Found common grounds

FIGURE 2.2: Steps of Coding

After completing the stage of coding I was able to establish results in terms of 3-tier categories and relevant codes. At highest level I have three groups namely SDLC - Software Development Life cycle Phases, Application Domains and Process Models. At second tier I have Requirements, Testing, IV 'I&' V and Traceability under SDLC, Medical, Automotive, Rail, Avionics, Nuclear Plants under Application Domains and finally Scrum, XP, Hybrid model, DSDN and ASD under Process Models. It is represented in Figure 2.3.

All the codes which I have extracted from selected studies belong to the category Or categories mentioned in figure 2.3. List of Codes along with their referenced study is shown in Fig 2.4

I have carefully selected and performed each step of chosen methodology.. At this point I had comprehensive and well sorted data with the help of tool 'R' for further analysis.

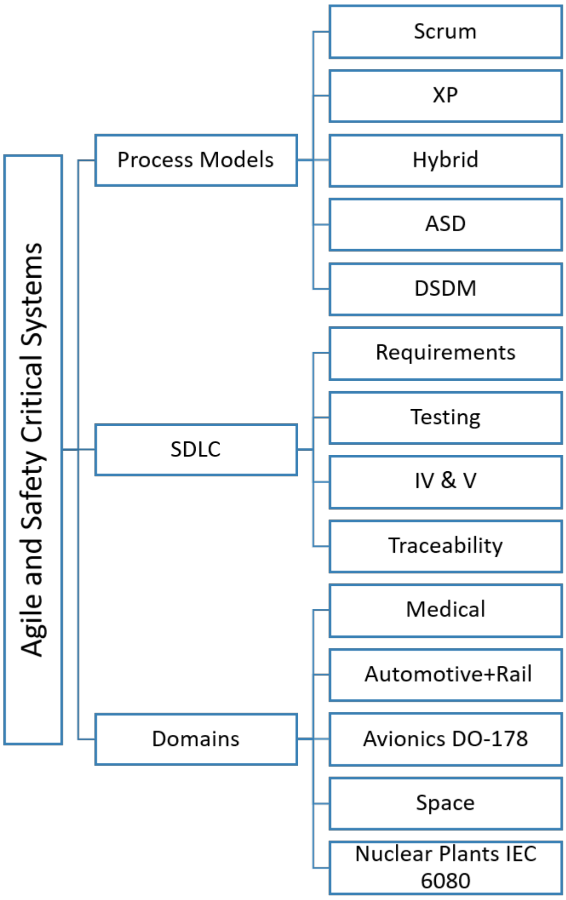


FIGURE 2.3: Categories

2.4 Analysis and Results

In this section, I present the overview of selected studies along with associated details. I have selected 76 studies that make final set of data as shown in figure 2.1. Further analysis of studies is presented in 2.5 The data is represented in terms of time, type of publication and the kind of research methodology adopted by papers.

It is visible that peak was raised in 2013,2017 and most recently also. This shows that this is an active field of research at present. Figure 2.5 also shows the type of publication of studies. For example most of the studies were presented in conferences, secondly in journals and then in workshops. The most adopted research methodology in selected studies was example applications. From 2001 to 2013 the research is more focused on experiments and case studies, where researchers focus to learn and see by applying agile to safety-critical systems. However, from 2014 on wards the research is more focused on lessons learned, challenges for application of agile in safety-critical systems and possible approaches. Specifically, hybrid approach is adopted

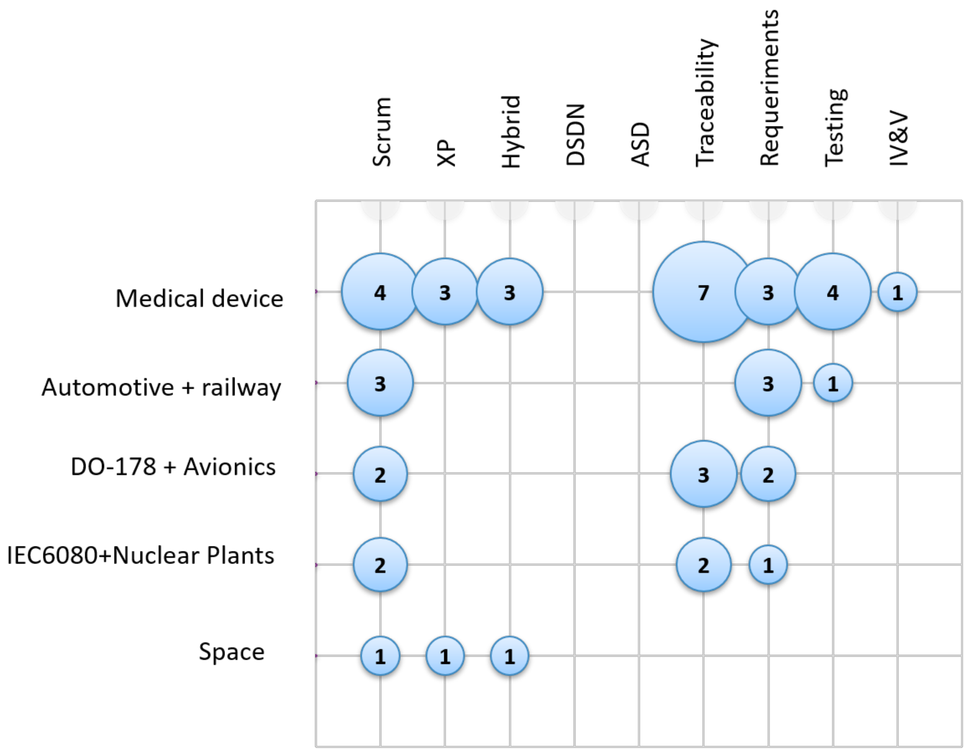


FIGURE 2.4: codes and studies

with reports of many benefits. It is combination of traditional approaches and agile. There are many reported benefits of the approach.

There are different research methodologies applied in set of selected studies. An overview of type of research methods presented by papers is shown in Figure 2.5. There are various techniques of categorising research methods (Wohlin, Höst, and Henningsson, 2003). Depending on the research methodology used the reliability of findings can vary. Generally, researchers give more value to the research performed in industrial setting for example case studies or example applications (Petersen and Gencel, 2013). There are other categories of research methods defined by (Easterbrook et al., 2008) (Sousa Santos et al., 2017) such as action research, field study,survey,experience report and literature review. I have selected categories most fitted to my selected studies and it includes all major research methods.

In appendix A I have presented the selected studies with following characteristics: distribution over time, venue of publication, type of study and applied research method. Studies have discussed different aspects of agile for safety-critical systems.

In early 2000, studies were more focused on case studies and experiments

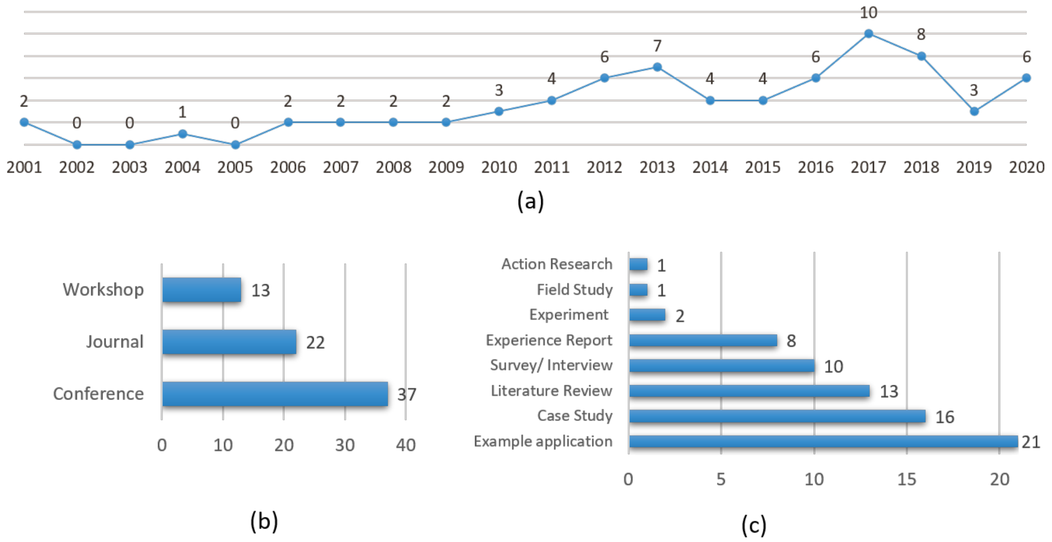


FIGURE 2.5: Types of studies Selected

while more recently, there is more focus on challenges of using agile in different safety domains and possible solutions for application of agile in development of safety-critical systems. Various topics are discussed in literature regarding multiple aspects of using agile for development of safety-critical systems. Here I present results from selected studies.

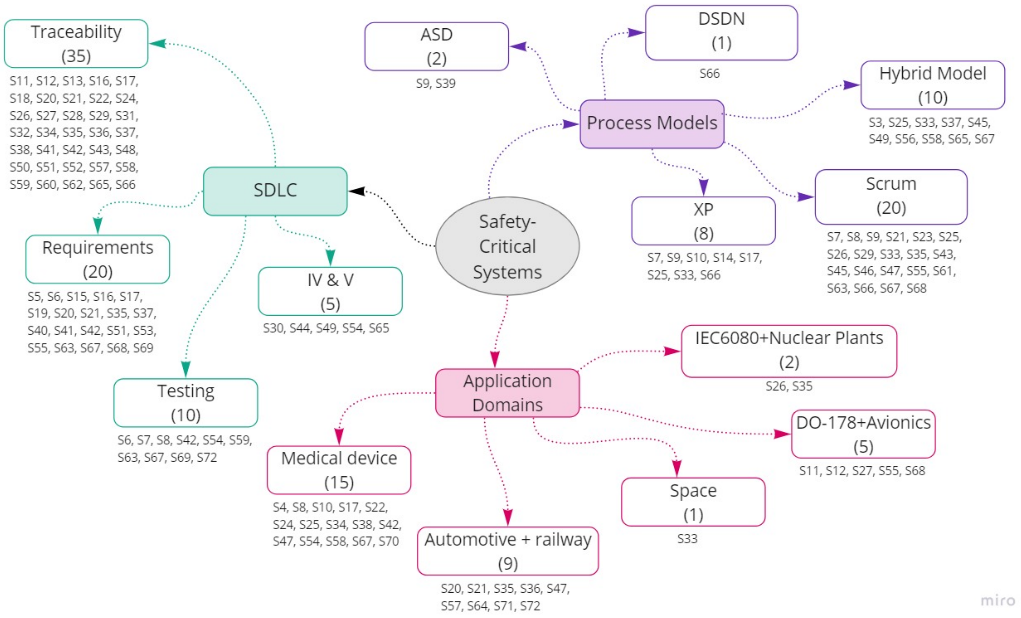

Traceability
(35)
S11, S12, S13, S16, S17, S18, S20, S21, S22, S24, S26, S27, S28, S29, S31, S32, S34, S35, S36, S37, S38, S41, S42, S43, S48, S50, S51, S52, S57, S58, S59, S60, S62, S65, S66
ASD
(2)
S9, S39
DSDN
(1)
S66
Hybrid Model
(10)
S3, S25, S33, S37, S45, S49, S56, S58, S65, S67
Process Models
SDLC
Safety-Critical Systems
XP
(8)
S7, S9, S10, S14, S17, S25, S33, S66
Scrum
(20)
S7, S8, S9, S21, S23, S25, S26, S29, S33, S35, S43, S45, S46, S47, S55, S61, S63, S66, S67, S68
Requirements
(20)
S5, S6, S15, S16, S17, S19, S20, S21, S35, S37, S40, S41, S42, S51, S53, S55, S63, S67, S68, S69
IV & V
(5)
S30, S44, S49, S54, S65
Testing
(10)
S6, S7, S8, S42, S54, S59, S63, S67, S69, S72
Application Domains
IEC6080+Nuclear Plants
(2)
S26, S35
DO-178+Avionics
(5)
S11, S12, S27, S55, S68
Medical device
(15)
S4, S8, S10, S17, S22, S24, S25, S34, S38, S42, S47, S54, S58, S67, S70
Space
(1)
S33
Automotive + railway
(9)
S20, S21, S35, S36, S47, S57, S64, S71, S72
miro

FIGURE 2.6: Results of studies

I have focused on all selected studies in particular to find answers to my research questions, these categories are discussed in next sections in detail.

2.4.1 Process Models

In this section I present my analysis of studies in reference to my first research Question stated as "Which agile process model/models are most used or discussed for developing safety-critical systems?" in section 2.2.1.

I found that SCRUM and XP are most discussed process models in literature (Taliga, 2017). There are multiple studies which discuss Scrum and claim that it is one of the best and most suited agile model for development of safety-critical systems. (Özcan-Top and McCaffery, 2018) say that they have applied Scrum and XP for Medical device development and found that Scrum was partially or fully covering five aspects of process model which are Project Planning; system Requirements Analysis and Software Requirements Analysis:Stakeholder Requirements Definition and Project Assessment and Control. (Özcan-Top and McCaffery, 2018) have looked into XP for development of medical device software and propose that XP provides only partial coverage of project planning, software requirements analysis,project assessment and control, software unit implementation and verification, software release and software problem resolution, software integration and integration testing. Further they present an interesting approach by combining two most used models, i.e XP and Scrum and found that still some additional practices were needed to develop medical device software (Özcan-Top and McCaffery, 2019). They further extended their study and combined DSDM with XP and Scrum, the results were better that combining two process models. With three process models combined, the resulting process provided better coverage and support for development of medical device software. This is a compelling work and can lead to better customized agile model for safety-critical systems.

Another interesting approach is shared by (Cordeiro et al., 2007) they state that they find two aspects of agile process models very intriguing namely incremental approach, adaptive planning and flexibility. To incorporate these aspects they combined Xp and Scrum and proposed agile patterns that can be termed useful for development of safety-critical systems.

Several studies have discussed only Scrum in perspective of its usage for safety-critical systems. It has been widely used on its own or there is a new version called safe scrum that has some modifications to better suit the needs of SCS. Scrum has shown some promising results in the domain of SCS. Most studies report that customized version of Scrum is more suitable for safety systems than the model in its pure form with. Studies also argue that there are certain aspects that need to be tailored in SCRUM or need modification. These are documentation, planning, proof of conformance, requirements specifically safety requirements and evaluation, mitigation of risks or simply put "risk analysis", verification and validation and last but most important is traceability. These ideas are very concretely explained by (Doss and Kelly, 2016). They discuss four basic underlying principals for all safety standards. They say that although standards differentiate from each other in their details but there are 4+1 principals that stay true. And based on these principals Scrum can be integrated in to development of safety-critical systems. First three principals are about safety requirements. Safety requirements should be inline with safety standards, they must be maintained throughout process and there must be criteria to satisfy these requirements. Fourth principal says that "hazardous behaviour" of software should be mitigated. Last principal is about balance between following these principals and software system.

After Scrum, XP has emerged as most used agile process model. Mostly studies have reported that there are key differences between the two and XP in its original form cannot be used for development of safety-critical Systems. By looking at each phase of development life cycle starting from requirements,design, code and till testing there are significant differences specifically in requirements and testing.Studies propose that changes must be made in Xp and then extended model can be a solution.

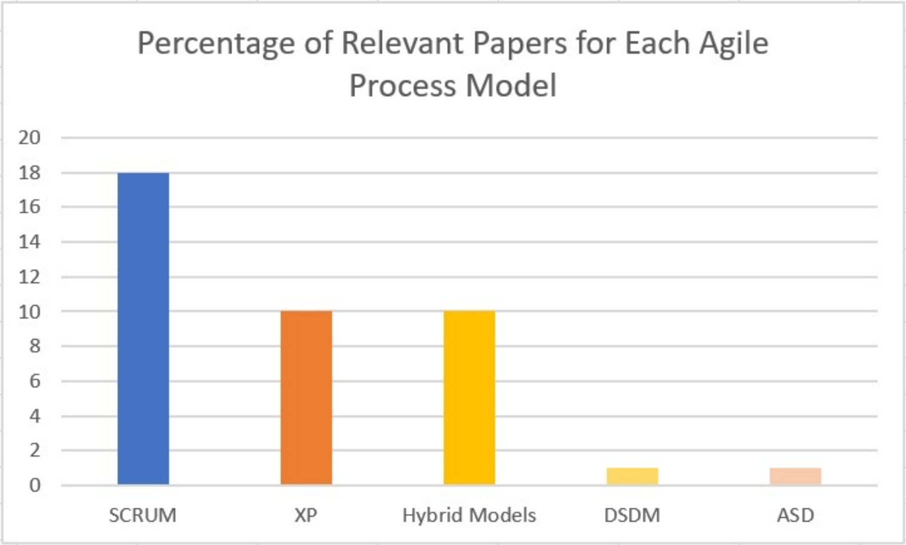


FIGURE 2.7: Process Models

Scrum and Xp have been used in combination too and provided good results. As agile it self is a way of doing things it does not impose strict steps, so combining the practices from two agile models is quite a possibility and it has shown success for development of safety-critical systems. Only one study has provided evidence of using ASD (Abdelaziz, El-Tahir, and Osman, 2015) and one has discussed DSDM (Özcan-Top and McCaffery, 2019). DSDM was adopted with combination of Scrum and XP. There is hardly any evidence of other agile process models being used, and there are no significant results or application presented for other models.

Another interesting aspect discussed in studies is human aspect. As agile puts a lot of focus on involving all stakeholders in the process as much as possible, hence there are certain facets that need more attention as compared to traditional approaches. People need to have more open acceptance towards adaption of new approaches, otherwise it becomes a huge hurdle to adapt to something as new as agile for safety-critical systems. (Grenning, 2001a), (Grenning, 2001b) have used Xp in industrial setting. They state that one of the parties is always reluctant to use new approach like XP and this mindset is biggest hurdle in the process.

More recent trend shown in studies is towards use of hybrid models. More specifically after 2010, there has been a huge trend of using hybrid process models In this approach researchers have combined traditional process model approach with agile methods. This approach has shown most promising results and is widely adapted. Still SCRUM or SAFE Scrum remains most popular one for development of SCS. Safe scrum is also a hybrid model, where people modify it a bit every time according to the project in hand. It takes basic concepts of Scrum and integrates traditional methods to form a process model that can develop SCS successfully with agile.

(Arthur and Dabney, 2017) have taken a deep dig into hybrid process models. They have discussed the approach in context of validation and verification. They have stated that validation and verification is still a daunting task even with hybrid approaches. They identified 30 practices of verification and validation and assessed them against hybrid approach.They further say that some methods of hybrid approach can be applied without any change, some need minimal modification whereas others cannot be applied and need replacement. (Dabney and Arthur, 2019) have further worked on same lines and described some approaches to overcome these hurdles specific to validation and verification with hybrid process models for SCS.

Hybrid models are mostly reported as combination of Scrum and traditional approaches. More popularly called "Safe Scrum". Studies report its wide usage and successful application in industry. Some studies report the alteration of traditional V-Model by combining it with agile and the new model is termed as AV model. (McHugh, McCaffery, and Casey, 2014) , (McHugh, McCaffery, and Coady, 2015) have presented the application of hybrid model called AV -model. They have documented that the approach was a huge success and medical device software was developed 7% faster as compared to when developed with traditional process model.

My analysis of process models reveals that there are certain aspects due to which one model is reported or used more as compared to others. Here I outline some key findings.

Scrum has appeared as most used model since it has following features.

- Covers all phases of SDLC. It has well defined phases and not just abstract approach to be followed in each phase.
- It is the only agile model with concrete steps.
- Concretely defined steps need very little modifications to comply to safety standards.
- Industrial proof of good performance is available, hence more people are relying on this approach.

XP is second most used model with following pros and cons

- Does not Cover all phases of SDLC. Talks about principals that should be followed to define phases.
- Presents ideas of steps only for certain phases of SDLC, does not give concrete steps.
- Since gives abstract ideas about phases and steps hence need major modifications to adopt to safety standards.
- Industrial proof of good performance is available, hence comparatively more studies have reported this approach.

Hybrid model is most discussed model in recent studies. I have found following aspects regarding hybrid model

TABLE 2.4: Analysis of Process Models

Process Model	SDLC-Phases	Activities/Steps of SDLC	Compliance to Safety Standards	Proof of Industrial Application
SCRUM	Fully covered	Well-Defined	Yes with small modifications	Yes
XP	Partially covered	Not defined	Needs Modifications	Yes
Hybrid	Fully covered	Well-Defined	Yes	Yes
ASD	Abstract	Abstract	Needs Modifications	No
DSDM	Abstract	Abstract	Needs Modifications	No

- Covers all phases of SDLC for SCS.Has well defined phases, mostly taken from traditional SDLC.
- Has well defined steps of each phase mostly taken from scrum.
- Adapts well to the needs of safety standards.
- Widely accepted in industry with proof of application.

In my point of view, the future is with hybrid agile models. They are most promising for development of SCS with agile. Still there is a lot of roam for research in this area but it can be deemed as most promising current topic.

2.4.2 Software Development Life Cycle

In this section I present my analysis of studies in reference to my second research question stated as"Which phases of SDLC are most discussed for adopting agile for development of safety-critical systems?" in section 2.2.1. The debate has been going on for several years as to use agile process models as they are or with modification. There are many proposed models with changes in basic agile models or with concepts of SCS integrated in them at different stages of SDLC.

I have found that agile SDLC models cannot be adapted without needed alterations in different phases to meet the needs of safety-critical systems. However after these amendments they have shown good results for SCS. The major areas of concern which need attention when one wants to adapt agile for SCS are traceability,requirements, testing and validation-verification techniques.

Eliciting and understanding requirements and having complete traceability of requirements is a daunting task. It becomes even more trivial with safety-critical systems. Safety requirements do not come from client only, they can come from assessors, safety standards and risk factors. These requirements must be dealt with great care hence making the whole process of requirement engineering one of the most discussed topics in my set of studies. Studies report that this phase involves all stakeholders and hence there are many aspects regarding people involved in the process. Acceptance of agile is still an issue where people be it development team, assessors or client are not ready for this change yet. Then there are issues of awareness about agile among stakeholders. (Alhubaishy and Benedicenti, 2017) have presented an approach to discuss the emotional contagion in agile teams. They propose that this is an important factor that is not given due importance till now. As agile has a lot of focus on people hence they try to analyze factors to increase positivity among people of teams. They argue that this will ultimately increase the ratio of successfully completed projects.

Another important conflicting point between traditional approaches and agile is formalization of requirements and traceability. Sometimes people confuse formalization of artifacts with agile. They argue that agile does not support formalization of any step or activity. This is not entirely false assumption as formalization adds time and cost and agile supports rather rapid development. However this can be amended if needed. As it is extremely important in safety-critical systems to formalize requirements and have traceability mechanism in place there is one compelling study by (Boström et al., 2006). They have presented two extensions to Scrum in the phase of user stories. They have proposed "Abuser stories (threat scenarios) and Security-related User stories (security functionalities)" for dealing with requirements. They tested the approach in a student project and conclude by saying that agile can have many benefits if integrated thoughtfully for development of safety-critical systems.

Heavy documentation required by safety standards for detailed requirements and traceability is in conflict with agile principal of less documentation. (Wang, Bogicevic, and Wagner, 2017) proposed an approach using Scrum for SCS where they categorized documentation in three ways namely safety story pattern, safety epic pattern and agile safety plan. Safety story pattern and safety epic pattern have shown promising results in finding a middle ground acceptable by agile and standards of SCS, where as agile safety plan needs further investigation.

Handling requirements in iterative manner (supported by agile) and adhering to safety standards is a tough task. Hence this area needs special attention in SDLC for using agile for SCS development.Heavy documentation is used by traditional approaches to document requirements in detail, since agile asks for as less as possible documentation hence making the task difficult. (Wang, Ramadani, and Wagner, 2017) have outlined key problem areas between agile and safety-critical systems regarding requirements and safety systems. They are safety requirements acceptance, communication, time frame for safety requirements and traceability.

When the system is large scale system the requirements phase becomes more even critical to handle. (Islam and Storer, 2020) have also presented a study about agile in avionics' systems which is a large scale system.They comment that there is need of tools for traceability of requirements that can work in an agile manner while keeping security measures intact. Traceability of all artifacts in the process is extremely important. Specifically with changing requirements it is even crucial to have clear traceability management in place. This is another extremely important aspect in SDLC that needs special attention.

After requirements and traceability I observe that there is humongous discussion about testing techniques for SCS with agile methods. In testing there is a roam to include agile concepts of automation to reduce time and perform it in iterations. (Duffau, Grabiec, and Blay-Fornarino, 2017) have discussed the issues related to increased testing and specification activities when developing embedded systems with agile. To overcome these they have proposed continuous integration ecosystem. They have presented end to end product tests and automatic production of justification documents. They have tested the approach in medical device production company, and conclude by saying that more research is required in the area as they can see potential benefits in terms of time and cost. (Kasauli et al., 2018) have presented a study about mapping agile on SCS. They have catered to studies from 2001 to 2017. They have also discussed the issues with six Swedish company representatives. They have proposed an interesting approach to handle testing. They argue that in testing the helpful approaches can be acceptance and unit testing, continuous building and following coding standards. To ensure safety practices they have proposed the solution as to define and implement certain SOPs that must be followed by all teams on the project. Basically, the measures elaborated by standards are so well defined that there is not much roam for agile to fit in. However, in requirement engineering it can be used as iterative method as it can help elicit and understand requirements in better way. Also the concepts of agile can be helpful in testing.

IV&V techniques are the next most discussed part of SDLC for adopting agile in development of safety-critical systems. Validation-verification techniques have emerged as most critical area that should be handled in a way different than typical agile process models do. This is one of the most critical activity in the life cycle of safety-critical systems but still there is not much evidence of agile amalgamation in this area. There are different approaches proposed in literature to bridge this gap however still industry endorsement is required by proposed approaches. Validation and Verification is still an area that requires further research to mature processes for adapting agile. With incremental development it is challenging and extremely important to have proper validation and verification techniques and requirement management. (Arthur and Dabney, 2017) have presented a study about compatibility of traditional IV&V techniques in hybrid model for developing SCS. They have further stated that there can be three categories of IV&V techniques in hybrid model. First is early life-cycle IV&V techniques that are fully compatible with hybrid model. Second is partially compatible IV&V techniques, which require modification. These are present in parts of requirements and testing and third is IV&V techniques that are not compatible with hybrid process. They cannot divert from traditional approaches. They conclude by saying that since two of the three categories have potential to be used with hybrid model there are good prospects of such models for SCS.

(McBride and Lepmets, 2016) say that there are agile methods that can be used efficiently for development of SCS but the issue is there are no formal validation methods in place for such models. Till now people rely on developers and not on formal methods to perform these tasks. In the same context they have presented a framework called CYNEFIN. The framework proposes to divide system or situation into four categories and proposes a way to deal with it. The four categories are Complex, complicated, chaotic and simple. They conclude by saying that verification and validation is the key area for future research for successfully integration of agile with SCS. Another interesting approach is presented by (Dabney and Arthur, 2019). They have categorized validation and verification techniques in three categories for hybrid agile processes. First is early life cycle techniques, these can be completely adapted with agile. Second category is for those techniques that require amendments but can be tailored and used with agile and third category is non-compatible techniques. They further investigated 7 techniques in last category, non-compatible techniques and proposed alternative approaches that can be used to have similar results with agile.

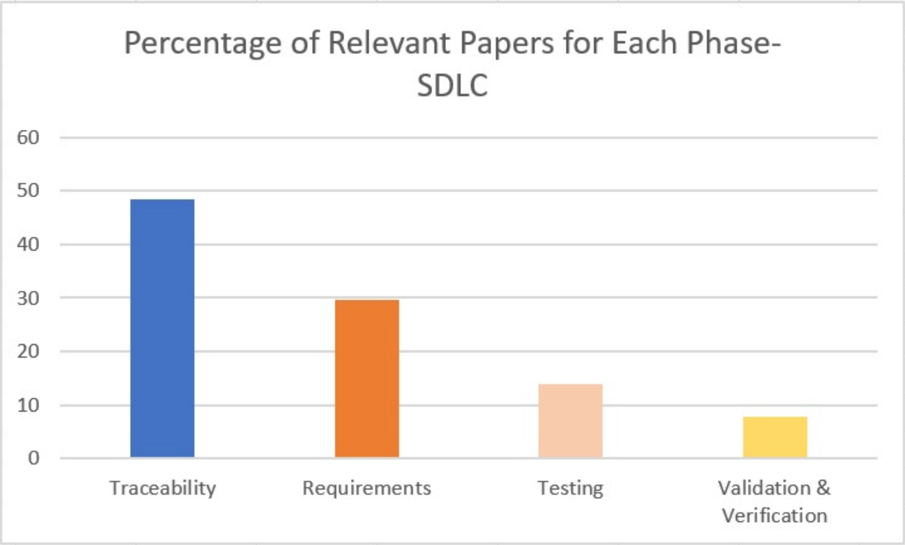


FIGURE 2.8: SDLC

Here I summarize my analysis about characteristics of most discussed phases of SDLC, for including agile in safety-critical systems.

Traceability is most discussed practice of SDLC since it is extremely critical for following reasons

- It is linked to all phases of SDLC.
- Heavy documentation is required for complete traceability which is in contrast to agile.
- Demanded by assessor and safety standard, cannot be compromised to adjust with agile.
- Adds time where as agile talks about as quick as possible deliverable.

Requirement engineering is second most discussed phase of SLDC. Following points of concern are reported in studies

- Heavy documentation is required by standards but agile does not support the idea.
- Well defined requirements are needed by safety standards where as agile talks about accepting changing requirements during SDLC.
- Requirements not only come from client, assessor is also an important factor, hence making it more complex.

TABLE 2.5: Analysis of Phases of SDLC

Phases/Practices of SDLC	Effects all stages of SDLC	Heavy Documentation	Huge Time and Cost	Imposed by Safety Standard - (Not much roam for change)
Traceability	Yes	Yes	Yes	Yes
Requirements	Yes	Yes	Yes	Yes
Testing	Yes	No	No	Yes
IV & V	Yes	Yes	Yes	Yes

- Safety standards impose restrictions-requirements that must be met, and planned before start of project which is in contrast to agile. Agile supports change in requirements even during development.

Testing is also among most discussed areas but relatively lesser touched as compared to traceability and requirements.

- Detail testing of every step is required by safety standards, which adds time to project hence against agile principals. However, it is now handled with automated testing hence does not add huge chunk of time as compared to other phases of SDLC.
- Multiple types of tests from unit level to integration level needed to be performed at every step, also after any change or new addition.
- Increased time and cost with traditional testing which is against agile.
- Upfront planning including testing is required while agile has no concept of upfront planning of any activity.

Validation and verification in light of agile is most newly discussed area, it is extremely critical for SCS and hence need to be handled with great caution. It is one of the hot topics of research in current times. The major prospects reported by studies are

- Strict rules by safety standards for validation and verification hardly any roam for new approach
- Required to be performed at every step of SDLC hence adding more time to timeline
- Difficult to stay in line with traditional validation and verification techniques if you want to accept change in requirements, a trait hugely supported by agile.
- Recently discussed area, currently hot topic but not much proof of application in industry is available as yet.

For future work, there is need of one concrete process model which can address specifically these concerns while finding a middle ground between agile and rules imposed by safety-critical systems. While doing so the above mentioned phases-practices are most critical to handle.

2.4.3 Domains

Here I try to answer my third research question stated as "Which domains of safety systems report highest adaption of agile for their systems?" in section 2.2.1. Agile process models are widely adapted for development of different software. However their use for critical systems has seen a comparatively lower raise. The reason is the contrasting approach between agile and critical software development approaches. However, there are some critical domains where the application of agile has seen more success and hence implementation as compared to other domains.

Medical devices have emerged as most discussed domain for development of safety-critical systems with agile process models. In medical domain there are reported case studies of using agile process models for development of device software. Hybrid process models have shown considerably good results in this case. Hybrid models refer to combination of traditional approaches and agile approaches. Most used approaches are XP and SCRUM which are extended by including traditional models. The main problem is mapping agile principals to FDA regulations,since almost all studies have discussed development in regulation with FDA standards. The major areas of concern for this domain are requirements, testing and verification.In particular (McHugh, McCaffery, and Casey, 2012) have outlined barriers in adopting agile for medical device software. They mentioned the barriers to be regulatory compliance, maintaining traceability, process of managing multiple releases, lack of up front planning and lack of documentation. They have also proposed ways of overcoming these five barriers. However, some studies have argued that it is the mindset of people that needs change and actually there are no huge barriers. One such study is presented by (McHugh, McCaffery, and Casey, 2014). They performed a comparison between actual and perceived barriers and concluded by saying that actually there are no external barriers and it is a mindset and acceptance of new approach that is needed. (Rottier and Rodrigues, 2008) have presented another case study in a company called Cochlear™ a medical device software development firm. They used Scrum instead of traditional approaches and tailored the approaches of requirement management, validation, quality metrics and testing. They say that in these areas they found a middle ground between agile and plan driven approaches. The final findings showed that this approach is better than the traditional plan driven approached for development of medical device software.

The second most discussed domain is automotive and railways. Here I see that it is still in emerging phase. I can find many proposed approaches, models, practices deemed suitable for using agile for development of these systems but not much is available on successful application of these strategies in industry. The strict standards for development in this domain make it difficult to find companies who agree on trying new approaches. Still I can safely say that hybrid agile approach is the next generation for development of these systems and a lot of work is done to find suitable agile approaches with promising results in development of automotive and railway domain. Here testing and verification are biggest challenges as they need detailed traceability of all artifacts which increases documentation, time and cost of project. (Wohlrab et al., 2019) have presented an approach to discuss interfaces in architecture of automotive systems. They have identified three categories of interfaces namely commodity interfaces, early stage interfaces and central vehicle interfaces. Commodity interfaces are mature interfaces that provide platform o long distance working team and changes take time for acceptance and approval. Early stage interfaces are more easy and open to changes although too many change requests are never desirable. Central vehicle interfaces are more critical to handle. They need more upfront planning and ideally there should not be any mid-project changes. Further the paper suggests certain practices to manage interfaces. These practices are proposed on the basis of experiences of interviewees and reasoning based on data. (Myklebust, Stålhane, and Lyngby, n.d.) have presented an intriguing approach. This paper is based on their previous work of using agile with standards like IEC 61508, IEC 60880, IEC 61508 which resulted in a process called SafeScrum. Here they propose to use agile for standard EN 51208. They have further pointed out the areas of concern or using agile for development of systems with EN 51208 which are that there must be software quality assurance plan, a verification and validation plan and software configuration plan. These can be made part of sprints of SafeScrum, an agile methodology. Testing by developer is acceptable only if the assessor agrees to it before the start of the project. The responsibility to ensure that safety requirements are met must lie with RAMS after each sprint. Change impact analysis is an important activity for such systems. (Stålhane, Katta, and Myklebust, 2014) This study have proposed a method to handle change impact analysis. They have covered wide range of topics from safety assessment for railways systems to software development for nuclear systems. In general they have addressed safety-critical systems. The main focus is on change impact analysis. They have proposed an improved method for change impact analysis based on IEC 61508. In this paper the particular focus and improvement is presented for IEC 60880 with inclusion of agile.

Some studies suggest guidelines for better adoption of agile for automotive systems. One such work is done by (Thawaba et al., 2020). The authors of this paper have developed guidelines that can help safety-critical systems' development team to improve the processes. They discuss four major characteristics which are development practices, failures, test techniques and standards.The papers reviews some traditional and agile processes for safety critical systems' development and provides an insight on how to choose the most appropriate one.The failures are discussed as stories to learn from and avoid similar mistakes in future, third they present testing techniques both classical and agile ones with their pros and cons to select most appropriate one, lastly standards are discussed to make the whole approach inline with required parameters of any standard, here the authors have discussed 50128 in particular. They further discussed the four points in light of a case study of railway system. In particular they used agile techniques of PMBOK and SAFE and reported the possibility of positive outcomes by including these in the system.

I can also spot significant papers about avionics who have proposed agile for development of such systems. Here papers stress more on including certain practices of agile into traditional way of development. Practices like continuous integration, pair programming, refactoring, test driven development are considered suitable agile practices for development of artifacts under DO-178. There is also good scope of adapting agile in phases of requirements elicitation and verification. (Coe and Kulick, 2013) have presented an agile based model for development under standard DO-178, an avionic systems' standard. They have taken into account the incremental and iterative nature of agile and applied it specifically in the areas of requirements whole cycle from elicitation to verification. They have further extended the model to perform testing of of requirements in same manner. However after extracting safety requirements they have proposed to make detail design with UML diagrams. After the detail design and requirements are agreed upon by all stake holders testing can start in agile manner. So they sliced the process and perform iterations to complete design and requirements and later they do same for testing. They claim that this process satisfies all requirements of standard DO-178 and can be applied to other safety-critical system with some amendments also. Some practices are considered more applicable in this domain from agile set of practices like pair programming, refactoring, test driven development and overall an iterative approach to develop the system. (VanderLeest and Buter, 2009) report use of such techniques and their benefits.

There are very few studies reported for Space and nuclear plant systems. The research in these areas is still in its initial phase for including agile for development of such systems. (Carpenter and Dagnino, 2014) have named Scrum and eXtreme Programming to be more promising than others, also hybrid approaches are highly suitable for space systems. (Stålhane, Katta, and Myklebust, 2013) have discussed important issues in development of safety-critical systems in this case specifically for nuclear plants conforming to standard IEC 60880. The issues are planning, documentation and proof of conformance.

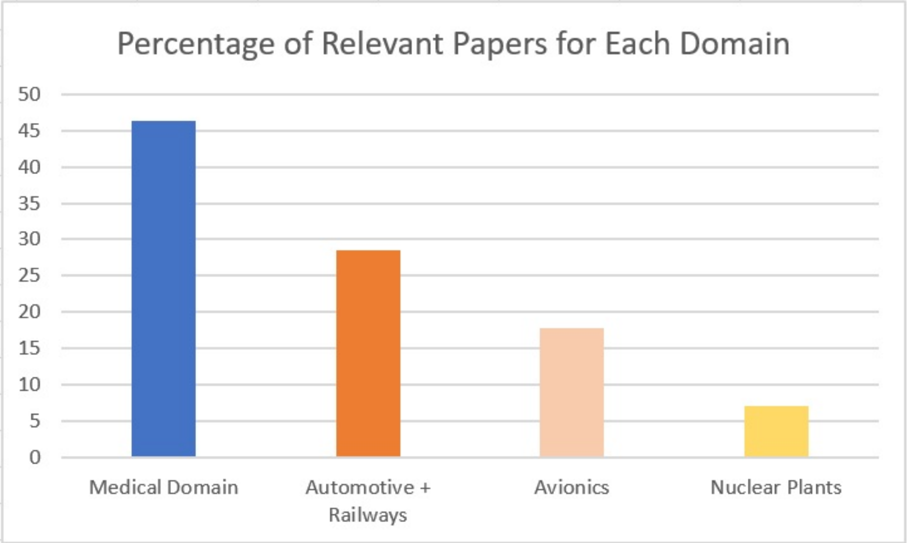


FIGURE 2.9: Domains

Our analysis of studies revealed that there certain aspects due to which medical device software have adapted agile more as compared to other domains. These aspects are

TABLE 2.6: Analysis of Domains

Domains	Size of Software	System of Systems	Complex	Detailed and Diverse Safety Standards	Areas of concern
Medical	Mostly small	Mostly No	No	No	Planning, Requirement management, Testing, Quality aspects, IV & V, Change Impact Analysis, documentation, proof of conformance
Automotive and Railways	Huge	Yes	Yes	Yes	
Avionics	Huge	Yes	Yes	Yes	
Nuclear Plants - Space	Huge	Yes	Yes	Yes	

- Small size of software as compared to software of other safety domains.
- Less complex as compared to software of other safety domains.
- Most reported cases do not report a software which is system of systems.
- Most studies report conformance to FDA guidelines. These guidelines remain unchanged for most kinds of medical device software, where as in other safety domains, safety standards show vast diversity with respect to product in hand and safety level required.

Second most discussed domain is automotive and railways, followed by avionics and nuclear plants. Automotive and railways can also be termed as most prevailing domain for including agile in development processes. I analysed same attributes for these domains as I did for medical domain. The following characteristics of attributes are common among automotive, railway, avionics and space. These factors make it more difficult to include agile in their development, as compared to medical domain.

- Huge size of software.
- Extremely complex.
- Most reported cases in studies are system of systems.
- Diverse safety standards with respect to product in hand and safety level required.

For future work, I perceive automotive and railways to be the most emerging domain for including agile process models in the development phases. However, there is still huge gap in terms of industrial implementation and formalization of process models that can fit best between agile and this domain. There is a lot yet to be explored and implemented.

2.5 Threats to Validity

Performing systematic literature review is a tedious and long process. I have carefully selected the matrix for calculating rigor and relevance of papers, this is internally validated and executed by two people to see its effectiveness of studies. However, external validity of such work is hard to determine.

2.6 Conclusion

I have performed systematic literature review of 76 studies relevant to safety-critical systems and agile. I have used a systematic approach to provide answers to my research questions. I critically analysed the selected studies and then extracted the information useful to answer most relevant and important questions regarding use of agile for safety-critical systems. My major findings are no agile model in its original form is deemed suitable for development of safety-critical systems, on the other hand hybrid process models have been reported in studies to show great success.Secondly, a complete agile based SDLC cannot be adapted for developing SCS, however iterative and incremental nature of agile is useful for approaches of SCS development. Requirements and planning have to be done completely before start of project, rest of the project can be build incrementally/iteratively. Then for testing and verification each project needs to seek appropriate balance between agile and traditional approaches as these have come out to be most critical activities especially in presented scenario. I have also highlighted the problems/challenges and experiences of using agile according to particular domains. It is observed that agile has been applied the most in medical domain followed by automotive/railways and then in avionics. I found some evidence for Space and nuclear programs as well. For medical and automotive/railway domain there is significant evidence to say that agile has shown promising results with major field of concern being requirements, testing and verification. For avionics also the areas of major concern are same however there is growing trend of adapting certain agile processes especially for continuous integration and traceability.

Overall, there is prodigious trend of adapting agile for development of safety-critical systems with some amendments in typical agile process models. They have shown promising results in terms of lowering time and cost and quality of products.

In further I plan to enhance this work further by developing an agile model suitable for development of products under safety standard ISO-26262 and more particularly in terms of systematic literature review I plan to generate a replicability package for this work. It will make it easier to reproduce results using different data hence making the approach more reliable.

Chapter 3

Agility of Security Practices

3.1 Introduction

Agile process models are widely used today for software development. There has been an immense increase in use of agile methodologies due to their major focus on delivering working software and accommodating changes in requirements. However, use of agile methodologies for developing secure systems still poses many challenges. This research, address the issue of observing the effect on agility of process models while security practices are applied in them. An approach is proposed which calculates level of agility of six agile process models (XP, Scrum, FDD, ASD, DSDM, and Crystal) and security practices against four fundamental parameters of agility. When security practices are applied to process models they lower the degree of agility.I propose a method to see this effect based on factor of agility and also that the degree of agility of process model can be adjusted at desired level by including or excluding security practices.

Agile focuses on rapid development and follows the rule of delivering working software in short intervals. One of the major advancement in the field of software is that from past decade developers and all stakeholders consider security as a proper issue (Allen et al., 2008). The current concern is I do not have very compact solutions to address the problem.

3.2 Related Work

Security is an afterthought during software development, it is addressed either very late in development or even after it. Now the question arises what will I call a secure product? According to Microsoft (Michael and Steve, 2006), a secure product is the one that can handle the integrity, confidentiality and customer information along with the confidentiality of processing resources under administrator. Many experts give advice on using agile methodologies for the development of secure systems (Moyon et al., 2018) as there are reported benefits of using agile for software development; however, security cycles are in contrast to agile approaches with the consequence of compromising the level of agility when incorporating security practices (Alnatheer, Gravell, and Argles, 2010).

There is no concrete method to monitor how agility will be affected. Security in its very own nature as a nonfunctional requirement is not easy to cater to. Still, it is the experience in this area that counts the most (Ashraf and Aftab, 2017).

3.3 Methodology

I provide a method to assess (numerically) agility of process models and security practices. There are two major contributions of this work, first is related to the degree of agility for six process models and second is about the degree of agility of twelve security practices in Scrum and XP. The agility of security practices varies from one process model to another; hence, there are dedicated tables to show values of security practices for both of my chosen process models.

The agility of a security activity or a process model refers to how an activity/process model behaves against basic parameters given by agile manifesto. The capacity of a security activity/process model to be flexible, lean, responsive and speedy defines how agile it is. If it possesses higher values of these attributes, it has a high degree of agility and vice versa.I have further assessed the agility of process models after including selected security practices by using a formula. The variation in the degree of agility of a process model before and after application of security practices shows how much agility is compromised to incorporate security. The four parameters of agility that I have used in my work are defined by agile manifesto (Beck et al., 2001a) as

- Flexibility

  Ability to adapt to expected or unexpected changes at any time.

- Leanness

It refers to the improvement of products and services based on the feedback of customers in terms of what they value.

- Responsiveness

  Responsiveness refers to appropriate reaction against expected or unexpected changes.

- Speed

  Speed refers to rapid and iterative development for small releases.

It is important to understand that they are not ranked in any Order. A process must have all of these attributes to be called an agile process.

3.3.1 Agility of Process Models

I have evaluated the agility of six agile process models against four basic parameters of agility based on the work of (Qumer and Henderson-Sellers, 2008). They proposed an approach to calculate the agility of process models in terms of their phases and practices however,I have taken both practices and phases into account and computed a single value that represents the degree of agility of certain model.

Following are six process models that I have considered.

- XP(Extreme Programming)
- Scrum
- ASD(Adaptive Software Development)
- DSDM(Dynamic System Development Method)
- FDD(Feature driven development)
- Crystal.

However, I have used XP and Scrum only for further evaluations, since I was not able to have significant amount of responses for rest of the process models through my survey. Extreme programming (XP) is an agile process model, which intends to improve responsiveness and software quality and cater to flexible requirements (Anwar and Pfahl, 2017).There are four basic phases of XP

- Planning

- Designing
- Coding
- Testing.

Scrum (Schwaber and Beedle, 2002) has major focus on courage, respect, openness and commitment. Scrum has the following basic activities

- Product backlog
- Building teams
- Scheduled meetings
- Sprint
- Sprint review.

The calculation of agility of process models is based on the work of Qumer B.Henderson (Qumer and Henderson-Sellers, 2008). They derived a formula to present the agility of each process model's phases and practices within the range 0 to 1 where 1 represents that the process model is highly agile. In this paper, I have calculated the agility of a process model as whole, including practices and phases. I will use following equation to perform calculations,

$$Degreeofagilityofprocessmodel = Sumofagilefactors/ (no.ofpracticesofprocessmodel * no.ofagileattributes) \tag{3.1}$$

For example, let us consider the process model XP.

Sum of agile factors in XP [flexibility + speed + leanness + responsiveness]

= [15+13+6+15] = 49

Number of practices + phases in XP = 18

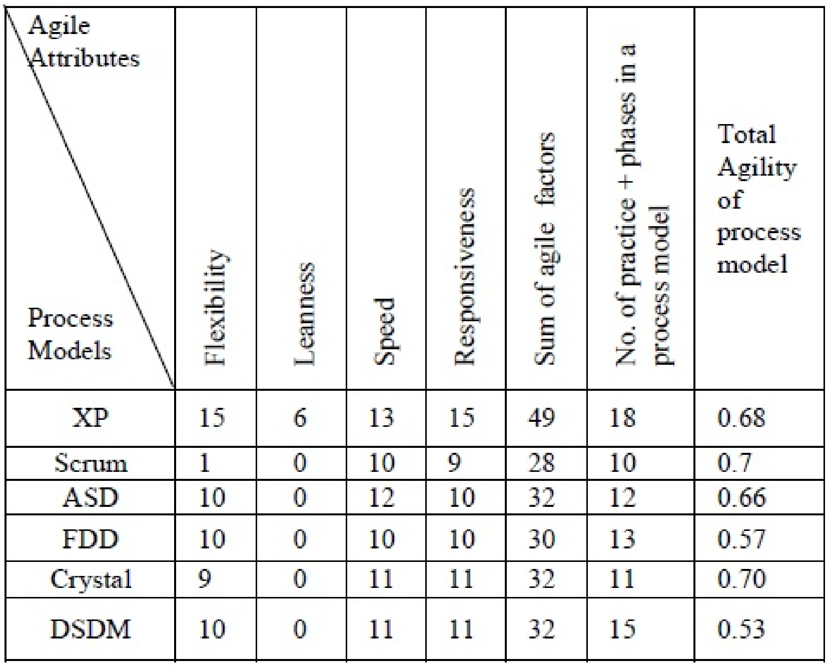
Number of agile attributes = 4

By applying the formula 3.1 I get

49÷ (18*4) = 0.68

It is evident that some process models have higher agility as compared to others. As given in Table 3.1 Crystal and Scrum have highest degrees of agility

TABLE 3.1: Process Models



Agile Attributes / Process Models	Flexibility	Leanness	Speed	Responsiveness	Sum of agile factors	No. of practice + phases in a process model	Total Agility of process model
XP	15	6	13	15	49	18	0.68
Scrum	1	0	10	9	28	10	0.7
ASD	10	0	12	10	32	12	0.66
FDD	10	0	10	10	30	13	0.57
Crystal	9	0	11	11	32	11	0.70
DSDM	10	0	11	11	32	15	0.53

and DSDM has lowest, which refers to the fact that Crystal and Scrum are more agile, and DSDM is least agile in this set of process models. However, my focal point is not comparison between process models; my focus is to see how agility of any one-process model is affected after including security practices.

3.3.2 Agility of Security Activities in Process Models

I have considered the degree of agility of few of the most widely used security activities against four agile parameters (as I did for process models). This approach is based on the work of (Keramati and Mirian-Hosseinabadi, 2008). The values of security attributes are based on survey from industry personnel. There was a need to perform this survey because as per my knowledge there is no numerical data available for such reasoning. I could only find theoretical reasoning through literature, hence I conducted a survey to provide grounds for empirical analysis.

Research Survey

I performed a questionnaire-based survey to observe the agility of 12 most common security practices against four agile parameters in six process models. The research survey serves as descriptive survey and it provides a descriptive analysis. As Oppenheim, (Oppenheim, 2000) describes a descriptive survey provides descriptive analysis only, which refers to frequencies and cross tabulation. According to Oppenheim, (Oppenheim, 2000) descriptive surveys are not meant to describe causal relationship of variables instead their focus is on describing what proportions of sample represent certain opinion or what is the frequency of occurrence of certain events/values.

While selecting the organizations for responses I used purposive sampling. Nardi (Nardi, 2014) describes this method as collecting samples from respondents based on some specific trait, which is important for the study. In this research survey, I have involved organizations that have experience of working with agile AND are using security techniques in agile methods. The research was designed to get responses of individuals', project managers or developers working with agile methodologies. One person can give response for more than one process model according to his experience in relevant model/models. All responses were collected through a web-based survey using Google forms. (Pamela, Settle, and Irwm, 1995) Mentions three major methods for collecting data when performing questionnaire-based survey. First is Personal interviewing second is mail data collection and third is telephone interviewing. However, now days there is very popular method of conducting web based survey. There are many reasons for selecting web-based surveys. All the data you get through them is already in electronic form so it is easy and fast to access the data. In addition, it prevents the chances of errors during manual entries of data as Nardi (Nardi, 2014) suggests. This method is speedy and cost effective too when compared to mail based surveys. It also proves to have higher response rate, which is one of the major issues with other methods of surveys.

Design of Research Survey

I conducted a survey through questionnaire by using 5 point Likert scale (Likert, 1932) method. The responses were collected through web-based questionnaire. Sample is given in Appendix A. I asked industry experts to rank the security activities against four parameters of agility on the scale of (Pamela, Settle, and Irwm, 1995).

5= Strongly Agree,

4=Agree,

3=Neutral,

2=Disagree and

1=Strongly Disagree.

Industrial personnel were chosen according to their line of work and experience. Chosen personnel have significant experience in software development industry. They have also worked with agile software development methods during their experience. Each expert has chosen the process model of his or her expertise and ranked security attributes for that particular process model (Krosnick, Wright, Marsden, et al., 2009). For example, an expert of XP ranks the security practice "attack identification" as 5 (strongly agree) against agile attribute "flexibility", this represents that expert strongly agrees that attack identification is highly flexible in XP. OR in other case, if he assigns 1 (strongly disagree) this refers to not flexible at all in this case. Flexibility and other three agile parameters are defined in accordance to agile manifesto. Figure 3.1 represents sample sources.

In my questionnaire, first two questions were dichotomous questions that represent an exclusive disjunction. My first question, "I have experience of agile process models for more than 5 years". Question 2 states, "I have more than 3 and less than 5 years of experience with agile process models and security practices". The detail of responses is shown in Figure 2. In total, I received 56 responses.I received 18 responses for Scrum out of which 10 had experience with agile and security for less than 3 years, 6 had more than 5 years of experience in agile whereas 2 had more than 3 years of agile experience with security practices.

For XP I got 20 responses, 10 of them have worked with security practices in agile for less than 3 years, 8 had greater than 5 years of agile experience and 2 had more than 3 years of agile experience. For Feature Driven Development I encountered five responses, three of them had experience with security attributes and agile for less than 3 years whereas 1 had agile experience of more than 5 years and last respondent has experience of more than 3 years. For Adaptive Software Development I received 6 responses 3 of which have worked with security and agile processes for less than 3 years and 2 of them

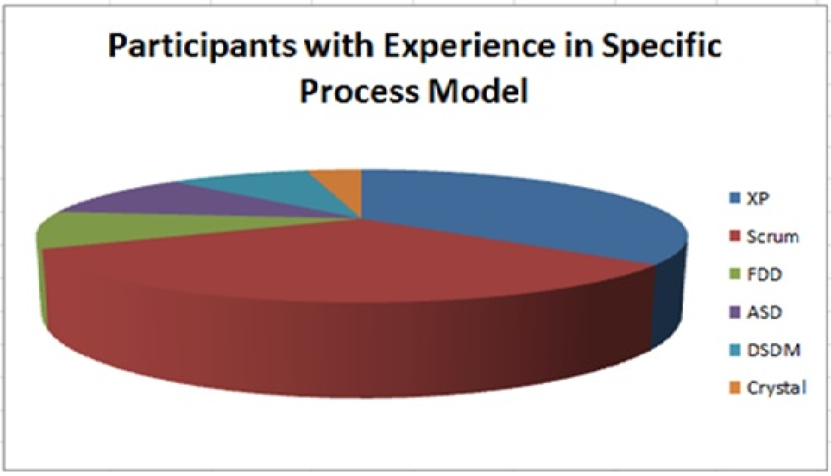


FIGURE 3.1: Process Models

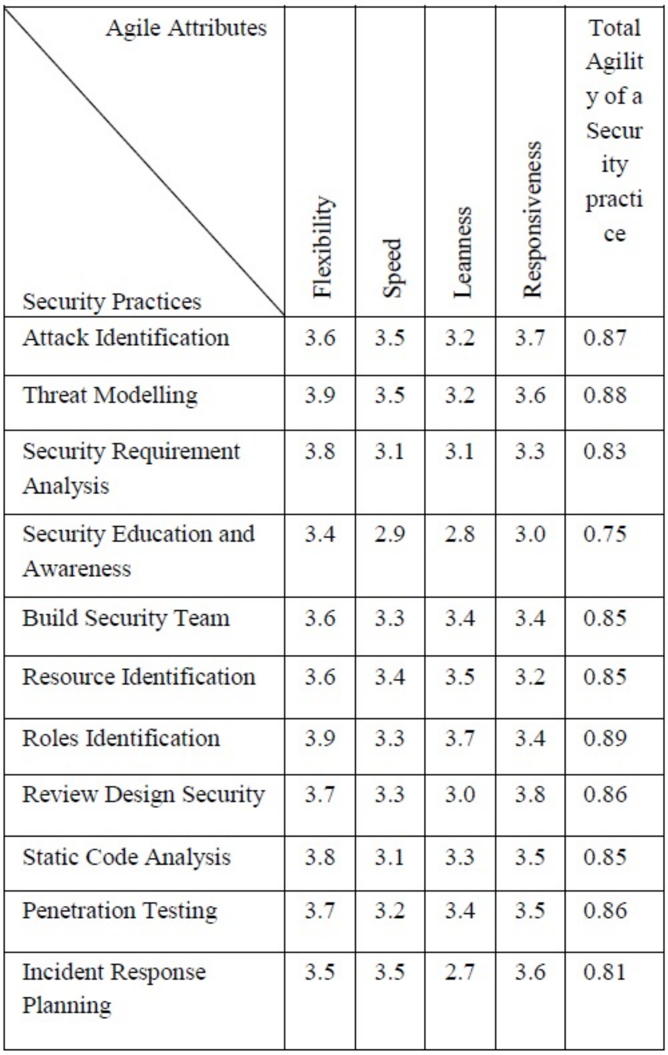
had experience of more than 5 years in agile development whereas 1 had experience of more than 3 years with agile and security practices. For DSDM (Dynamic System Development Method) I had 5 respondents, 3 of them had experience with agile process models and security attributes for less than 3 years, 1 had more than five years of agile experience and last one had more than three years of agile and security experience solely. For Crystal process model I received two responses 1 had experience with agile and security attributes for more than 3 years and 1 had experience with agile for less than 3 years. Further evidence is required for process models, to deduce more reliable conclusions about their agility. However, in this work I provide the application of my methodology on two process models namely Scrum and XP, since I have better number of responses against these models as compared to rest of the four process models. Further work, is required to verify and validate the results of process models with lower number of responses.

Results

The tables in this section represent the final values of security practices against agile parameters for Scrum and XP based on the expert's opinions taken through a survey. I have processed the data of the survey by using a formula given in equation 3.2.

First I have assigned weights to the responses. Weightage 3 is assigned to category A which has people with more than five years of experience, weightage 2 is assigned to category B which has people with more than 3 and less than 5 years of experience and finally category C has weightage 1 for responses from people with less than 3 years of experience.

TABLE 3.2: Agility of Security Practices in Scrum



Agile Attributes / Security Practices	Flexibility	Speed	Leanness	Responsiveness	Total Agility of a Security practice
Attack Identification	3.6	3.5	3.2	3.7	0.87
Threat Modelling	3.9	3.5	3.2	3.6	0.88
Security Requirement Analysis	3.8	3.1	3.1	3.3	0.83
Security Education and Awareness	3.4	2.9	2.8	3.0	0.75
Build Security Team	3.6	3.3	3.4	3.4	0.85
Resource Identification	3.6	3.4	3.5	3.2	0.85
Roles Identification	3.9	3.3	3.7	3.4	0.89
Review Design Security	3.7	3.3	3.0	3.8	0.86
Static Code Analysis	3.8	3.1	3.3	3.5	0.85
Penetration Testing	3.7	3.2	3.4	3.5	0.86
Incident Response Planning	3.5	3.5	2.7	3.6	0.81

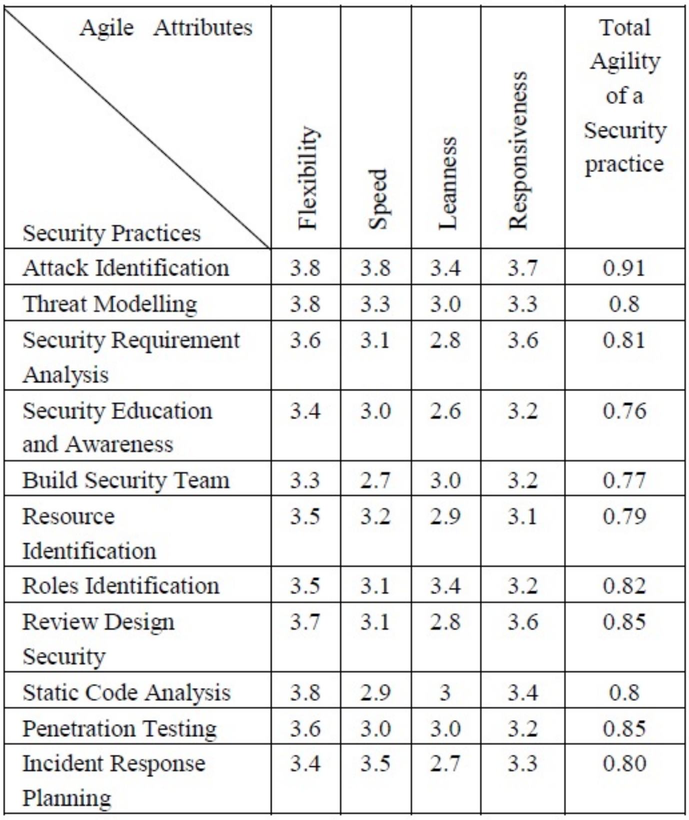
The value of each agile attribute against each security practice is calculated by using the following formula,

$$= [(sumofvaluesbycategoryA * 3) + (sumofvaluesbycategoryB * 2) + (sumofvaluesby categoryC * 1)] \div [(Totalno.ofresponsesofcategoryA * 3) + (Totalno.ofresponsesof categoryB * 2) + (Totalno.ofresponsesofcategoryC * 1)] \tag{3.2}$$

For example, let us consider the process model Scrum.

For calculating flexibility of Attack Identification (security practice), I have (data by experts through survey).

TABLE 3.3: Agility of Security Practices in XP



Agile Attributes / Security Practices	Flexibility	Speed	Leanness	Responsiveness	Total Agility of a Security practice
Attack Identification	3.8	3.8	3.4	3.7	0.91
Threat Modelling	3.8	3.3	3.0	3.3	0.8
Security Requirement Analysis	3.6	3.1	2.8	3.6	0.81
Security Education and Awareness	3.4	3.0	2.6	3.2	0.76
Build Security Team	3.3	2.7	3.0	3.2	0.77
Resource Identification	3.5	3.2	2.9	3.1	0.79
Roles Identification	3.5	3.1	3.4	3.2	0.82
Review Design Security	3.7	3.1	2.8	3.6	0.85
Static Code Analysis	3.8	2.9	3	3.4	0.8
Penetration Testing	3.6	3.0	3.0	3.2	0.85
Incident Response Planning	3.4	3.5	2.7	3.3	0.80

Responses in Category A = 6

Responses in Category B = 2

Responses in Category C = 10

Sum of values (for 'flexibility' of 'attack identification') marked by experts in Category A=21

Sum of values marked by experts in Category B=9

Sum of values given by experts in Category C=36

By applying 3.2 I get,

$$=[(21 * 3) + (9 * 2) + (36 * 1)] \div [(6 * 3) + (2 * 2) + (10 * 1)]$$

Flexibility of Attack Identification in Scrum is = 3.6 (Table 2) (Since I want to find agility of each security practice)

Number of agile attributes = 4

Total agility of Attack Identification is sum of flexibility, speed, leanness, responsiveness.

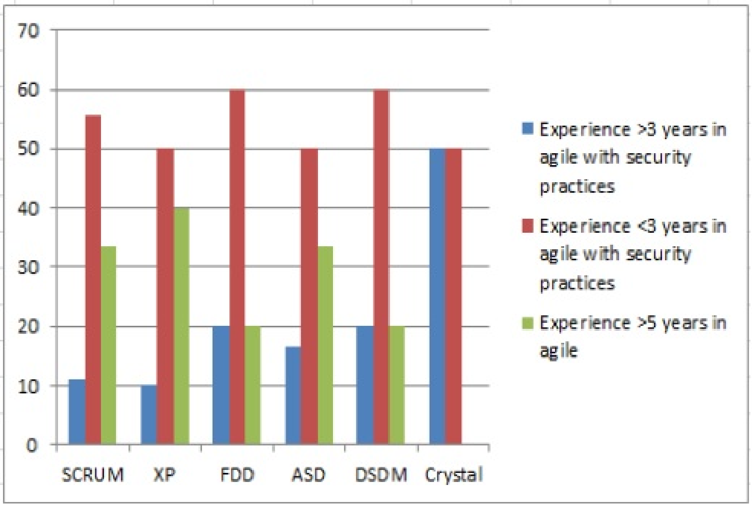


FIGURE 3.2: Percentage of Recorded Responses

= [3.6+3.5+3.2+3.7] = 14

Number of practices considered = 1

(Since I want to find agility of each security practice) Number of agile attributes = 4

By applying 3.1 I get

[14÷(1 x 4)]

Total agility of Attack Identification =3.5

Since I need to see the effect of this value on agility of any process model, I must have it between the ranges 0-1. Hence,

Total agility of Attack Identification

=3.5÷4=0.87

There are different practices and phases in each model and so the agility of security activities differs for each model. For example, Build Security Team Roles has value of 0.85 in Scrum whereas in XP it is 0.77. This is explained by the fact that Scrum has an inherent process of building teams in terms of making Scrum teams thus making the activity more flexible, speedy, lean and responsive whereas, XP has no such inherent process. However, further study is required to understand the changing behaviour of security practices in different process models.

3.4 Applying Selected Security Practices To Process Models

The main purpose of calculating agility of process models and security practices is to see how security practices will affect the agility of process models. Here I will perform the calculations and analysis. Few abbreviations will be used

- AOM - Actual Agility of Model (Calculated previously as Degree of Agility of Process Model) (Range 0-1).
- ART - Agility Reduction Tolerance (Calculated Previously as Degree of Agility of Security practices). (Range 0-1)
- AAAS - Agility after Applying Security practices (Calculated in this section). (Range 0 – AOM)

Note I call agility of security practice, as "Agility Reduction Tolerance" because this value represents the cost a process model will bear for including a security practice. Hence, ART (Agility Reduction tolerance) is the factor responsible for reducing agility of process models.

3.4.1 Formula

$$AAS = [((ARTactivity1 + ARTactivity2...... + ARTactivityn)n)xAOM] \tag{3.3}$$

In this formula, I sum up the agility of selected security activities and divide it by total number of selected activities then multiply the obtained value by agility of selected process model. This would give us new agility of process model after taking out the cost of including security practices. Further explanation and application of this formula is provided with examples.

It is evident that some process models have higher agility as compared to others. As given in Table 1. Further I have applied the approach to Scrum and XP, keeping in view that their data is more unswerving with greater number of responses as compared to other process models.

Scrum

Let us see the effect of including following security practices in Scrum.

Let us take the values for ART of these activities from (Since I want to find agility of each security practice)

Number of agile attributes = 4. By applying 3.1 I get,

$$[14 \div (1 \text{ x } 4)]$$

Total agility of Attack Identification =3.5.

Since I need to see the effect of this value on agility of any process model, I must have it between the ranges 0-1.

Hence,

Total agility of Attack Identification =3.5÷4=0.87 (Table 3.2)

ART of Attacks Identifications =0.87

ART of Review design security =0.86

ART of Static code analysis=0.85

Here,

$$n=3$$

$$\text{AOM (Actual Agility of Model Table 1) = 0.7}$$

By putting the values in 3.3 I get,

$$\text{Agility After Application of Security (AAAS)}$$
$$= [((0.87+0.86+0.85) \div 3) \times 0.7]$$
$$\text{AAAS}=0.62$$

Here, the new agility of Scrum is 0.62 whereas. It is obvious that I will bear cost of including security in Scrum; this work provides an empirical way to see that cost. This can serve as one major factor, (there can be other factors like time (Ayalew, Kidane, and Carlsson, 2013)for selection of security practices.

XP

Let us see the effect of including same security practices (Attacks Identifications, Review design security and Static code analysis) in XP.

Let us take the values for ART of these activities from (Table 3)

ART of Attacks Identifications =0.91

ART of Review design security =0.85

ART of Static code analysis=0.8

Here,

$$n=3$$

$$\text{AOM (Actual Agility of Model Table 1) = 0.68}$$

By putting the values in 3.3 I get,

$$\begin{gathered}\text{Agility After Application of Security AAAS} \\ = [((0.91+0.85+0.8) \div 3) \times 0.68] \\ \text{AAAS}=0.58\end{gathered}$$

In this case the AAAS of XP becomes 0.58 which is lower than its original value (0.68). The effect of including selected security activities in XP is visible in terms of reduced degree of agility. This represents the cost that one has to bear in terms of agility for including security practices.

3.5 Threats to Validity

This work is performed on selected process models and security practices however to enhance this work for other agile process models or security practices further research is required in this area. The formulae presented in this chapter are internally validated and constructed by meticulous examination of problem in hand however, external validity needs further research.

3.6 Conclusion

The effect of including selected security activities can be seen in both process models. This leads to two conclusions. Firstly you are firm to use certain security practices, let us say as your prime factor in this case you can perform the calculations to see the effect of your decision on agility of different process models. Secondly, you are firm to use certain process model and you are ready to adjust security practices keeping the degree of agility of process model as prime factor. Both of above-mentioned approaches can be handled by proposed method.

Chapter 4

Agile Methods for Safety-Critical Systems' Development Life-Cycle

4.1 Introduction

There are multiple issues/challenges that must be addressed for making agile more suitable for safety-critical systems. In safety-critical systems keeping complete trace of requirements and detailed testing is an extremely relevant part of software development life cycle. Safety standards like ISO 26262, DO178C and many others prescribe that critical requirements must be completely traceable. These standards also demand detailed and regression testing of system. Here I present some patterns that deal with these concerns in an agile way. First pattern describes the key mechanism to list the sources of safety requirements and a mechanism for traceability of those requirements. It uses an approach that satisfies safety standards and adapts agile behavior where possible. The next pattern is about up front testing as planning of upfront testing is very important to build safety-critical systems.Up-front testing goes well in line with agile principles, this is the reason I choose it for our proposed approach. Then I discuss test automation for safety-critical systems, which complements our first set of patterns. It decreases the amount of documentation required for traceability and testing of features but without any compromise on essential testing. These patterns will facilitate the team to perform requirement's traceability and regular, rigorous testing in a timely and cost efficient manner. Lastly I will discuss the issues relevant to team formation and communication. Safety-critical systems are defined as those systems whose failure can cause harm (Avizienis et al., 2004). The system is considered safety-critical if its failure can lead to unacceptable circumstances such as loss of human lives or damage to the environment (Avizienis et al., 2004). Development of these systems in an agile way can be very beneficial in terms of time and cost.

Agile methodologies are based on 12 principles and four basic values (Beck et al., 2001b). A project is said to be truly agile if it follows all of these principles and values. However, in safety-critical systems, it is mandatory to be compliant with safety regulations that advocate detailed documentation of each step. This makes use of any single agile technique very difficult as they advocate rapid development without including too much documentation (McHugh, McCaffery, and Casey, 2012). For safety-critical projects, the lack of safety-related documentation can impact communication and can harm the overall process of communication, especially in terms of safety requirements (Wang, Ramadani, and Wagner, 2017).

Along with eliciting requirements their traceability and testing have always been considered a very important property for well-engineered software. The definition is given by Center of Excellence for Software and Systems Traceability (Cleland-Huang et al., 2011) (CoEST). They state that “ability to interrelate any uniquely identifiable software engineering artifact to any other, maintain required links over time, and use the resulting network to answer questions about both the software product and its development process is traceability" (Cleland-Huang et al., 2011). Traceability is an integral part of the certification and approval process of most of the safety-critical systems. Though traceability is a very important factor in safety-critical systems (Cleland-Huang et al., 2014) (Górski and Łukasiewicz, 2012), yet it is a very elusive quality of the software development process. The effort, cost, and formalism required to maintain links for traceability are considered extremely high and do not go hand in hand with rapid development like agile (Arkley and Riddle, 2005) (Gotel and Finkelstein, 1994). Considering the importance of formal procedure for traceability of requirements, the U.S Department of Defense has identified the procedure as one of the seven most critical and needed research areas. According to them, it must be targeted to ensure the safe and accurate operations of present and future software-intensive systems (Council et al., n.d.).

The agile manifesto states that it gives priority to working software over detailed documentation (Beck et al., 2001b). It seems there is a conflict between heavy documentation and agile principles (Stettina and Heijstek, 2011). Cohen (Cohn, 2010) argues that there are issues that must be looked upon in written communication and we should not abandon documentation, instead we should use the documentation at and for appropriate points, especially for the development of safety-critical systems (Wang, Ramadani, and Wagner, 2017).

Test automation is one of the key concepts when we try to develop safety-critical systems with agile methodologies. There are certain challenges that a team needs to overcome when they want to develop safety-critical systems by using Scrum or XP (the two most widely used agile approaches). These challenges can be test automation, short iterations, continuous integration and regression testing (Tyagi et al., 2018). As the project moves, it becomes more complex to make sure that previously developed modules are working fine. In this scenario, automated tests can prove to be very useful.

'Everyone Responsible for Safety' is an idea about making safety as part of team responsibility. In Safety Verification I will work on an approach similar to Test Driven Development that argues to define test cases with requirements.

In current practices, development of safety-critical systems is carried out using traditional approaches like waterfall, V model etc. Our proposed patterns are developed for the teams who want to entirely replace traditional models or want to use agile models in collaboration with traditional models.

There are multiple issues/challenges that must be addressed for making agile more suitable for safety-critical systems. The basic principles of agile say that there should be rapid development, strong communication among all stake holders and changes should be welcome at any stage of development (Alliance, 2001). As there is a lot of focus on people so every individual in team should be motivated and must be given suitable environment and support to perform their jobs (Stavru, 2014). Hence team building and communication is an important aspect for this approach.

(Stålhane-IDI, n.d.) To build a successfully functioning team It is extremely necessary that whole team is on board regarding safety aspects of the system. (Leite, 2017) Agile teams provide room for requirement changes during the development process. In case of safety-critical systems this needs to be handled with great care. Agile has major focus on team collaborations and interaction among all stakeholders (McHugh, Conboy, and Lang, 2011) and promotes self organizing and cross functional teams (Boehm, 2002).

The critical nature of safety-critical systems requires that if not all , maximum risks are handled during development of these systems. However, it is not enough to just perform risk analysis, certain safety standards must be Incorporated during development of these systems, for this reason they must have testing strategy in place (Zimmermann et al., 2009). There is a lot of room for improvement of this mechanism to fit perfectly to the needs of safety-critical systems especially in agile perspective (Tracey, 2000). In short we can say that

- Agile stresses on cross-functional autonomous teams (Chen et al., 2015) whereas, safety-critical systems need specific experts doing specific tasks.
- There is also a gap to be filled about how organizations should arrange teams to achieve the right level of autonomy in any particular scenario, in our case for safety-critical systems (Hoda, Noble, and Marshall, 2012b).
- It is a challenging task to create cross-functional teams and keep the size of the team small .(10-15) Persons on team, which is a standard agile team size (Stray, Moe, and Hoda, 2018).
- (Holcombe, Ipate, and Grondoudis, 1995) states that formal verification methods and test driven development can be used together and there can be many enhanced benefits of this approach. It can ensure safety of systems along with reliability. It further argues that safety-critical systems have to comply to certain safety standards, currently there are not many techniques of testing that can fit into this scenario.

The documentation of these patterns is part of an effort to identify practices that can be used to apply agile in safety-critical projects. I have performed a systematic literature review to find out the key issues and problems in adapting agile for safety-critical systems. It was concluded that agile cannot be adapted completely for development of safety-critical systems. However, there are some phases of software development life cycle where we can incorporate agile principles. As shown in figure 4.1 there are some common grounds where we play to indulge agile in development of safety-critical systems. This chapter has four major sections. First section describes proposed pattern for requirement elicitation and traceability, second and third section discuss the approaches for testing and fourth section is for Agile teams, their characteristics and communication for development of safety-critical systems.

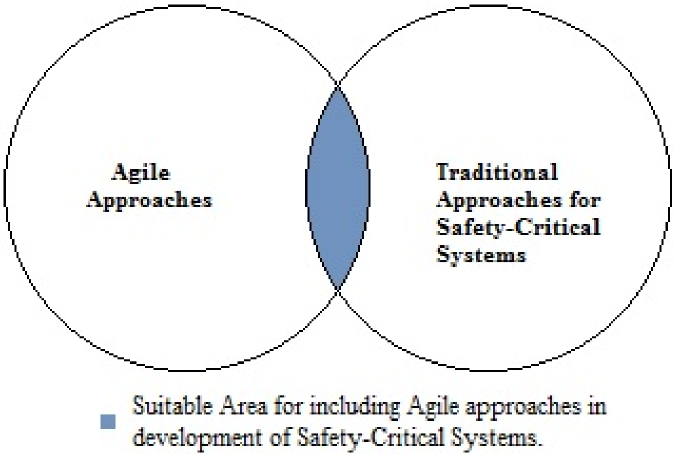

Agile Approaches
Traditional Approaches for Safety-Critical Systems
Suitable Area for including Agile approaches in development of Safety-Critical Systems.

FIGURE 4.1: Collaboration of Agile Approaches with Traditional Approaches for Safety-Critical Systems

4.2 Identification and Traceability of Safety Requirements

4.2.1 Context

In agile process models requirements are communicated in a very informal way, mostly by on-site meetings with customers. On the other hand, the traceability matrix requires a formal requirement specification document to proceed with the process (Ghazarian, 2008). The principles of agile processes make it difficult to follow a static document-centric model. To establish a link between requirement changes and design constructs, new/changed requirements must be mapped onto previous specifications. Agile in its nature does not provide room for traceability, hence to incorporate it in agile, the process has to be integrated from the very first step. The initial informal meetings about customer needs and their refinement to requirements must be addressed formally. Developing software is an exploratory process and thus there must be some mechanism to handle the change or more formally a mechanism of change management must be in place. To create safety-critical systems with agile, quality of process must be maintained with best practices for planning, traceability and continuous gathering and validation of requirements (Ghazarian, 2008).

Solutions proposed for agile processes must be supportive of these human-centric practices and must be adapted in the form of user stories, plans or tasks to accommodate changes.

For example, the DO-178C standard (SC, 2011), which the USA Federal

Aviation Administration (FAA) has established as the means of certifying that software aspects of airborne systems comply with airworthiness requirements, specifies a very detailed set of traceability requirements including the need to provide traceability between source code and low-level requirements "to enable verification of the absence of undocumented source code and verification of the complete implementation of the low-level requirements." Similarly, the USA Food and Drug Administration (FDA) states that traceability analysis must be used to verify that the software design implements the specified software requirements, all aspects of the design are traceable to software requirements and that all code is linked to established specifications and test procedures (Freude and Königs, 2003).

Christopher Lee et al. (Lee, Guadagno, and Jia, 2003) claim that many informal techniques are used in agile methodologies especially for elicitation of requirements and implementing them. They further argue that the document-centric approach does not fit well with agile principles. To deal with the issue of traceability, it is important that we first deal with the technique for gathering requirements and refining them into formal specifications.

Scrum is one of the most widely used process models. It has user stories to deal with requirements which are a very informal way of handling requirements. It is based on meetings and views of different stakeholders, taken in terms of user stories written in no particular format. Jacobsson (Jakobsson, 2009) argues that teams who follow Scrum consider it to be the magical solution. They think that, by following Scrum from start to end at each step, all problems will be solved, which is not the case in reality. Especially the informal way of handling requirements is not sufficient for traceability. Scrum suggests that for each sprint there is a meeting of all stakeholders and they come up with user stories as requirements written in an informal style. However, the challenge is, if we start making detailed documentation in every sprint, the process will loose its ability to deliver modules in short time sprints.

There is a need for some mechanism that can suggest a way of taking and tracing requirements (Wang, Bogicevic, and Wagner, 2017) while staying agile in Scrum. By staying agile we mean doing it in a short time for each sprint of Scrum. This (Rasmussen, 2003) becomes more critical when we talk about safety-critical systems where traceability of requirements is not an option instead it is a must-have property of the process when dealing with the development of safety-critical requirements (Gayer et al., 2016) (Cleland-Huang, 2006). This difference in the basic nature of both processes makes it difficult to perform traceability with agile process models (Wirfs-Brock, Yoder, and Guerra, 2015).

4.2.2 Forces

- **Traceability**: Traceability is an issue in safety-critical systems in its very own nature also, as for safety certifications it is required to have a complete track of traceability of requirements. To perform this efficiently is a challenge in itself.
- **Up-Front Design**: Traditional approaches advocate up-front design; hence all requirements are agreed upon in the beginning and are locked at the end of the requirement phase. On the other hand, agile advocates evolutionary design instead of planned upfront design and hence accommodates changes throughout the development phase.
- **Safety Analysis Techniques**: The safety analysis techniques are designed for traditional approaches; they do not fit well with agile methodologies since they need a pre-defined stable architecture and set of requirements.
- **Requirement Elicitation**: Requirements elicitation takes place in the form of conversations, where insights into the problem space, clarification of assumptions, and deeper understanding takes place. While these conversations are rarely captured because of their ad-hoc nature, tools should provide a mechanism to record unstructured requirements elicitation and transcribe them to a more structured model when appropriate (Lee, Guadagno, and Jia, 2003).

4.2.3 Eliciting and Tracing Safety Requirements

Problem

It is evident that to find a middle ground between agile principles and requirement traceability matrices we need to keep documentation composite. Hence all requirements do not need to be traced, there must be an approach to identify critical safety requirements which must be documented and updated.More specifically, in traceability pattern I address the problem of

“What factors determine that any given requirement is safety related requirement?"

Solution

In this pattern, I propose a method to identify safety requirements. As advocated before I propose to use an amalgamation of the traditional modeling approach and agile principles. It will shorten the documentation and time the agile team needs to spend on documenting requirements. Instead of taking a formal approach for all requirements, they will do it only for selected ones.

I propose a feature based criteria to select requirements for traceability. At the beginning of the project, the requirements can come from:

- Safety standards,
- Project Stakeholders,
- Safety analysis.

At this point, in the beginning, I take these requirements and relate them to each feature of the product. One requirement may be related to more than one feature. For example user authentication is a requirement which will be fulfilled by developing a feature of login. But "login" can be required to access many other features of the system as well. Hence, when those features are developed it must be checked if login is still working properly OR it is not effected adversely. I design iterations around these features. In each iteration, I track safety requirements relevant to the feature being developed figure 4.2.

At the start of every iteration, the whole team will have a meeting to decide safety requirements or if there is any change/update in these requirements. During the development, the change can come through two sources:

- Changes to existing safety requirements,
- Changes that will influence code that directly or indirectly belongs to a safety requirement.

The rest of the process for enlisting and tracking requirements remain the same during the whole development process.

To mark any requirement as “DONE”, it must be verified that a relevant feature has been completely built and delivered AND all dependent features of that requirement are also built and delivered as represented in figure 4.2. The requirement will not be considered "DONE" if there is any feature of the system directly or indirectly related to the requirement that is still in the process of development. Then it proposes to have upfront design, enough to start the project but allows space to change or update it during the whole development life cycle. It can fit into the iterative nature of agile while carefully keeping track of changes relevant to safety without compromising safety standards.

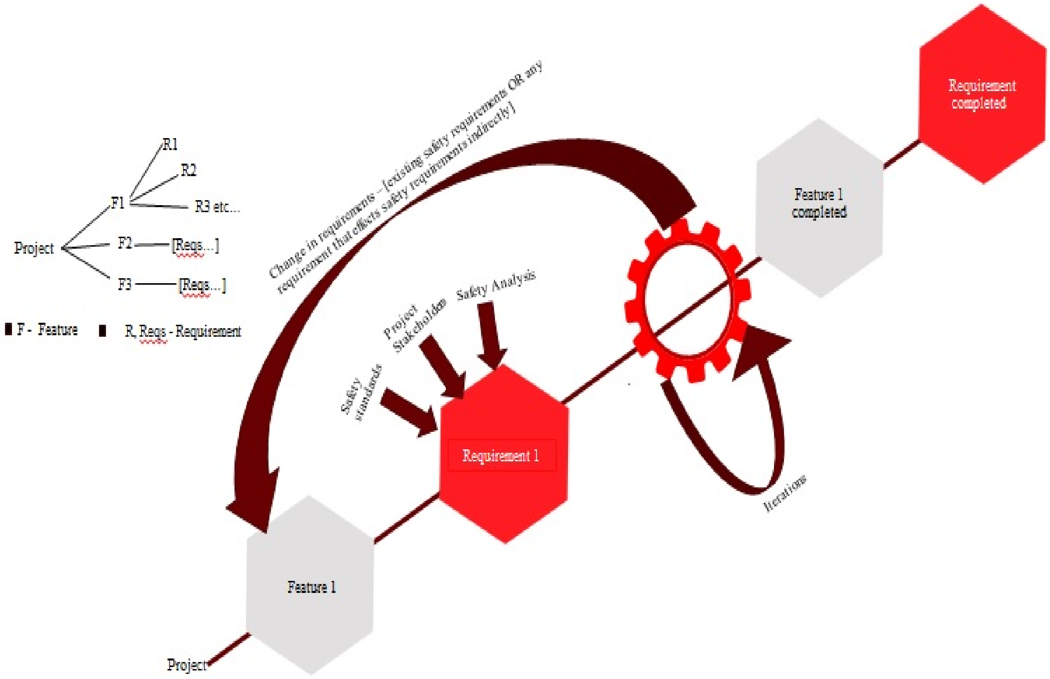


FIGURE 4.2: Requirement marked as "DONE"

Consequences

(+) A list of upfront high-level safety requirements makes it compliant to many safety standards.
(-) An important requirement may not be identified and will not be traced.

Problem

"How to trace safety requirements throughout software development?"

Solution

This is a pattern for traceability of selected requirements throughout the software development life cycle. The method is a hybrid approach of traditional safety-critical development approaches and agile principles. Identify safety requirements and trace only selected requirements during the whole development process. Use an iterative approach to develop the project as it will pave a way for more validated short and timely outputs, also team does not need to plan everything upfront. However, it is important to note that the team must have enough knowledge of requirements in hand that they can sketch an idea of the whole system to start with.

The method to keep track of requirements must be formal. It can be a document, database or some online tool that facilitates such tasks. Each safety requirement will have a code attached to it which will represent some basic information like relevant features, unique id, and version. Codes for features can be selected by the team working on a project. If there is any update/change in the requirement, its updated version will be created. If there is any update or change in any requirement, all of its mentioned "relevant features" must be checked for updates and tested for validity and verification.

This will balance out the problems that teams face when adopting agile process models for the development of safety-critical systems. It will provide a form of upfront planning but can deal with the iterative nature of agile. It will not create too heavy documentation to handle, as not all requirements are accounted for. The time required to perform these steps at the beginning of the project and start of every iteration is not very long and hence the principle of "working software in short iterations" is not affected. It will move the team from the ad-hoc approach of taking requirements to formal documentation but only for a particular set of requirements, safety requirements in this case. Since it is only for specific kinds of requirements it will not add burden on a process in terms of time or documentation.

At the end of each iteration, the whole team will discuss all requirements for next iteration, as is normal practice in agile, but now they will select safety requirements that will be traced throughout the process OR will update the trace document of safety requirements. This update can be in terms of adding a new requirement, modifying any current requirement, giving the status of "DONE" to any requirement OR selecting a set of requirements for the next iteration. The second idea addresses the complete goal in terms of short, small goals. By completing the goal, I refer to the final product required. A team can collectively decide the goals to be safety-critical. So when there is any change in requirements, which is directly related to one of these goals or it has an impact on these goals, I need to include that change in traceable requirements. These values can be mapped to create a matrix that can be evaluated for each iteration.

This pattern can bridge the gap between agile principles and safety standards. It can work with comparatively less documentation but at the same time it maintains log for safety requirements.

Consequences

(+) The maintenance of the traceability documentation is reduced because of a short number of requirements to be traced.
(+) The proposed pattern complements the meetings of agile teams before each iteration, it will help to decide and update traceability document for safety requirements.
(+) The pattern supports the idea of using automated tools OR document self-created by the team.
(-) With iterative development, it is difficult to validate and verify system-level behavior.

4.2.4 Known Uses

Wang et al. (Wang, Bogicevic, and Wagner, 2017) applied Scrum, an agile process model to the Safe Home project, a safety-critical system. They applied an approach to safety stories and STPA for developing this safety-critical system. Their research provided the first practical and empirical view of applying agile to safety-critical systems. With this approach, they found that there were challenges in communication, priority management, and acceptance criteria for safety requirements. They suggested looking for solutions in terms of safety stories, pre-planning meetings or regular safety meetings (Wang, Ramadani, and Wagner, 2017).

Implemented (McHugh, McCaffery, and Coady, 2015) the agile practices within Abbott Diagnostics. The company completed two projects side by side one with agile methodology and another with the plan-driven approach. They concluded that the projects they completed with agile have cost savings of 35 percent to 50 percent as compared to plan-driven projects. They also faced some challenges with an agile approach, which included accommodating changes, applying the agile approach completely. They had to adopt a hybrid approach by combining VModel and Scrum. They reported having traceability issues, lack of upfront planning and managing multiple releases with some other ones.

(Gary et al., 2011) Applied agile approach to image-guided surgical toolkit development. It reports among major issues the problem to perform hazard analysis, specifying and analyzing safety requirements, testing those requirements and compliance with safety standards and certification.

All of the above mentioned have used Scrum from agile process models and have reported problems in the tracking of requirements and analyzing critical requirements in the process, this is the issue addressed by our proposed pattern.

4.3 Pattern for Up-front Testing for Safety-Critical System

In this section I present a pattern for testing of safety-critical systems. This pattern has two subdivisions which give complete guide on how to plan up front testing and how to deal with testing of changing requirements. Here again like our previous pattern I propose to adapt the pattern with all its sections however according to suitability with project in hand, any one section can also be applied.

4.3.1 Context

Failure of safety-critical system can result in loss of lives, loss of huge financial amount or can be a threat to environment. In the light of criticallity of software, it is highly required to develop testing techniques that can give maximum test coverage and hence ensure that the system is safe. The approach to develop tests upfront have gained huge popularity among practitioners. This strategy proposes that tests should be written well in advance, as soon as requirements are defined. Tests must be written to fulfill those requirements (Janzen and Saiedian, 2005). Safety-critical systems are complex and there is need of rapid development in current time. Developers are implementing approaches that support the idea of testing in this context (Hause et al., 2010).

Testing can be tough task to handle especially when developing with agile approach. Testing needs to be handled in same way and given same importance as other artifacts during the process, or even more. Preparing a right suit of tests can save a lot of time and effort; carefully developed test strategy can be used repeatedly and incrementally. In today's world it even has more importance due to rapidly changing nature of software requirements and speedy integration needs. This brings in the demand of efficiency and re-usability. According to (Almog and Tsubery, 2015) Almog testing is the most expensive part of development.

4.3.2 Forces

- **Rapid Development**: Agile promotes rapid development but safety-critical standards need prolonged and detailed procedures for testing.
- **Safety Standards**: Safety-critical systems have to comply to certain safety standards, currently there are not many techniques of testing that can fit into this scenario.
- **Rigorous Testing**: Safety-critical systems need rigorous testing which is time consuming.Agile concentrates on shortening of time.

Problem

"How to inform developers about verification's required for safety-critical system, as early as possible?"

Solution

Include testing resources in the requirements elaboration so there is a plan from the beginning on how to test. Each feature should have its own defined tests in the beginning. The team should identify the requirements concerning one particular feature of the system, then create iterations required to completely develop that feature, covering all the listed requirements for that feature. Consider safety standards and break safety standards into safety requirements with traceability to testing and test results. As soon as the requirements are finalized, in the start of the project, there must be tests written for every requirement. A requirement will be considered done when it passes its relevant test at end of iteration. This way tests will be designed up-front, before starting any iteration for development of any particular feature.

Consequences

(+)Up-front testing goes hand in hand with safety-critical approach of up-front planning whereas, it reduces the time for testing which is required by agile principles.
(-)The approach is highly dependant on clearly and correctly defined requirements which is a challenging task on its own.

Problem

"How to handle testing for changed requirements during development of safety-critical systems?"

Solution

In case there is any change in requirements, all the tests, directly or indirectly relevant to that requirement will be repeated to consider it "DONE" again. When more than one feature is developed, there will be tests at feature level too. All features must be tested individually and as whole after deploying any new feature. Tests for previous features will be automated whereas, new one will be tested manually. It will ensure that new functionality has not effected any of the previously properly working features. This way the tests will increase as I move to the completion of software, making sure that each requirement and each feature is tested. Once tests are designed and performed, they can be automated for next cycles of testing, it will save time and effort of developers. This will provide a middle ground for agile and safety-critical principles to work together without neglecting any important aspect of the system. Still there is need of manual testers even after good automated test coverage. Incorporate security into the requirements backlog. This can be done in two ways: associate risk and failure modes and safety into the attributes of each requirement, and have safety requirements, both should be done in parallel. They are not alternatives to each other instead these are parallel running processes. They will operate the HIL (hardware in the loop testing) and they need to be domain people. They need to be able to poke holes in the operation of the system (and these guys are super to also be involved in the requirements!!).

Consequences

(+) This approach will provide rigorous testing in comparatively less time frame as traditionally consumed for testing in safety-critical systems.
(+)This approach makes it easy to come up with automated tests once they are designed and performed for first time, it will save cost and effort of developers without compromising quality aspects.
(-)The approach is highly dependent on humans so inherently it is prone to errors and dependant on their skills.

4.3.3 Known Uses

The approach of doing up-front testing was adopted by Ericsson (Damm, Lundberg, and Olsson, 2005), they used an automation tool for the process. They used a typical Test Driven Approach along with automated testing tool. They did it at component level by exchanging XML data in the place of methods and classes. This methodology made it easier for them to automate tests. Their team reported that project's deployment time will be shortened with every new version of the project that uses the same tool and technique. They further added that it has decreased fault rates significantly (Damm, Lundberg, and Olsson, 2005).

In another case study (Williams, Maximilien, and Vouk, 2003) the approach of up-front automated tests is applied at IBM. They created automated tests after creating UML design. The team at IBM changed their practices from ad-hoc processes to test driven development. The team was not experienced with this methodology still they reported positive results. They particularly mentioned there was reduction in density of defects in new or changed code, as compared to experienced team who used ad-hoc processes. Another huge benefit with test driven development was its reuseability. So these tests became an asset of organization and further will be used again in other projects to enhance their quality without going through all the effort. They reported 40 percent less defects during testing and verification processes as compared to the product developed through traditional ways (Williams, Maximilien, and Vouk, 2003).

4.4 Pattern for Test Automation of Safety-Critical Systems in an Agile Way

4.4.1 Context

In recent years agile methodology has proven to be useful in the process of software development for different kinds of software. Here I am particularly concerned with safety-critical software development. Testing is another huge area that needs to be addressed when I talk about safety-critical systems. Testing can be manual or automatic. Safety-critical systems go through rigorous testing as a part of their basic development process. When we try to address the development of safety-critical systems automated testing can be helpful to adapt agile for their development as in agile we are looking forward to reduce documentation (Zhang et al., 2010) (Hoda, Noble, and Marshall, 2012a) and time. Dustin et al. (Dustin, Rashka, and Paul, 1999) give definition of automated testing as the automation of tests include the usage of automated tools for testing, execution of scripts for testing and verification process for testing of requirements.

Karhu says (Karhu et al., 2009) that manual testing takes a lot more time as compared to automated testing. Automation of tests also increases the efficiency for performing the steps to produce same functionality of system repeatedly. This is in particular a huge help in performing the tests repeatedly and iteratively when there are any changes to the software. Automation of tests can take up to 50 percent of total project effort, but it is worth the investment.

4.4.2 Problem

It is evident that testing is one of the major areas of concern for safety-critical systems. When we try to develop such complex systems in an agile way, automation of testing needs to be handled with extreme care.In this pattern I address the problem of

“How to perform automated testing in order to achieve the reduced documentation and time in the process of developing safety-critical systems by agile approaches?”

4.4.3 Forces

- **Documentation**: Agile suggests using lean documentation, testing needs detailed documentation that needs regular updates.
- **Traditional Testing**: In traditional approaches, testing is not an iterative or incremental process, instead, it is considered a linear process. In agile, testing must be handled iteratively. Since all agile approaches work to deliver working software in small iterations. For example, scrum works in sprints; each sprint is an iteration and at the end generates some output. The idea is to deliver running modules in short sprints. Similarly XP, another widely used agile process model, advocates small releases of running software
- **Cost and Time**: Testing is most expensive part of project, needs a lot of time and cost to perform, whereas agile focus on rapid development in minimum cost possible.

4.4.4 Solution

I present a testing strategy as continuation of requirement traceability pattern defined above. As I am grouping requirements on the basis of system features, I propose to do the same with testing. I propose to have automated tests to check each feature of the system. As the process is iterative and incremental, with each iteration new changes can be checked manually, whereas the whole feature can be tested automatically to see the effect of change. And to make sure that even after the proposed change system's particular feature is behaving appropriately. In safety-critical systems I have to keep track of every requirement, along with any change requested in the requirements, testing in this scenario can be automated to certain extent only and with great care. In the light of critical nature of safety systems, I propose to perform manual testing of each requirement when it's implemented for first time. When the second requirement belonging to same feature is implemented, first is tested again automatically. In this way the automated testing is performed incrementally along with requirement implementation. Each new requirement is tested manually, and all previous requirements are tested automatically related to one particular feature of the system.4.3 When one feature is marked as complete, the same procedure is repeated. Let's say we have feature 1 completely developed and tested, now we are implementing requirement 1 of second feature, this requirement will be manually

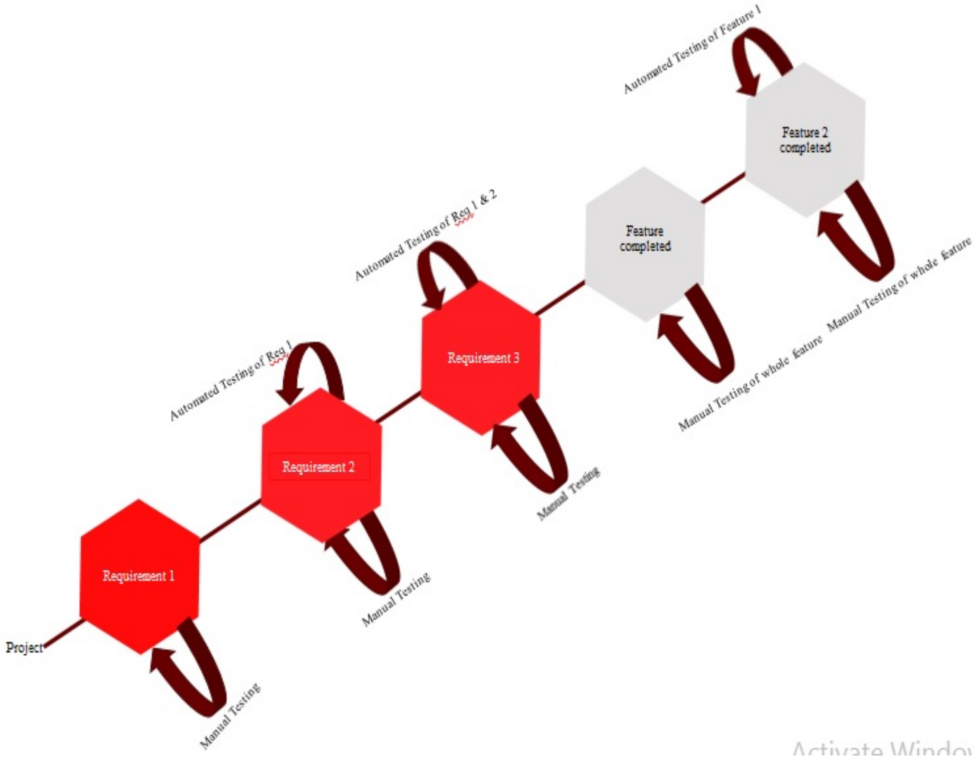

Automated Testing of Feature 1
Feature 2 completed
Automated Testing of Req 1 & 2
Feature completed
Requirement 3
Automated Testing of Req 1
Requirement 2
Manual Testing of whole feature
Manual Testing of whole feature
Manual Testing
Manual Testing
Requirement 1
Project
Manual Testing

FIGURE 4.3: Test Automation

tested whereas feature one will be automatically tested during testing phase. When any change comes in a requirement of any feature, the changed requirement's implementation will be tested manually and all completed features will be tested automatically in testing phase, to make sure that there is no harm done to already completed features of the system. The proposed approach will be repeated with every new requirement, any change in current requirements and every iteration of development. This will decrease the load of manual testing, although we cannot eliminate it completely, but this hybrid approach paves a way to develop safety-critical systems with agile approach incorporated by finding a middle ground to suit the needs of both.

4.4.5 Consequences

(+)Through automation of testing, there is no compromise on testing at any stage, automation only helps in reduced documentation and effort.
(+)The proposed pattern fits well with the current team structure of agile where teams are multi-functional. The same team can work on requirements and testing to consider the requirement as "DONE".
(+)This pattern supports the idea of using automated tools to shorten the time span of testing.

(+)The proposed approach of testing will enable to test multiple features at every iteration; hence can be useful to test system level behavior.
(-)The process of testing is dependent on identification and implementation of requirements if an important requirement is not identified it will not be tested and traced.
(-)The process is completely dependent on people working in a team and hence chances of human error cannot be neglected.

4.4.6 Known Uses

(Jee et al., 2010) Have presented a case study on testing of software for nuclear reactor. These systems are safety-critical systems and hence highly fragile. The process adapted for development must be thoroughly traceable and it must conform to certain safety standards. Jee et. al (Jee et al., 2010) have presented an approach for automation of testing for this system. Their tool provides test coverage for function block diagrams, which are intensively used in such systems. They took test cases prepared by experts of this domain for protection system software. By using automated testing through the tool they developed they found out that there were many paths not covered by manual testing performed by expert testers. Further they have demonstrated in a quantitative way a detailed analysis of results generated by tests. Their case study promisingly conveys that the idea of automated testing is extremely effective in safety-critical systems. Further they argue that this approach is more effective in terms of accuracy. It is highly intuitive and provides continuous monitoring progress.

They (Höller et al., 2014) applied the software based self-tests referred to as SBST frame work for automation of tests to support the design of an industrial programmable logic controller. This unit was designed for hydroelectric power plants. The approach provided extremely efficient feedback on software based self-tests in terms of detection of faults and summary of the diagnostic coverage. They further argue that this approach can be used in up front design, since it does not require hardware. Up front design is one of the main requirements for development of safety-critical systems. It works by injecting faults in the system for testing and hence approach can be used in an agile manner of development too by introducing new faults to check newly developed features of the system. Both of the case studies presented above argue the importance and usefulness of automated testing in safety-critical systems. Furthermore, it can be observed that the approach used for automation of tests in these cases can be adapted easily in agile way of development and is in line with our both patterns.

4.5 Patterns for Agile Teams for Development of Safety-Critical Systems

In this section I present a pattern that will help in choosing team members, forming a team and communication among them. This pattern has three parts in it. I urge that pattern should be adapted as whole however there is possibility of applying any one suitable section also.

4.5.1 Context

The process of building autonomous teams in agile software development is still not given the due share of importance. It is very important to carefully choose a team, assign roles to them and establish effective communication among them specifically when we are building safety-critical systems. People with different back grounds and expertise have different working norms (Stray, Moe, and Hoda, 2018). Agile is less focused on documentation so teams should make only necessary documents to ensure that risks and vulnerabilities are managed. Agile starts with minimal documentation and contracts because otherwise these documents might be wasted as changes invalidate the upfront work. It is a challenging task to have rigorous communication while keeping documentation minimal. There are many challenges to come up with good working teams that have different skill sets and establish successful communication mechanism between them.

4.5.2 Forces

- **Cross-functional Teams**: Agile suggests using cross-functional teams to keep size of team as small as possible. In safety-critical systems teams are divided and each team is focused on any one particular area hence generally it ends up with bigger number of teams.
- **Small Size of Teams**: Agile suggests creating small teams generally consisting of 10-12 people whereas, Safety-critical system's development teams are large because they need experts of every area in the team working on specific aspects of project.

- **Communication Among Teams**: Agile supports rapid development which makes it harder to have formal communication mechanisms in place, whereas safety-critical systems need formal ways of communication.
- **Multi-Tasking Teams**: Agile culture is very different from safety-critical system's development team's culture. Agile promotes that people do multi- tasking whereas safety-critical teams have dedicated people for dedicated jobs. Hence, role identification must be done with great care and some common grounds need to be defined to have a smoothly functioning team.

Problem

"What traits should an agile team have for developing safety-critical systems?"

Solution

The first point to be considered is team size. It should be kept small ideally 10-15 people on each team. Every member of the team must have prior experience of safety-critical systems in terms of development. If agile is new to them there must be a workshop or training to have them on board with the idea. As for teams using traditional approaches for development this can be entirely new way of looking at things. There must be at least one person in each team with prior experience of working with agile.

Consequences

(+) Having certain qualities in each member increases the chances of having an efficiently working team.
(-) Different cultural and work backgrounds can create issues.

Problem

"How to create an agile team best suited for safety-critical system's development?"

Solution

Create software according to its features. An ideal team should have people from all phases of development life cycle. In other words a single team should have requirement engineer, developer, tester, quality assurance person and team leader. According to the project in hand more than one person can be assigned to perform any specific task, for example there can be more than one tester on a team. The key point is everyone should be on board from day one. It should be a mutual decision to decide which features should be build first and which should be dealt later on. Some people can shuffle the roles e.g for any one particular feature a team member can be both a requirement engineer and developer, for second feature may be he is requirement engineer and tester. Roles assigned to every member of team would remain same until one feature is developed and deployed successfully. Creating one feature can take more than one iteration of SDLC (software development life cycle). Another aspect is to include people who are skilled enough to juggle their roles when needed.

Consequences

(+) More insight into ongoing development for management.
(+) Team is cross-functional hence a small team performing all activities.
(+) process is highly adaptive to updates or changes requested.
(-) The process is completely dependant on members of team, if anyone falls short there is a problem.

Problem

"How to achieve fast and reliable communication among team?"

Solution

Team should use some tool for documenting requirements and progress during development. This should be a more formal step than writing user stories, as is normally done in agile development. Since safety-critical systems need to have an extensive documentation about details of each step. Using a tool can help the team in two ways, they can communicate faster, they can put daily log of tasks assigned to them and tasks completed, this should be done on daily basis. Work assigned can be on weekly or bi-weekly basis but every member of team should log his activity for that particular day on daily basis. Project manager can create chapters based on features of system. Each chapter can have many iteration cycles, numbering or codes can be used to identify these iterations. Using codes is more efficient as it increases the efficiency of communication, code can represent feature and iteration cycle. For each feature, roles of members will remain same, but in every iteration there will be different outputs required from team. Usage of tool makes it easier to communicate, integrate and update the work of team in a time efficient manner. Also if there is any change of plans every one can be informed quickly and people can give their feedback on proposed change at collective platform. As agile proposes informal meetings with no formal documentation, this approach can stay in contact if outcome of meetings and processes adapted during development are gathered at collectively shared electronic platform. This will save the redundancy of work and people will be notifying and communicating in real time. However, any change in requirements or any kind of failure must be reported and approved by project manager before deciding on taking any counter measures. Client or User should be incorporated in meeting at start of every iteration, however once the iteration has begun it should be development team only communicating to each other.

Consequences

(+) Better collaboration and communication among team.
(+) Using a tool makes communication faster and trackable.
(+) Each member of team feels increased responsibility.
(+) Better chances of improvement as continuous communication is maintained within team.
(-)Cultural differences of people can be hindrances in communicating and understanding each other which is frequently required.

4.5.3 Known Uses

NASA switched to agile software development for their mission called Orion Program. It was a human-rated mission with great complexity. They argue that the objectives of critical missions demand a new way of developing these systems and gave them motivation to adapt agile for this particular project.

They indicated the short term interactions with deployment after every iteration and effective communication among major benefits. They used almost the similar approach as proposed in this work for teams and communication. They argue that enhanced communication and meetings benefited the project to a great extent. This enabled the work progress smooth and made it easy to deal also they were able to take suggestions for improvement and incorporate them in more effective way as compared to traditional approaches. They further reported that this method of communication provided a clear picture of daily operations to higher management and based on this fact planning of responsibilities was made easy. Also the accountability measures were taken more effectively. They highly recommended it for projects who are having issues in traditional approach to create safety-critical systems, especially in terms of communication (Smith et al., 2019).

Gupta et al (Gupta and Reddy, 2016) reported the journey of adopting agile methodologies for a project called Global Configuration Project (GCP). The team was using traditional approaches for building critical systems before this project. Here they adapted Scrum to be more specific. They particularly worked on creating agile teams, which were named as team 1, team 2 etc. Each team had people from different sections of development on board, they had tester, developer and client in every team. They created multi skilled and cross functional teams and communication was done as described by agile. They reported that this structure worked very well for them in terms of enhanced communication and shared ownership. This made whole team more responsible for their work. They used a digital tool (digital board) to document details of meetings which gave them "excellent environment for collaboration" as they quote. They further argue that they received positive feedback from customer also by switching to agile methods. They measured improvement in communication through a survey which reported positive feedback on the method.

4.6 Threats to Validity

These patterns can be considered as initial steps towards the solution of a bigger and complex problem which is reforming agile process models to suit the needs of safety-critical systems' development. Although, there are reported usage of similar patterns in industry, but my proposed patterns need validation in industrial scenario. These patterns needs validation with respect to different fields of safety-critical systems. For example to use this approach in avionics, these patterns must be validated in this particular field. Further, research is required to develop patterns that can suit specific demands of different safety-critical domains. These patterns are internally validated however, my future work will be based on external validity of these patterns.

4.7 Conclusion

This section describes a few patterns for adapting agility in the process for design and realization of safety-critical Systems. The first pattern for requirement elicitation traceability paves a way to develop safety-critical systems with a hybrid approach that finds a middle ground between traditional safety approaches and agile development principles. It is a fact that safety-critical systems cannot be developed completely with agile approaches, however, since I have proven advantages of agile process models for development I have given a method to include them in the requirement engineering phase of development for safety-critical systems.

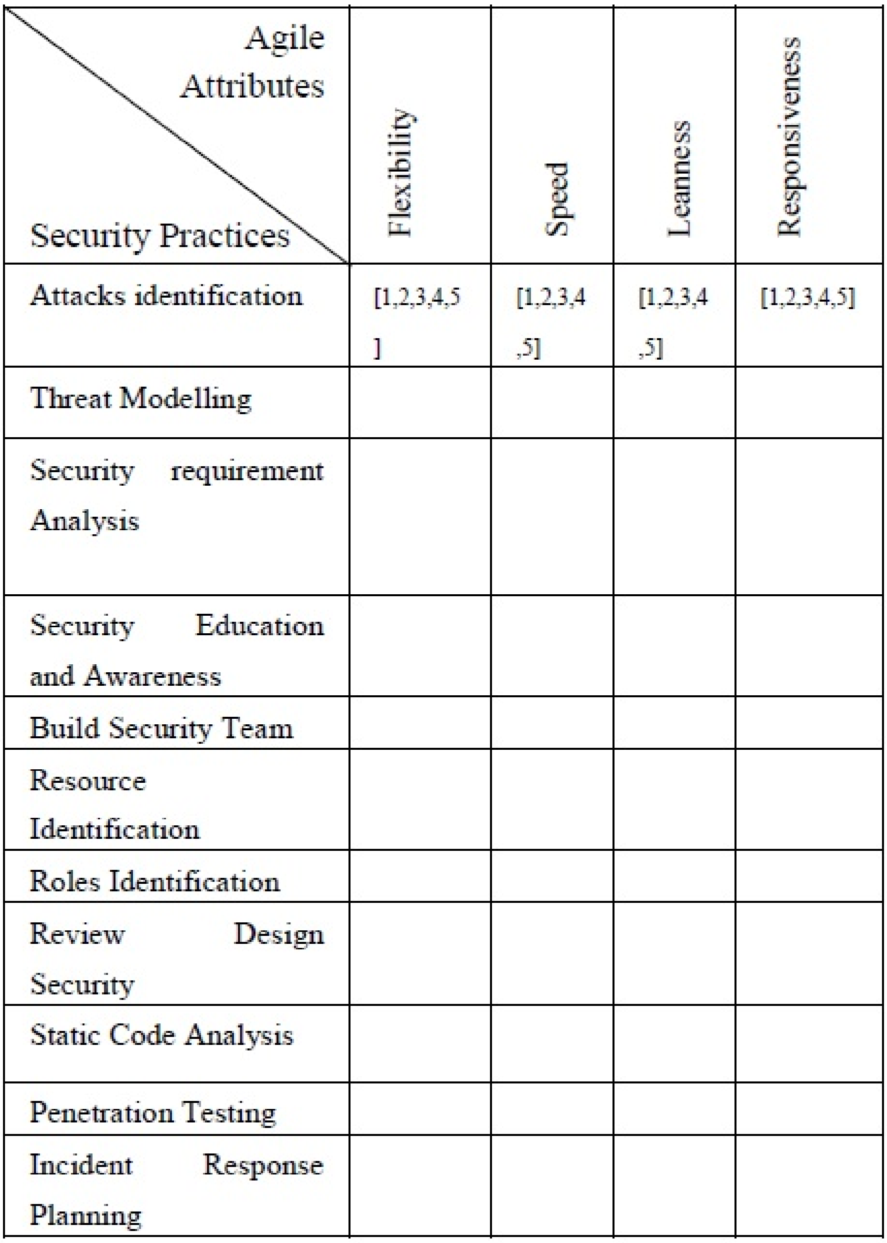
The second and third set of patterns deals with testing. Test pattern describes a way of automation of testing for safety-critical systems. Testing can be a hybrid approach of traditional testing strategies for safety-critical systems and automated tools where possible. Our proposed pattern pin points the areas and applications of automated testing in the agile iterations while keeping the rules of safety standards in tact in the whole procedure.Testing patterns in this work support the approach of test driven development for producing safety-critical systems. I have proposed the patterns that can be used with team pattern or can be used separately. Testing patterns provide a way to use the approach by keeping safety standards intact and pave a way to encourage the use of agile methods for development of safety-critical systems. They provides rigorous testing which is mandatory for safety-critical systems in a time and cost efficient way, which is the basic principle of agile. In a nutshell, it provides a middle ground to use agile approach while honouring the rules of safety standards.

Finally I have presented patterns relevant to teams which give a direction to use agile software development methodologies for safety-critical systems. They focuses on team building and communication among the team in an agile way. I propose that agile can give much faster and efficient ways of communication, better insight to projects and enhanced team collaboration by our proposed method. This can be used in collaboration with traditional software development approaches for safety-critical systems for creating a hybrid approach that will result in fast paced work and better communication.

These patterns can be used as complimentary patterns with each other or can be used separately also. This leads to a complete approach of designing iterations in agile manner with complete traceability of requirements and testing in time efficient manner through careful selection of safety requirements, test automation and team formation.

Appendix A

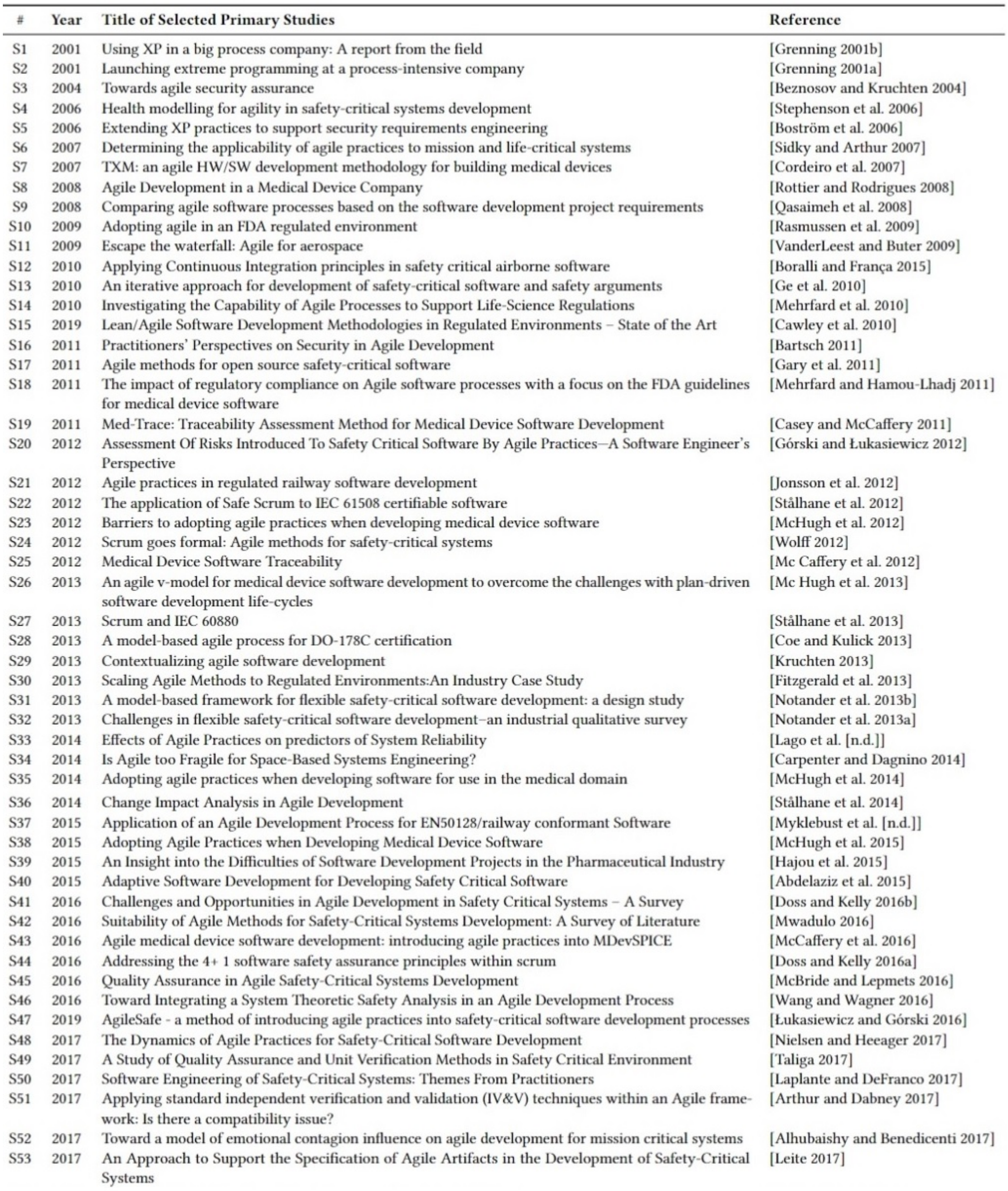
Questionnaire



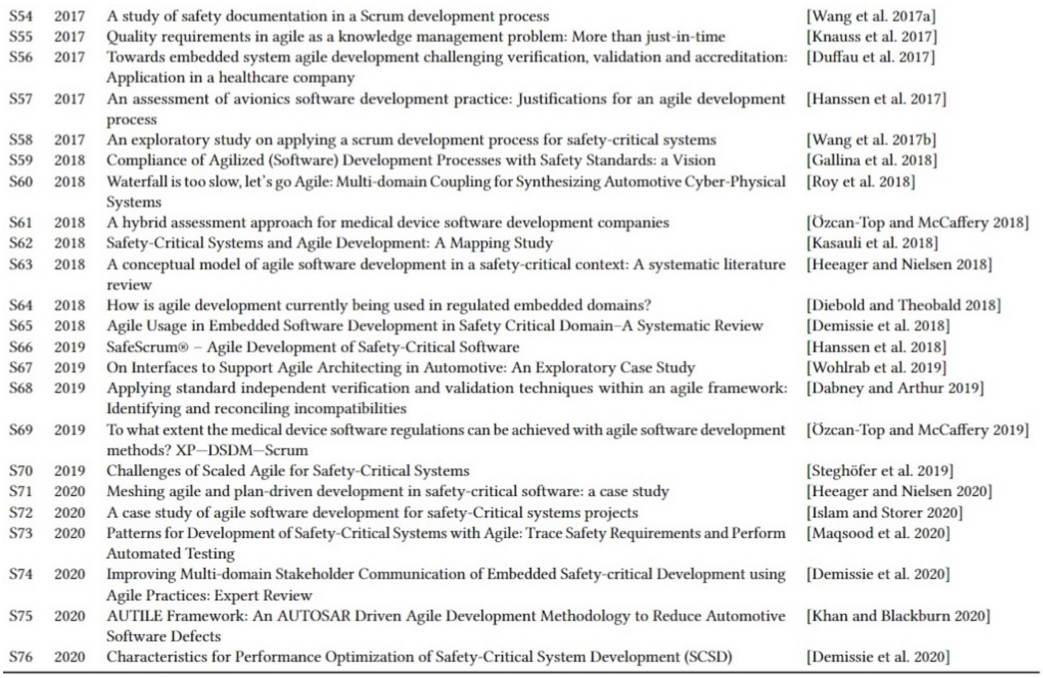
Agile Attributes / Security Practices	Flexibility	Speed	Leanness	Responsiveness
Attacks identification	[1,2,3,4,5]	[1,2,3,4,5]	[1,2,3,4,5]	[1,2,3,4,5]
Threat Modelling				
Security requirement Analysis				
Security Education and Awareness				
Build Security Team				
Resource Identification				
Roles Identification				
Review Design Security				
Static Code Analysis				
Penetration Testing				
Incident Response Planning				

Appendix B

List of Studies for SLR



#	Year	Title of Selected Primary Studies	Reference
S1	2001	Using XP in a big process company: A report from the field	[Grenning 2001b]
S2	2001	Launching extreme programming at a process-intensive company	[Grenning 2001a]
S3	2004	Towards agile security assurance	[Beznosov and Kruchten 2004]
S4	2006	Health modelling for agility in safety-critical systems development	[Stephenson et al. 2006]
S5	2006	Extending XP practices to support security requirements engineering	[Boström et al. 2006]
S6	2007	Determining the applicability of agile practices to mission and life-critical systems	[Sidky and Arthur 2007]
S7	2007	TXM: an agile HW/SW development methodology for building medical devices	[Cordeiro et al. 2007]
S8	2008	Agile Development in a Medical Device Company	[Rottier and Rodrigues 2008]
S9	2008	Comparing agile software processes based on the software development project requirements	[Qasaimeh et al. 2008]
S10	2009	Adopting agile in an FDA regulated environment	[Rasmussen et al. 2009]
S11	2009	Escape the waterfall: Agile for aerospace	[VanderLeest and Buter 2009]
S12	2010	Applying Continuous Integration principles in safety critical airborne software	[Boralli and França 2015]
S13	2010	An iterative approach for development of safety-critical software and safety arguments	[Ge et al. 2010]
S14	2010	Investigating the Capability of Agile Processes to Support Life-Science Regulations	[Mehrfard et al. 2010]
S15	2019	Lean/Agile Software Development Methodologies in Regulated Environments – State of the Art	[Cawley et al. 2010]
S16	2011	Practitioners' Perspectives on Security in Agile Development	[Bartsch 2011]
S17	2011	Agile methods for open source safety-critical software	[Gary et al. 2011]
S18	2011	The impact of regulatory compliance on Agile software processes with a focus on the FDA guidelines for medical device software	[Mehrfard and Hamou-Lhadj 2011]
S19	2011	Med-Trace: Traceability Assessment Method for Medical Device Software Development	[Casey and McCaffery 2011]
S20	2012	Assessment Of Risks Introduced To Safety Critical Software By Agile Practices—A Software Engineer's Perspective	[Górski and Łukasiewicz 2012]
S21	2012	Agile practices in regulated railway software development	[Jonsson et al. 2012]
S22	2012	The application of Safe Scrum to IEC 61508 certifiable software	[Stålhane et al. 2012]
S23	2012	Barriers to adopting agile practices when developing medical device software	[McHugh et al. 2012]
S24	2012	Scrum goes formal: Agile methods for safety-critical systems	[Wolff 2012]
S25	2012	Medical Device Software Traceability	[Mc Caffery et al. 2012]
S26	2013	An agile v-model for medical device software development to overcome the challenges with plan-driven software development life-cycles	[Mc Hugh et al. 2013]
S27	2013	Scrum and IEC 60880	[Stålhane et al. 2013]
S28	2013	A model-based agile process for DO-178C certification	[Coe and Kulick 2013]
S29	2013	Contextualizing agile software development	[Kruchten 2013]
S30	2013	Scaling Agile Methods to Regulated Environments:An Industry Case Study	[Fitzgerald et al. 2013]
S31	2013	A model-based framework for flexible safety-critical software development: a design study	[Notander et al. 2013b]
S32	2013	Challenges in flexible safety-critical software development–an industrial qualitative survey	[Notander et al. 2013a]
S33	2014	Effects of Agile Practices on predictors of System Reliability	[Lago et al. [n.d.]]
S34	2014	Is Agile too Fragile for Space-Based Systems Engineering?	[Carpenter and Dagnino 2014]
S35	2014	Adopting agile practices when developing software for use in the medical domain	[McHugh et al. 2014]
S36	2014	Change Impact Analysis in Agile Development	[Stålhane et al. 2014]
S37	2015	Application of an Agile Development Process for EN50128/railway conformant Software	[Myklebust et al. [n.d.]]
S38	2015	Adopting Agile Practices when Developing Medical Device Software	[McHugh et al. 2015]
S39	2015	An Insight into the Difficulties of Software Development Projects in the Pharmaceutical Industry	[Hajou et al. 2015]
S40	2015	Adaptive Software Development for Developing Safety Critical Software	[Abdelaziz et al. 2015]
S41	2016	Challenges and Opportunities in Agile Development in Safety Critical Systems – A Survey	[Doss and Kelly 2016b]
S42	2016	Suitability of Agile Methods for Safety-Critical Systems Development: A Survey of Literature	[Mwadulo 2016]
S43	2016	Agile medical device software development: introducing agile practices into MDevSPICE	[McCaffery et al. 2016]
S44	2016	Addressing the 4+ 1 software safety assurance principles within scrum	[Doss and Kelly 2016a]
S45	2016	Quality Assurance in Agile Safety-Critical Systems Development	[McBride and Lepmets 2016]
S46	2016	Toward Integrating a System Theoretic Safety Analysis in an Agile Development Process	[Wang and Wagner 2016]
S47	2019	AgileSafe - a method of introducing agile practices into safety-critical software development processes	[Łukasiewicz and Górski 2016]
S48	2017	The Dynamics of Agile Practices for Safety-Critical Software Development	[Nielsen and Heeager 2017]
S49	2017	A Study of Quality Assurance and Unit Verification Methods in Safety Critical Environment	[Taliga 2017]
S50	2017	Software Engineering of Safety-Critical Systems: Themes From Practitioners	[Laplante and DeFranco 2017]
S51	2017	Applying standard independent verification and validation (IV&V) techniques within an Agile framework: Is there a compatibility issue?	[Arthur and Dabney 2017]
S52	2017	Toward a model of emotional contagion influence on agile development for mission critical systems	[Alhubaishy and Benedicenti 2017]
S53	2017	An Approach to Support the Specification of Agile Artifacts in the Development of Safety-Critical Systems	[Leite 2017]



S54	2017	A study of safety documentation in a Scrum development process	[Wang et al. 2017a]
S55	2017	Quality requirements in agile as a knowledge management problem: More than just-in-time	[Knauss et al. 2017]
S56	2017	Towards embedded system agile development challenging verification, validation and accreditation: Application in a healthcare company	[Duffau et al. 2017]
S57	2017	An assessment of avionics software development practice: Justifications for an agile development process	[Hanssen et al. 2017]
S58	2017	An exploratory study on applying a scrum development process for safety-critical systems	[Wang et al. 2017b]
S59	2018	Compliance of Agilized (Software) Development Processes with Safety Standards: a Vision	[Gallina et al. 2018]
S60	2018	Waterfall is too slow, let's go Agile: Multi-domain Coupling for Synthesizing Automotive Cyber-Physical Systems	[Roy et al. 2018]
S61	2018	A hybrid assessment approach for medical device software development companies	[Özcan-Top and McCaffery 2018]
S62	2018	Safety-Critical Systems and Agile Development: A Mapping Study	[Kasauli et al. 2018]
S63	2018	A conceptual model of agile software development in a safety-critical context: A systematic literature review	[Heeager and Nielsen 2018]
S64	2018	How is agile development currently being used in regulated embedded domains?	[Diebold and Theobald 2018]
S65	2018	Agile Usage in Embedded Software Development in Safety Critical Domain–A Systematic Review	[Demissie et al. 2018]
S66	2019	SafeScrum® – Agile Development of Safety-Critical Software	[Hanssen et al. 2018]
S67	2019	On Interfaces to Support Agile Architecting in Automotive: An Exploratory Case Study	[Wohlrab et al. 2019]
S68	2019	Applying standard independent verification and validation techniques within an agile framework: Identifying and reconciling incompatibilities	[Dabney and Arthur 2019]
S69	2019	To what extent the medical device software regulations can be achieved with agile software development methods? XP—DSDM—Scrum	[Özcan-Top and McCaffery 2019]
S70	2019	Challenges of Scaled Agile for Safety-Critical Systems	[Steghöfer et al. 2019]
S71	2020	Meshing agile and plan-driven development in safety-critical software: a case study	[Heeager and Nielsen 2020]
S72	2020	A case study of agile software development for safety-Critical systems projects	[Islam and Storer 2020]
S73	2020	Patterns for Development of Safety-Critical Systems with Agile: Trace Safety Requirements and Perform Automated Testing	[Maqsood et al. 2020]
S74	2020	Improving Multi-domain Stakeholder Communication of Embedded Safety-critical Development using Agile Practices: Expert Review	[Demissie et al. 2020]
S75	2020	AUTILE Framework: An AUTOSAR Driven Agile Development Methodology to Reduce Automotive Software Defects	[Khan and Blackburn 2020]
S76	2020	Characteristics for Performance Optimization of Safety-Critical System Development (SCSD)	[Demissie et al. 2020]

Bibliography

Abdelaziz, Adil A, Yaseen El-Tahir, and Raheeg Osman (2015). “Adaptive Software Development for developing safety critical software”. In: *2015 International Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE)*. IEEE, pp. 41–46.

Agbo, Friday Joseph et al. (2021). “Application of Virtual Reality in Computer Science Education: A Systemic Review Based on Bibliometric and Content Analysis Methods”. In: *Education Sciences* 11.3, p. 142.

Alhubaishy, Abdulaziz and Luigi Benedicenti (2017). “Toward a model of emotional contagion influence on agile development for mission critical systems”. In: *2017 International Conference on High Performance Computing & Simulation (HPCS)*. IEEE, pp. 541–544.

Ali, Zulfiqar and S Bala Bhaskar (2016). “Basic statistical tools in research and data analysis”. In: *Indian journal of anaesthesia* 60.9, p. 662.

Allen, Julia H et al. (2008). *Software security engineering*. Pearson India.

Alliance, Agile (2001). *Manifesto for agile software development*.

Almog, Dani and Yaron Tsubery (2015). “How the Repository Driven Test Automation (RDTA) will make test automation more efficient, easier & maintainable”. In: *Proceedings of the 8th India Software Engineering Conference*, pp. 196–197.

Alnatheer, Ahmed, Andrew M Gravell, and David Argles (2010). “Agile security issues: an empirical study”. In: *Proceedings of the 2010 ACM-IEEE international symposium on empirical software engineering and measurement*, pp. 1–1.

Anwar, Hina and Dietmar Pfahl (2017). “Towards greener software engineering using software analytics: A systematic mapping”. In: *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, pp. 157–166.

Arkley, Paul and Steve Riddle (2005). “Overcoming the traceability benefit problem”. In: *13th IEEE International Conference on Requirements Engineering (RE’05)*. IEEE, pp. 385–389.

Arthur, James D and James B Dabney (2017). "Applying standard independent verification and validation (IV&V) techniques within an Agile framework: Is there a compatibility issue?" In: *2017 Annual IEEE International Systems Conference (SysCon)*. IEEE, pp. 1–5.

Ashraf, Sara and Shabib Aftab (2017). "IScrum: An Improved Scrum Process Model." In: *International Journal of Modern Education & Computer Science* 9.8.

Avizienis, Algirdas et al. (2004). "Basic concepts and taxonomy of dependable and secure computing". In: *IEEE transactions on dependable and secure computing* 1.1, pp. 11–33.

Axelsson, Jakob et al. (2016). "Notes on agile and safety-critical development". In: *ACM SIGSOFT Software Engineering Notes* 41.2, pp. 23–26.

Ayalew, Tigist, Tigist Kidane, and Bengt Carlsson (2013). "Identification and evaluation of security activities in agile projects". In: *Nordic Conference on Secure IT Systems*. Springer, pp. 139–153.

Bandara, Wasana et al. (2015). "Achieving rigor in literature reviews: Insights from qualitative data analysis and tool-support". In: *Communications of the Association for Information Systems* 37.1, p. 8.

Beck, Kent et al. (2001a). "Manifesto for agile software development". In.

– (2001b). *The agile manifesto*.

Boehm, Barry (2002). "Get ready for agile methods, with care". In: *Computer* 35.1, pp. 64–69.

Boström, Gustav et al. (2006). "Extending XP practices to support security requirements engineering". In: *Proceedings of the 2006 international workshop on Software engineering for secure systems*, pp. 11–18.

Brereton, Pearl et al. (2007). "Lessons from applying the systematic literature review process within the software engineering domain". In: *Journal of systems and software* 80.4, pp. 571–583.

Carpenter, Scott E and Aldo Dagnino (2014). "Is agile too fragile for space-based systems engineering?" In: *2014 IEEE International Conference on Space Mission Challenges for Information Technology*. IEEE, pp. 38–45.

Chambers, John (2008). *Software for data analysis: programming with R*. Springer Science & Business Media.

Chen, Jiyao et al. (2015). "The relationship between team autonomy and new product development performance under different levels of technological turbulence". In: *Journal of Operations Management* 33, pp. 83–96.

Cleland-Huang, Jane (2006). "Just enough requirements traceability". In: *30th Annual International Computer Software and Applications Conference (COMPSAC'06)*. Vol. 1. IEEE, pp. 41–42.

Cleland-Huang, Jane et al. (2011). "Grand challenges, benchmarks, and TraceLab: developing infrastructure for the software traceability research community". In: *Proceedings of the 6th international workshop on traceability in emerging forms of software engineering*, pp. 17–23.

Cleland-Huang, Jane et al. (2014). "Software traceability: trends and future directions". In: *Proceedings of the on Future of Software Engineering*, pp. 55–69.

Coe, David J and Jeffrey H Kulick (2013). "A model-based agile process for DO-178C certification". In: *Proceedings of the International Conference on Software Engineering Research and Practice (SERP)*. The Steering Committee of The World Congress in Computer Science, Computer . . ., p. 1.

Cohen, David, Mikael Lindvall, and Patricia Costa (2003). "Agile software development". In: *DACS SOAR Report* 11, p. 2003.

Cohn, Mike (2010). *Succeeding with agile: software development using Scrum*. Pearson Education.

Cooper, Harris M (1988). "Organizing knowledge syntheses: A taxonomy of literature reviews". In: *Knowledge in society* 1.1, pp. 104–126.

Cordeiro, Lucas et al. (2007). "TXM: an agile HW/SW development methodology for building medical devices". In: *ACM SIGSOFT Software Engineering Notes* 32.6, 4–es.

Council, National Research et al. (n.d.). "Committee for Advancing Software-Intensive Systems Producibility.(2010)". In: *Critical Code: Software Producibility for Defense* ().

Dabney, James B and James D Arthur (2019). "Applying standard independent verification and validation techniques within an agile framework: Identifying and reconciling incompatibilities". In: *Systems Engineering* 22.4, pp. 348–360.

Damm, Lars-Ola, Lars Lundberg, and David Olsson (2005). "Introducing test automation and test-driven development: An experience report". In: *Electronic notes in theoretical computer science* 116, pp. 3–15.

Doss, Osama and Tim Kelly (2016). "Addressing the 4+ 1 software safety assurance principles within scrum". In: *Proceedings of the Scientific Workshop Proceedings of XP2016*, pp. 1–5.

Duffau, Clément, Bartosz Grabiec, and Mireille Blay-Fornarino (2017). "Towards embedded system agile development challenging verification, validation and accreditation: Application in a healthcare company". In: *2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. IEEE, pp. 82–85.

Dustin, Elfriede, Jeff Rashka, and John Paul (1999). *Automated software testing: introduction, management, and performance*. Addison-Wesley Professional.

Dybå, Tore, Barbara Kitchenham, and Magne Jørgensen (Feb. 2005). “Evidence-Based Software Engineering for Practitioners”. In: *Software, IEEE* 22, pp. 58 –65. DOI: 10.1109/MS.2005.6.

Easterbrook, Steve et al. (2008). “Selecting empirical methods for software engineering research”. In: *Guide to advanced empirical software engineering*. Springer, pp. 285–311.

Er, Thomas (2005). “Service-oriented architecture: concepts, technology, and design”. In: *Book Service-Oriented Architecture: Concepts, Technology, and Design Prentice Hall PTR Upper Saddle River, NJ, USA ISBN* 131858580.

Fager, Edward W (1957). “Determination and analysis of recurrent groups”. In: *Ecology* 38.4, pp. 586–595.

Field, Andy P (2014). “K endall’s Coefficient of Concordance”. In: *Wiley StatsRef: Statistics Reference Online*.

Freude, René and Alexander Königs (2003). “Tool integration with consistency relations and their visualization”. In: *Proc. Workshop on Tool-Integration in System Development (TIS 2003)*. Citeseer, pp. 6–10.

Gallina, Barbara, Faiz Ul Muram, and Julieth Patricia Castellanos Ardila (2018). “Compliance of agilized (Software) development processes with safety standards: a vision”. In: *Proceedings of the 19th International Conference on Agile Software Development: Companion*, pp. 1–6.

Gary, Kevin et al. (2011). “Agile methods for open source safety-critical software”. In: *Software: Practice and Experience* 41.9, pp. 945–962.

Gayer, Sebastian et al. (2016). “Lightweight traceability for the agile architect”. In: *Computer* 49.5, pp. 64–71.

Gearhart, Amanda et al. (2013). “Use of Kendall’s coefficient of concordance to assess agreement among observers of very high resolution imagery”. In: *Geocarto International* 28.6, pp. 517–526.

Ghazarian, Arbi (2008). “Traceability patterns: an approach to requirement-component traceability in agile software development”. In: *Proceedings of the 8th conference on Applied computer scince*. World Scientific, Engineering Academy, and Society (WSEAS), pp. 236–241.

Górski, Janusz and Katarzyna Łukasiewicz (2012). “Assessment Of Risks Introduced To Safety Critical Software By Agile Practices—A Software Engineer’s Perspective”. In: *Computer Science* 13.4, p. 165.

Gotel, Orlena CZ and CW Finkelstein (1994). "An analysis of the requirements traceability problem". In: *Proceedings of IEEE International Conference on Requirements Engineering*. IEEE, pp. 94–101.

Grenning, James (2001a). "Launching extreme programming at a process-intensive company". In: *IEEE Software* 18.6, pp. 27–33.

– (2001b). "Using XP in a big process company: A report from the field". In: *XP Agile Universe*. Citeseer.

Gupta, Rajeev Kumar and Prabhulinga Manik Reddy (2016). "Adapting agile in a globally distributed software development". In: *2016 49th Hawaii International Conference on System Sciences (HICSS)*. IEEE, pp. 5360–5367.

Hajou, A, RS Batenburg, and S Jansen (2015). "An insight into the difficulties of software development projects in the pharmaceutical industry". In: *Lecture Notes on Software Engineering* 3.4.

Hause, Matthew et al. (2010). "Testing safety critical systems with SysML/UML". In: *2010 15th IEEE International Conference on Engineering of Complex Computer Systems*. IEEE, pp. 325–330.

Heeager, Lise Tordrup and Peter Axel Nielsen (2018). "A conceptual model of agile software development in a safety-critical context: A systematic literature review". In: *Information and Software Technology* 103, pp. 22–39.

Hoda, Rashina, James Noble, and Stuart Marshall (2012a). "Documentation strategies on agile software development projects". In: *International Journal of Agile and Extreme Software Development* 1.1, pp. 23–37.

– (2012b). "Self-organizing roles on agile software development teams". In: *IEEE Transactions on Software Engineering* 39.3, pp. 422–444.

Holcombe, Mike, Florentin Ipate, and Andreas Grondoudis (1995). "Complete functional testing of safety critical systems". In: *IFAC Proceedings Volumes* 28.25, pp. 199–204.

Höller, Andrea et al. (2014). "FIES: a fault injection framework for the evaluation of self-tests for COTS-based safety-critical systems". In: *2014 15th International Microprocessor Test and Verification Workshop*. IEEE, pp. 105–110.

Islam, Gibrail and Tim Storer (2020). "A case study of agile software development for safety-Critical systems projects". In: *Reliability Engineering & System Safety* 200, p. 106954.

Ivarsson, Martin and Tony Gorschek (2011). "A method for evaluating rigor and industrial relevance of technology evaluations". In: *Empirical Software Engineering* 16.3, pp. 365–395.

Jacobson, Ivar (2002). "A resounding" Yes" to agile processes-But also to more". In: *Cutter IT Journal* 15.1, pp. 18–24.

Jakobsson, Marcus (2009). *Implementing traceability in agile software development*. Department of Computer Science, Lund University.

Janzen, David and Hossein Saiedian (2005). "Test-driven development concepts, taxonomy, and future direction". In: *Computer* 38.9, pp. 43–50.

Jee, Eunkyoung et al. (2010). "Automated test coverage measurement for reactor protection system software implemented in function block diagram". In: *International Conference on Computer Safety, Reliability, and Security*. Springer, pp. 223–236.

Karhu, Katja et al. (2009). "Empirical observations on software testing automation". In: *2009 International Conference on Software Testing Verification and Validation*. IEEE, pp. 201–209.

Kasauli, Rashidah et al. (2018). "Safety-critical systems and agile development: a mapping study". In: *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, pp. 470–477.

Kendall, Maurice George (1948). "Rank correlation methods." In.

Keramati, Hossein and Seyed-Hassan Mirian-Hosseinabadi (2008). "Integrating software development security activities with agile methodologies". In: *2008 IEEE/ACS International Conference on Computer Systems and Applications*. IEEE, pp. 749–754.

Kitchenham, Barbara, Stuart Charters, et al. (2007). "Guidelines for performing systematic literature reviews in software engineering version 2.3". In: *Engineering* 45.4ve, p. 1051.

Kitchenham, Barbara et al. (2009). "Systematic literature reviews in software engineering–a systematic literature review". In: *Information and software technology* 51.1, pp. 7–15.

Kitchenham, Barbara A et al. (2002). "Preliminary guidelines for empirical research in software engineering". In: *IEEE Transactions on software engineering* 28.8, pp. 721–734.

Krosnick, Jon A, James D Wright, Peter V Marsden, et al. (2009). "Handbook of survey research". In.

Laplante, Phillip A and Joanna F DeFranco (2017). "Software engineering of safety-critical systems: Themes from practitioners". In: *IEEE Transactions on Reliability* 66.3, pp. 825–836.

Lee, Christopher, Luigi Guadagno, and Xiaoping Jia (2003). "An agile approach to capturing requirements and traceability". In: *Proceedings of the 2nd International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE 2003)*. Vol. 20.

Legendre, Pierre (2005). "Species associations: the Kendall coefficient of concordance revisited". In: *Journal of agricultural, biological, and environmental statistics* 10.2, pp. 226–245.

Leite, Ana Isabella Muniz (2017). "An Approach to Support the Specification of Agile Artifacts in the Development of Safety-Critical Systems". In: *2017 IEEE 25th International Requirements Engineering Conference (RE)*. IEEE, pp. 526–531.

Likert, Rensis (1932). "A technique for the measurement of attitudes." In: *Archives of psychology*.

Marozzi, Marco (2014). "Testing for concordance between several criteria". In: *Journal of Statistical Computation and Simulation* 84.9, pp. 1843–1850.

Mc Hugh, Martin et al. (2013). "An agile v-model for medical device software development to overcome the challenges with plan-driven software development lifecycles". In: *2013 5th International Workshop on Software Engineering in Health Care (SEHC)*. IEEE, pp. 12–19.

McBride, Tom and Marion Lepmets (2016). "Quality Assurance in Agile Safety-Critical Systems Development". In: *2016 10th International Conference on the Quality of Information and Communications Technology (QUATIC)*. IEEE, pp. 44–51.

McHugh, Martin, Fergal McCaffery, and Valentine Casey (2012). "Barriers to adopting agile practices when developing medical device software". In: *International Conference on Software Process Improvement and Capability Determination*. Springer, pp. 141–147.

– (2014). "Adopting agile practices when developing software for use in the medical domain". In: *Journal of Software: Evolution and Process* 26.5, pp. 504–512.

McHugh, Martin, Fergal McCaffery, and Garret Coady (2015). "Adopting Agile Practices when Developing Medical Device Software". In: *J Comput Eng Inf Technol 4: 2. doi: http://dx. doi. org/10. 4172* 2324.9307, p. 2.

McHugh, Orla, Kieran Conboy, and Michael Lang (2011). "Agile practices: The impact on trust in software project teams". In: *Ieee Software* 29.3, pp. 71–76.

Michael, Howard and Lipner Steve (2006). *The Security Development Lifecycle: SDL: A Process for Developing Demonstrably More Secure Software*.

Mohammadi, Seyed Abolghasem and BM Prasanna (2003). "Analysis of genetic diversity in crop plants—salient statistical tools and considerations". In: *Crop science* 43.4, pp. 1235–1248.

Moyon, Fabiola et al. (2018). "Towards continuous security compliance in agile software development at scale". In: *2018 IEEE/ACM 4th International Workshop on Rapid Continuous Software Engineering (RCoSE)*. IEEE, pp. 31–34.

Myklebust, T, T Stålhane, and N Lyngby (n.d.). "Application of an Agile Development Process for EN50128/railway con-formant Software". In: ().

Nardi, PM (2014). "Doing survey research: a guide to quantitative methods 3rd ed. Ridley, D., 2012". In: *The literature review: a step-by-step guide for students*.

Oppenheim, Abraham Naftali (2000). *Questionnaire design, interviewing and attitude measurement*. Bloomsbury Publishing.

Özcan-Top, Özden and Fergal McCaffery (2018). "A hybrid assessment approach for medical device software development companies". In: *Journal of Software: Evolution and Process* 30.7, e1929.

– (2019). "To what extent the medical device software regulations can be achieved with agile software development methods? XP—DSDM—Scrum." In: *Journal of Supercomputing* 75.8.

Pamela, LA, B Settle, and RD Irwm (1995). "The Survey Research Handbook. Guidelines and strategies for conducting a survey". In: *Chicago University*.

Petersen, Kai and Cigdem Gencel (2013). "Worldviews, research methods, and their relationship to validity in empirical software engineering research". In: *2013 joint conference of the 23rd international workshop on software measurement and the 8th international conference on software process and product measurement*. IEEE, pp. 81–89.

Petersen, Kai, Sairam Vakkalanka, and Ludwik Kuzniarz (2015). "Guidelines for conducting systematic mapping studies in software engineering: An update". In: *Information and Software Technology* 64, pp. 1–18.

Qumer, Asif and Brian Henderson-Sellers (2008). "An evaluation of the degree of agility in six agile methods and its applicability for method engineering". In: *Information and software technology* 50.4, pp. 280–295.

Rasmussen, J (2003). "Introducing XP into greenfield projects: Lessons learned". In: *Ieee Software* 20.3, pp. 21–28.

Rasmussen, Rod et al. (2009). "Adopting agile in an FDA regulated environment". In: *2009 Agile Conference*. IEEE, pp. 151–155.

Rottier, Pieter Adriaan and Victor Rodrigues (2008). "Agile development in a medical device company". In: *Agile 2008 Conference*. IEEE, pp. 218–223.

Roy, Debayan et al. (2018). "Waterfall is too slow, let's go Agile: multi-domain coupling for synthesizing automotive cyber-physical systems". In: *2018*

IEEE/ACM International Conference on Computer-Aided Design (ICCAD). IEEE, pp. 1–7.

Sabir, Fatima et al. (2019). "A systematic literature review on the detection of smells and their evolution in object-oriented and service-oriented systems". In: *Software: Practice and Experience* 49.1, pp. 3–39.

SC, RTCA (2011). "205/EUROCAE WG-71. Software Considerations in Airborne Systems and Equipment Certification". In: *No. RTCA DO-178C, RTCA Inc* 1140.

Schwaber, Ken and Mike Beedle (2002). *Agile software development with Scrum*. Vol. 1. Prentice Hall Upper Saddle River.

Smith, Justin et al. (2019). "Agile Approach to Assuring the Safety-Critical Embedded Software for NASA's Orion Spacecraft". In: *2019 IEEE Aerospace Conference*. IEEE, pp. 1–10.

Sousa Santos, Ismayle de et al. (2017). "Test case design for context-aware applications: Are we there yet?" In: *Information and Software Technology* 88, pp. 1–16.

Stålhane, Tor, Vikash Katta, and Thor Myklebust (2013). "Scrum and IEC 60880". In: *Enlarged Halden Reactor Project meeting, Storefjell, Norway*.

– (2014). "Change Impact Analysis in Agile Development". In: *EHPG Røros*.

Stålhane-IDI, Tor (n.d.). "Safety standards and Scrum–A synopsis of three standards". In: *Nbl. SintefNo*.

Stavru, Stavros (2014). "A critical examination of recent industrial surveys on agile method usage". In: *Journal of Systems and Software* 94, pp. 87–97.

Stettina, Christoph Johann and Werner Heijstek (2011). "Necessary and neglected? An empirical study of internal documentation in agile software development teams". In: *Proceedings of the 29th ACM international conference on Design of communication*, pp. 159–166.

Stray, Viktoria, Nils Brede Moe, and Rashina Hoda (2018). "Autonomous agile teams: challenges and future directions for research". In: *Proceedings of the 19th International Conference on Agile Software Development: Companion*, pp. 1–5.

Taliga, Miklos (2017). "A Study of Quality Assurance and Unit Verification Methods in Safety Critical Environment". In: *International Journal of Electrical and Information Engineering* 9.11, pp. 2407–2411.

Templier, Mathieu and Guy Paré (2015). "A framework for guiding and evaluating literature reviews". In: *Communications of the Association for Information Systems* 37.1, p. 6.

Thawaba, Abdulaziz Ahmed et al. (2020). “Characteristics for Performance Optimization of Safety-Critical System Development (SCSD)”. In: *Journal of Advanced Computational Intelligence and Intelligent Informatics* 24.2, pp. 232–242.

Tracey, Nigel James (2000). “A search-based automated test-data generation framework for safety-critical software”. PhD thesis. Citeseer.

Tyagi, Sulabh et al. (2018). “Development of Reusable Hybrid Test Automation Framework for Web Based Scrum Projects”. In: *Journal of Applied Science and Engineering* 21.3, 455462.

VanderLeest, Steven H and Andrew Buter (2009). “Escape the waterfall: Agile for aerospace”. In: *2009 IEEE/AIAA 28th Digital Avionics Systems Conference*. IEEE, pp. 6–D.

Wang, Yang, Ivan Bogicevic, and Stefan Wagner (2017). “A study of safety documentation in a Scrum development process”. In: *Proceedings of the XP2017 Scientific Workshops*, pp. 1–5.

Wang, Yang, Jasmin Ramadani, and Stefan Wagner (2017). “An exploratory study on applying a scrum development process for safety-critical systems”. In: *International Conference on Product-Focused Software Process Improvement*. Springer, pp. 324–340.

Williams, Laurie, E Michael Maximilien, and Mladen Vouk (2003). “Test-driven development as a defect-reduction practice”. In: *14th International Symposium on Software Reliability Engineering, 2003. ISSRE 2003.* IEEE, pp. 34–45.

Wirfs-Brock, Rebecca, Joseph Yoder, and Eduardo Guerra (2015). “Patterns to develop and evolve architecture during an agile software project”. In: *Proceedings of the 22nd Conference on Pattern Languages of Programs*, pp. 1–18.

Wohlin, Claes, Martin Höst, and Kennet Henningsson (2003). “Empirical research methods in software engineering”. In: *Empirical methods and studies in software engineering*. Springer, pp. 7–23.

Wohlrab, Rebekka et al. (2019). “On interfaces to support agile architecting in automotive: an exploratory case study”. In: *2019 IEEE International Conference on Software Architecture (ICSA)*. IEEE, pp. 161–170.

Wolfswinkel, Joost F, Elfi Furtmueller, and Celeste PM Wilderom (2013). “Using grounded theory as a method for rigorously reviewing literature”. In: *European journal of information systems* 22.1, pp. 45–55.

Zhang, Zheying et al. (2010). “Towards lightweight requirements documentation”. In: *Journal of Software Engineering and Applications* 3.09, p. 882.

Zimmermann, Fabian et al. (2009). "Risk-based statistical testing: A refinement-based approach to the reliability analysis of safety-critical systems". In: *12th European Workshop on Dependable Computing, EWDC 2009*, 8–pages.