I.J. Information Technology and Computer Science, **2022, 5, 44-57**
Published Online on October 8, 2022 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2022.05.04

Applying Scrum Development on Safety Critical Systems

Mustafizur Rahman, Shusmita Islam, Rubiyet Fardous, Lamisa Yesmin
Department Of Computer Science, American International University-Bangladesh, Dhaka, Bangladesh
E-mail: mrahaman59@yahoo.com, shusmitashoron@gmail.com, rubiyetfardous@gmail.com, lamisa.lorin@gmail.com

Dip Nandi
Faculty of Science and Technology, American International University-Bangladesh, Dhaka, Bangladesh
E-mail: dip.nandi@aiub.edu

Received: 23 January 2022; Revised: 28 April 2022; Accepted: 19 July 2022; Published: 8 October 2022

Abstract: Scaled agile approaches are increasingly being used by automotive businesses to cope with the complexity of their organizations and products. The development of automotive systems necessitates the use of safe procedures. SafeScrum® is a real example of how agile approaches may be used in the creation of high-reliability systems on a small scale. A framework like SAFe or LeSS does not facilitate the creation of safety-critical systems in large-scale contexts from the start. User stories are a wonderful approach to convey flexible demands, the lifecycle is iterative, and testing is the initial stage in the development process. Scrum plus extra XP approaches may be used to build high-reliability software and certification by the IEC 61508 standard is required for the software. This adds a slew of new needs to the workflow. Scrum's quality assurance measures proved to be inadequate in a recent industry situation. Our study's overarching goal is to provide light on the Scrum development process so that it may be improved for use with life-or-death systems. Our study of the business world was a mixed-methods affair. The findings demonstrated that although Scrum is helpful in ensuring the security of each release, it is less nimble in other respects. The difficulties of prioritization, communication, time constraints, and preparing for and accepting new safety standards were all discussed. In addition, we have had some helpful feedback from the business world, but the generality issue arising from this particular setting has yet to be addressed.

Index Terms: Safety-Critical System, Scrum, SafeScrum, IEC61508, Agile, Software Development.

1. Introduction

A growing quantity and complexity of safety-critical embeddable systems are becoming more commonplace in our daily lives. In the years after 2001, Agile software development for safety-critical goods has been documented in a research paper, and several books and articles have been written on it. A literature review, on the other hand, is recommended since there is no obvious accumulation of information on the subject; consequently, we believe it is necessary. Over the last decade, the software industry has seen a significant utilization of agile software development methods is on the rise, particularly Scrum variations, which are often used in conjunction with extreme programming approaches. The fact that quality assurance is embedded into the process rather than being documented afterwards is one of the many advantages of Scrum and other approaches. This final duty has traditionally been carried out by line groups participating in development based on plans. It is essential for a project to be visible and often reviewed in order to be successful under the Scrum framework. The most recent and up-to-date information is utilized to re-plan and improve the project. There may be a retrospective as part of the sprint review, concludes each brief period of work, known as a sprint. Retrospective meetings are also held at the end of each sprint to address quality concerns with both the product being developed and the development process itself. The constant interaction with the client or the issue owner is also emphasized heavily in Scrum development practices. This is important in order to guarantee that the system's functionality as well as its overall quality satisfy the needs and expectations of the users. Shortly said, Scrum may be thought of as a self-sustaining planning, development, and quality assurance process that is combined and self-sustaining, although lacking traceability. The remaining sections of the paper are structured as follows. Section 2 describes what safety critical systems are, and Section 3 discusses the difficulties that safety critical systems face. Section 2 is divided into two sections. The process of developing software for safety-critical systems is described in Section 4. Section 5 concludes with a summary of the data collection method and findings of the review of the literature.

2. Literature Review

2.1. Safety Critical Systems

Computers are becoming more and more widespread, and they are becoming an integral part of our everyday lives. In so many ways, they make it hard to imagine contemporary society without them. That being said, the widespread use of these technologies necessitates a high degree of confidence, which necessitates the need for technologies that are easily available and reliable as well as trustworthy and safe [1]. The use of computers in a variety of life-sustaining processes, such as medical treatment, biological transportation, aircraft and vehicle systems, and systems with a high level of intensity, is common. Medical procedures, human transportation, aerospace and military systems, and increased energy management are just a few of the numerous applications that rely on cloud servers. Safety-critical systems are a major source of worry because ineffective regulation of these systems may have serious consequences for the environment, property, and persons [2, 3], as well as companies, the economy, and citizens' and society's quality of life.

Although many applications have historically been regarded safety-critical, the definition must continue to broaden as computer systems are introduced into more and more sectors that have an influence on our lives. Safety-critical systems might have a variety of meanings, but the core principle is sound. The consequences of failure are the primary concern here, both philosophically and practically. In the event of a system's failure, the system is said to be "safety critical". When we can depend on a system to keep us safe, we may say that it is safe. There are repercussions to this idea that are highlighted in terms of the systems that need to be considered safety-critical. Many important systems are being developed and deployed as software-intensive systems become more common. Increased computer control of safety essential systems is a goal of ours as we seek to improve output as well as performance and cost efficiency. Many safety-critical systems (such as those in the chemical and pesticide industries) may grow in size, complexity, and the potential for catastrophic failure thanks to centralized processing management. Operators and users may make important decisions based on information provided by software even if the program does not directly operate the safety-critical hardware (e.g., air traffic control or medical information such as blood bank records, organ donor information). Increasing reliance on software-intensive systems has made us more dependent on the systems operating reliably as a result. In the future, safety-critical systems will be increasingly widely used and influential. There will be a need for considerable breakthroughs in the areas of configuration, structures, proof of identity, and methods if safety critical systems are to be developed in sufficient numbers.

2.2. Types of Systems that should be Viewed as Safety-Critical

Systems that may result in catastrophic effects if they fail are deemed safety-critical. The term "safety-critical" refers to a system whose proper functioning is dependent on its existence. As a result of this approach, the following-level categories of systems that are really vital are examined.

A. Traditional System

Safety-critical systems have typically been associated with healthcare, civilian aviation, power plants, and weaponry. A failure to address these issues may put people's lives in danger, ruin valuable equipment, and cause a host of other problems. The use of computers in healthcare is significantly more widespread than most people realize. Microprocessor-controlled insulin injections are not a new idea. It's not well known that pacemakers are essentially computers. The widespread use of computers in operations is almost unknown outside of specialized fields like orthopedics and neurosurgery. Automated devices are making gains in operations including major surgery, back surgery, and ophthalmologic surgery. Individuals in all three of these situations have received major therapy thanks to the use of computer-controlled automated devices, which have taken the place of traditional surgical equipment. The Boeing 777 is referred to be "the most high-tech aircraft on Earth" by the manufacturer. There have been several technical advancements, notably safety-critical computer systems, that have benefited the aircraft.

B. Non-Traditional System

Management and academics working with particular systems must take into account the wide-ranging possibilities of the safety-critical system idea. Consider how many new technologies are capable of having significant ramifications in the proper setting, and how these systems should probably be regarded as safety-critical. A commercial plane's demise will very certainly lead to the annihilation of humanity. People's lives might be put in jeopardy if a telecommunications network goes down. However, if the 911 system goes down for an extended period of time, there will very certainly be fatalities. The emergency 911 service is a vital piece of infrastructure that can't be done without. A few more examples are financial and banking systems, electricity production and transmission as well as telecommunications and the water management system If a computer fails in any one of these areas, it may cause an outage that affects all of these applications. In certain cases, the stoppage is so severe that it is impossible to continue. If water or electricity are shut off in large regions, health and safety are at danger. Disruptions to major modes of transportation, such as rail and trucking, would have a similar effect on the distribution of food and energy. Computer systems that support critical infrastructure should be classified as critical.

C. System Design and Manufacturing

Typical operating systems discussed in earlier sections are safety-critical systems. If these mechanisms fail, there is an immediate threat. Other software programs used in the design and construction of other systems may still fail with serious repercussions. Compilers, for example, are regarded safety-critical if the product they support is likewise considered safety-critical. Non-computer artifacts can't be made without computer systems. MSC Corporation's NASTRAN structural analysis system is used in a broad range of industries [4]. Structural engineers take its analysis at its value and apply it to help with structural design autonomously. A shoddy job might lead to a shoddy building if it's done appropriately. As a consequence, even if the final product may be a building or a bridge, the design system software used throughout the design process is important to the safety of the final product.

D. Information System Security

It is clearly clear that data security breaches are a major and growing problem. The consequences of attacks on both business networks might be dire. Internet security breaches are a major worry for network users because of the increased use of the World Wide Web for connectivity services. In spite of the dangers posed by the Internet, network providers pose a larger menace. Through the use of private networks managed by financial institutions, money may be transmitted both locally and worldwide. Secure networks are widely used to monitor and evaluate transportation systems. Private networks may be breached, causing financial losses or disrupting transportation services if the attack is successful. Because of the seriousness of the consequences of security breaches, even technologies that just convey information should be deemed safety-critical due to the high potential for loss.

2.3. Formal Framework and its Advantage

Real-time software must be functionally accurate and timely. These "hard" real-time limits should be taken into consideration: An alarm must be sent if a nuclear reactor's core temperature reaches a threshold. When a train approaches an intersection, car and pedestrian traffic must be stopped; if the computer driving a robot does not tell it to stop or turn in time, the robot may crash with another item. Modeling, defining, and developing real-time systems is difficult [5-11]. In the past, system software components have proven more difficult to repair than mechanical or other physical components. Small systems have an uncontrollable amount of test cases [12]. It's not resilient (little failures may have severe repercussions) and it's difficult (even simple modules need documentation) (small errors have major consequences). That doesn't imply the software used to operate real-time systems are undesirable. Many companies leverage existing design principles to increase product reliability. Software becomes more complex as demands on a computer's resources increase. Traditional servo-control systems could be isolated tested, while current, in order to validate complex software controllers, it is more challenging. Safety-critical systems that operate in real time, formal, mathematically exact methodologies are recommended. Many pragmatic software developers have struggled to realize this idea. But why do "theorists" need a formal framework? Among the advantages are:

1) There are several ambiguities and contradictions that might be found while formalizing informal requirements.
2) For example, this formal paradigm might lead to hierarchical, semi-automated (or perhaps fully automated) system development methodologies.
3) Mathematical methods may be used to validate the validity of the formal model (throughout the course of arduous and time-consuming experimentation with each individual situation).
4) More confidence may be placed in the performance of a subsystem that has been explicitly verified.
5) Comparing and testing different designs is possible.

2.4. Problems with Safety Critical Systems

For example, the SCS research found multiple instances when equipment failed due to misconceptions or a lack of clearly expressed demands, ultimately resulting in environmental harm, injuries, and even fatalities [13, 14]. Companies in charge of the SCS involved in an accident suffer greatly as a result. When it comes to complicated technological systems, there are often several explanations for an occurrence. As SCS gets more complex, governments and international organizations are strengthening the requirements and broadening the scope of sustainability reporting and associated procedures [15]. The safety certification program relies heavily on the requirements specifications and the associated requirements engineering methods [16], both in terms of methodology compatibility and quality assurance for safety [4, 17]. Ariane V's inability to take off [2] and the loss of the Mars Polar Lander [15] and the Mars Climate Orbiter [14] are just a few of the well-known disasters [18]. Neumann provides several examples throughout the book [19]. A few lesser-known cases are utilized to demonstrate this problem's extent rather than repeating the intricacies of previous failures.

1) Design assistance systems, such as discrete analysis programs, are an excellent illustration of what may go wrong. These applications are widely used in industrial design, particularly architecture. NRC Information Notice 96-29 was sent to all nuclear power plant operators and permit holders in May 1996 [1]. However, it was not known at the time how these programs were used in nuclear power plant safety studies, which led to the Commission's conclusion that 150 errors had been documented. Notification recipients were asked to

review the information to determine whether it was relevant. Self-evident consequences follow.

2) The Sizewell B nuclear power plant's principal protective system had a different kind of malfunction in the United Kingdom. This system, which was constructed in software, must have no more than 10 failures per demand in order to meet the dependability requirements. After everything was said and done, the system contained over 650 microchips and 1,200 wires, while the program itself had over 70,000 lines [18]. Only 48% of the test cases were successful when the system was put to the test [13]. Even though the system was said to have failed another 22% of the tests, it was difficult to know for sure whether any of them had been successfully completed. Safety-critical: When anything goes wrong, this mechanism will automatically shut down the reactor. When there are so many computers, code sources, and validation records, it's difficult to ensure the stability of a system.
3) The London Ambulance Service replaced a manual scheduling system on October 26, 1992 with a computer-aided one. [1] To guarantee that the computerized system would perform efficiently for at least the duration of the coverage area, the transfer was completed all at once. After a series of mishaps, the system became virtually nonfunctional as demand grew throughout the day. Delays in ambulance dispatch in a number of scenarios suggest that the failure has resulted in deaths or serious injuries.

It is becoming more difficult to distinguish between an embedded platform and a typical control system, since computers are becoming "integrated" into society. As a consequence, not only do classic safety-critical applications face major failure implications, but so do whole new application domains. DDoS assaults on networked information systems are getting more and more widespread. In addition to physical pressures, damage may happen from service interruptions or data corruption. Many software engineers are involved in the creation of systems requiring a high level of safety technologies. The number of gadgets that must be designated "safety critical" is expanding at an ever-increasing rate. Is there anything else we should know?

1) One of the primary causes of software and systems engineering failures is the breakdown of communication between the two disciplines. System modeling approaches must be created to assess the full system's properties. Techniques of this kind must provide high-fidelity representations of fundamental software attributes while still being able to accommodate software.
2) It's obvious that we're having trouble describing exactly what software is meant to perform, and this is at the root of many major failures. Even if specified processes are followed, there is a lack of collaboration that prevents thorough requirement analysis from taking place.
3) When a system has to operate in what is known as the mega range, verification via testing is out of the question. There are, however, a limited number of possibilities. However, the usage of formal verification and model checking is limited. We'll need high-performance, quick, and thorough verification methods in order to have trust in the large array of safety-critical equipment that customers anticipate.
4) For safety-critical systems, confidentiality has become an increasingly important issue that must be addressed completely if such systems are to operate correctly. Rather than a security issue, this is a software engineering one. For the most part, networked information system security issues arise as a result of security defects in the software. Buffer-overrun attacks are well-known, yet they continue to occur because susceptible systems remain in use.
5) Rather than a security issue, this is a software engineering one. For the most part, networked information system security issues arise as a result of security defects in the software.

3. Software Development Process of Safety Critical Systems

Institutionalization of approval processes and certification is a result of the seriousness of possible safety failures. For medical equipment like the digital insulin pump to be sold in the US, they must be approved by the FDA (U. S. Department of Health, 2010). Equivalent regulatory organizations in other countries must confirm that safety-critical systems are fit for their intended use before they can be authorized and certified for their usage of safety-critical equipment. By doing so, we may be certain that suitable development procedures have been followed and that the ultimate result is system accuracy. It is essential that the applicable standards' goals can be proven. These specifications specify the requirements that must be met before a system or project may be put into operation in its intended setting. Most of these standards have focused on setting process objectives and criteria for verifying those procedures have been followed since proving deterministic correctness for any considerable piece of software is basically impossible. Since these components are so crucial to public safety, their development must be closely monitored and regulated by legislation [20]. External auditors must be able to evaluate whether or not the system can be authorized since the FDA demands a high degree of auditability inside the software development process itself [21].

Process models and maturity models for ISO standards-based safety-critical systems development are required. In order to make sure that the program is produced correctly, validation verifies that the correct system is built [22]. It is not uncommon for a variety of procedures and paperwork to be required for software verification and validation by the FDA [23]. Most of the FDA's regulations are based on the activities that take place throughout the development phase

(e.g., study, design, and execution of a project's specific requirements, etc.).

3.1. Traditional Methods

When building safety-critical software, the conventional approach has been to construct a formal development process based on project management and quality assurance expertise [24-26]. The DO-178C (Software Considerations in Airborne Systems and Equipment Certification), DO-331 (Model Based Development and Verification), and DO-332 (Considerations in the Certification of Airborne Systems and Equipment) standards are only a few examples (Object Oriented Technology and Related Techniques). As stated in these standards, every process used to build safety-critical systems must achieve certain goals, although these standards do not prescribe the methods themselves. Weiguo and Xiaomin presented a technique for FDA-compliant medical device projects [27]. However, rather than dictating a specific procedure or model, DO-178B sets forth an overall framework for the development process. The DO-178B standard specifies the connections between three distinct processes that are common to most life-cycles: planning, technical development, and integration. A software life-cycle is built for each software product to be created in accordance with DO-178B that involves these three steps.

Stakeholder and client requirements are gathered at the beginning of a project and a sequential project plan is then built to meet those demands using the Waterfall method of project management [28]. With the waterfall methodology, project management is the most prevalent project management method across a wide range of sectors and research organizations. Chronological sequencing simplifies the execution and comprehension of engineering concepts in various engineering areas, from building to information technology and software development, among others.

Developing a safety-critical software system often takes a long time. The V-model, which is shown graphically in Figure 1, serves as the foundation for many of these procedures. Here, we've broken down the development process into a series of distinct phases. Manage communication and scale, organize stages and disciplines, regulate traceability (needed by sufficient safety requirements), and get certification. Certification is the last step in the development process, and it is typically the most pressing issue for the team working on mission-critical systems. Testing and evaluation by an independent third party (i.e., a certifying body) is required to ensure the safety-critical system may be placed into operation; we call this "certification," although the language used may differ across industries.

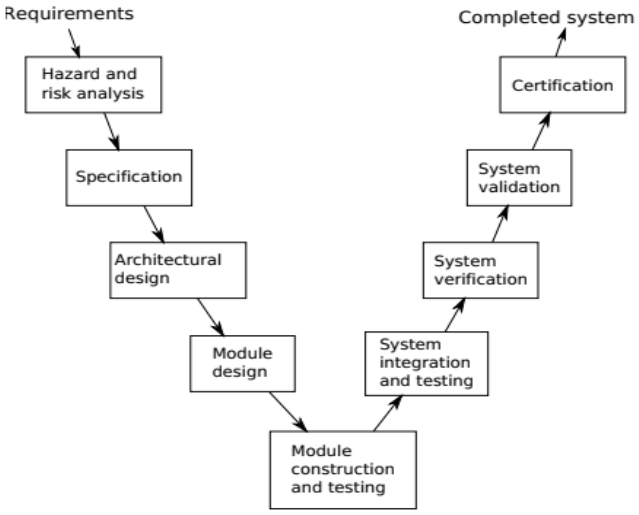

Fig.1. System development using the V model [29].

Limitations of Traditional Methods

In the following session, we have discussed the shortcomings of formal approaches, as well as their limits and tried to prove them as thoroughly as we can.

- A basic problem is that a first-order framework cannot fully describe so-called "nonfunctional" criteria and qualities like safety and security.
- There are a lot of intricacies in formal specs. Most individuals working in the computer sector at the moment are used to informal approaches and lack formal notation training. A mathematical notation is frequently more difficult to grasp than an informal statement.
- All features of reality cannot be represented using formal techniques. To cope with modeling and abstraction, formal methods use techniques that are both static and dynamic, while the actual world has both. Formal approaches may be used to model dynamic characteristics, however the model created does not accurately represent the dynamic behavior of the systems that are being modeled.
- Proofs of correctness are time-consuming. This is due to the fact that formal specifications take a long time to create. Proofs of correctness must be performed manually or interactively with machines due to the

inherent difficulties of doing them automatically (e.g., assertion creation, related knowledge management, and efficient usage).

- The price of creation goes up (for some companies and projects). This is mainly due to the fact that many firms and enterprises have to spend more money on formal techniques training for their employees.
- Even with formal specifications, mistakes might occur. Formal approaches, as previously indicated, may aid in the reduction of mistakes caused by misunderstanding. However, there is no assurance that formal specifications will be error-free (e.g., syntax mistakes and semantic discrepancies). For formal specifications, no formal technique has yet been developed that automatically verifies semantic consistency (for syntax and type checking, there are just a few options).
- Many popular formal techniques cannot define a planned system or an individual component of an existing system. Some formal techniques (e.g., VDM and Z) were initially designed for non-time-critical systems rather than time-critical systems. The definition of timing characteristics is difficult or impossible in time-critical applications.
- Formal techniques can't be used since there aren't any environments that support them. Some formal approaches can be supported by tools that are already available (e.g., The University of York produced CADIZ for Z notation, while the University of Manchester constructed a mural for VDM). These technologies are unable to enable formal methods tasks including consistency checks, accuracy proofs, improvement of specifications, and software testing.
- It's important to note that Waterfall-based engineering product development has a number of limitations. This is mostly due to the process's rigidity and sequential development, and the absence of client involvement in product development beyond a certain point in the process that these issues arise.
- Documentation is the strength of the V- Model, but it also makes it vulnerable. If one criterion is altered, it may have an influence on others, necessitating further documentation rewrites. As a result, it is possible for the project team to operate for a lengthy amount of time following the "internal" parts of the V-Model without engaging with the client once the requirements elicitation process has been completed.
- Customer engagement is only guaranteed at the beginning and the conclusion of a V-model process. Due of its intrinsic resistance to changing client requirements, the V-Model is frequently criticized for its rigidity and inflexibility.

3.2. Agile Development (Scrum Development)

Iterative cycles of "plan, construct, revise" are used in Agile software development. They produce progressively ready product features at the conclusion of each cycle. Agility is a good fit for software development because of the latency, adaptability, and speed of reaction that are all necessary in today's world. There are two foundations for agile software development: the Agile Manifesto [30] and the Principles behind the Manifesto [31]. Openness to changing client requirements and a strong focus on providing viable software in close cooperation are hallmarks of agile techniques. There are no designated responsibilities for quality control, safety management, or documentation in agile methodologies like Scrum [32].

Scrum is the most popular framework in industry, and it serves as an important representation of Agile development processes [33], as shown by books like "The Scrum Guide" and "The Scrum Primer, version 1.2." Components such as teams (with duties assigned to individuals), events and artifacts, and rules, are all included. In order to produce the product in a uniform way, each component of the framework must be accurately implemented. Sprints are time-boxed periods of no more than one month in which the development activity is divided into the objectives that must be fulfilled throughout each iteration. Progress is recorded on a daily basis, in the form of brief sessions that are focused on creating daily objectives and eliminating any impediments as quickly as possible. The Sprint review provides a summary of the work that has been done in each Sprint.

A. Existing Safety-Critical Software with Agile Development

Increased IT market competition has had a negative impact on critical safety systems. They have expanded from serving solely hospitals and physicians to delivering individualized e-health software solutions and equipment for individual patients, as well. In order to compete in this ever changing and expanding industry, it is essential to deliver better software that is more attractive to the customer while keeping prices down. Software development processes must be made more efficient (in terms of work, user happiness, and time) while still adhering to the safety criteria set by applicable standards, rules and recommendations. More and more studies show that implementing agile principles into safety-critical projects is possible and even beneficial. There is some example-

1) According to Lukasiewicz [34], " It's possible that the limited adoption of agile methodologies in regulated safety-critical areas reflects a reluctance on the part of firms in these fields to disclose their internal procedures." when it comes to adopting agile methods in regulated settings. It was revealed in their article that agile approaches had several problems and recommended ways to fix them.
2) For high-integrity system development, Paige [35] explored the use of agile, including the examination of aspects of agile and the adaptation of agile procedures. While agile approaches may be used to safety- critical

development, the authors found that they work best when used in conjunction with plan-driven procedures.

3) To the contrary, Gary [36] believes that agile development methods are more suited for open-source safety-critical software than other approaches. A perspective that allows for conventional safety-oriented techniques to the degree that they are necessary is a hallmark of agile approaches. This supports Boehm's [37] claim that safety-critical software development requires sophisticated project management.
4) Agile Assessment, according to Pikkaraine [38], is an effective tool for clarifying whether agile methods are appropriate for product development and client service. One of the findings was that the use of the most suitable agile methodologies during incremental development would boost the tracking and monitoring of requirements.
5) Sidky's [39] three key phases correspond to Douglass' six steps for effective agile adoption. They all believe that it's important to understand why agile software is a good fit for their business.

B. Problems with Agile Software Development and Safety-Critical Design

Agility in a safety-critical software development setting has four areas of concern that need to be addressed:

1) Light documentation;
2) User stories are a great way to describe flexible needs;
3) The lifecycle is iterative and gradual;
4) Testing is a first step.

Light documentation

Documentation is a major roadblock in the expansion of agile software development for safety-critical products. Several studies have examined how documentation might become a major component of agile development rather than an afterthought. They say agile procedures prioritize functioning software over detailed documentation. This does not imply that all documentation is useless, and some agile techniques appreciate the value of documents. As a rule of thumb in agile software development, documentation should be just enough. Moreover, Scrum encourages that developers utilize documentation and models to organize their thoughts.

Documentation is vital in safety-critical software development because it proves that all procedures were followed and the program is safe. Documentation is a cornerstone of the regulatory process norms and criteria. Regulations and the prospect of regulatory inspections encourage substantial record keeping [40]. Agility's low or nonexistent attention on documentation is inadequate to identify the quality of safety-critical systems, hence regulatory authorities will not consent to less documentation.

Additional documentation is required for the development of other safety-critical software features. In order to develop safety-critical systems, it is feasible that staff members may be changed numerous times over the course of years. During the 30-year lifetime of safety-critical goods, a third party must regularly maintain, update, and enhance the software [41]. Thus, safety-critical projects must concentrate on documentation, since adequate documentation is required to verify software safety. Documentation should be kept to a minimum and treated lightly.

User stories are a great way to describe flexible needs

In terms of demand management, agile and conventional software development approaches vary in two respects. First, whereas agile procedures promote requirement modifications, safety-critical development approaches discourage them owing to the increased expenses of software redesign, testing, and requirement documentation. Second, agile procedures abandon conventional requirements in favor of loosely organized customer-authored user stories stated in simple business-like language.

Securing safety is a requirement management issue, and thorough requirement specifications are vital. Modifications to the software architecture [41] may affect the important evidence needed for validation and verification of the program's safety [42]. This is especially true in projects that are safety-critical. It is, nonetheless, necessary in software development. Even if the initial study is rigorous, the requirements are precise, and numerous approval signatures are acquired, modifications will occur over time [43]. These challenges are particularly likely in security software development [44]. A complicated medical equipment takes years to build. During this period, it is conceivable that requirements may alter or be added. Some research demonstrate that modifications are less prevalent in less crucial software development [42]. There are two types of requirements: safety and functionality. Less stable are the functional needs. It is therefore recommended to add a safety-product backlog to the original Scrum approach. This is used to distinguish regularly changing functional needs from more stable safety requirements [45].

User stories expressed in normal business language cannot be utilized for validation because safety-critical software development necessitates well-structured requirement engineering [36]. To avoid this, the user stories' informal requirements should be defined. When it comes to formal papers, paper cards may be used, but instruments must also be employed to store the requirements. As an example, an example was given by the authors of how they used it a use case document, together with a definition of software requirements, to outline criteria that are not functional. Security-related user stories and abuser tales (threat circumstances) should be two distinct types of stories [46].

The lifecycle is iterative and gradual

The recommended stages of a project's lifetime range vary significantly between agile and conventional methodologies. All elements of the development process are included in iterations according to agile approaches, allowing for more adaptability and the ability to respond to changes. An increment (a functioning, tested version of the system) should be released at the end of each cycle. Scrum employs sprints in which developers work uninterrupted for short periods of time.

The very iterative and incremental approach that is at the foundation of agile methodologies must be preserved in contexts where software development is safety-critical. It is thus necessary to adapt the plan-driven lifecycle models [47] to be iterative. In certain studies, it has been observed that this is an obstacle to the widespread adoption of agile software development methods [48]. Developing an incremental safety assurance approach is the main problem [49].

Research shows that iterative safety-critical development is possible and even beneficial, since developers are pushed to break down activities and get a better knowledge before trying to address problems [50]. Incremental development may be used in safety-critical projects, although it is more challenging than in non-critical projects [44]. During iterations spanning four to ten weeks, Rasmussen et al. supplied executable software increments. It is possible that longer iterations than recommended for non-critical software development may be required. He eager [50] report on dissatisfied engineers who had to implement a complete increment over the course of four weeks, which lends credence to this theory. As a result of the plan-driven project milestones and sequential process, software development iterations were subject to a specific emphasis (for example, documentation) in a system engineering project that was primarily focused on documentation [50].

Testing is a first step

Agile teams often employ test-driven development. For example, test cases are developed before the program. Passing all checks determines whether or not an increment is complete. In order to accomplish quick iterations, agile methodologies rely on automating tests.

Deploying safety-critical software requires substantial [49]. Testing is completed at the conclusion of development using the V-model. Conciliating these two testing procedures has been challenging since developers are not accustomed to prioritizing tests [50]. Other empirical studies describe effective implementation of test-first approaches in safety-critical software development [41, 48]. A study of regulatory requirements shows that test-driven development is compatible.

According to other studies, the implementation of an agile test-first approach may work, but it must be further improved to meet safety standards [52]. A problem with test-driven development is that developers must create their own tests since certain standards (such as EN 50128) state that the tester is responsible for defining the test and that the developer and the tester must be separate persons. Additional testing is also required to meet the tight regulatory standards [49]. As a result, it takes longer and is more expensive for agile teams to produce software.

3.3. SafeScrum- Scrum Development of Safety Critical System

At its inception, Scrum was intended to be used on modest, self-managed development initiatives and to address challenges of a limited scope. According to studies conducted over the previous decade, projects managed using Scrum are more successful than those based on traditional methods. Scrum, on the other hand, is rapidly being adopted in more complicated environments in the software business today. We've seen Scrum teams from across the world working together to construct massive software systems. The creation of safety-critical systems, which must meet stringent quality and safety criteria, is another area where Scrum is being applied. The fundamental ideas of self-sustaining, multidisciplinary, and self-managed teams are being tested as complexity rises.

For the sake of complying with the IEC61508 standard, SafeScrum is an adaptation of the well-known and widely used Scrum methodology [53]. Based on a thorough assessment of the software requirements, Hanssen provided a set of changes to Scrum that might be used to the development of safety-critical software [54]. It should be noted that the functional requirements and the safety requirements have two separate backlogs. A product's functional needs may vary often, but its safety criteria are generally consistent across projects and products, and even re-usable. Using relationships, keep track of the safety criteria that are impacted by functional requirements. SafeScrum must also be able to be tracked. During the course of the project, every decision and modification must be recorded, preserved, and made accessible to the assessor. A record of all comments and how they were handled should be preserved for code reviews as well. As a third step, the current system's safety is verified during each sprint. The product backlog may be revised throughout each sprint's review. CIAs [55] are performed and recorded when a modification is thought to have an effect on the system's safety. Two backlogs may be used to determine the impact a functional change may have on a safety requirement in this case. Additionally, agile approaches like as test-driven development, every day stand-ups, numerous work iterations, and continuous integration are very vital.

Code and assessment are no longer necessary under this architecture, which uses a series of sprints instead. This implies that, contrary to the V-model, documentation is generated in real time and as part of the development process. Replaces just the bottom and right of the V-model SafeScrum. For software development, this is a crucial need that allows a project to be more adaptable to changes while still providing the necessary documentation and traceability to the assessor.

In order to satisfy the standards of IEC61508 standard, several additional aspects must be added to Scrum. Consequently, the concept of a lightweight process is thrown out the window, since a great deal of extra labor, checkpoints, and documentation is necessary. The most effective countermeasure is to automate as much of the extras as possible via the intelligent and efficient use of tools. We've categorized tools into four groups. When it comes to managing processes and workflows, we need a tool that can help us define and track the many roles and responsibilities of each individual activity. An automation tool for the Scrum board, to put it plainly. As a second need, we require tools for establishing and maintaining the backwards compatibility, tests, and source code. An even tiny project generates an enormous quantity of data, and this data has to be supported by tools. Thirdly, we need specialized tools for organizing information such as design, code, and architectural documents. Also necessary in light of IEC61508, is a set of tools to aid with code quality assurance. Test automation, test coverage analysis, and static code analysis are all included. Tools for the SafeScrum process may be found in a variety of forms, many of which are interchangeable and can be connected together. This standard, IEC61508, specifies specific criteria for tools, and they should be examined (IEC61508-3). We'll see instances of this kind of tool chain in action as we go through our scenario.

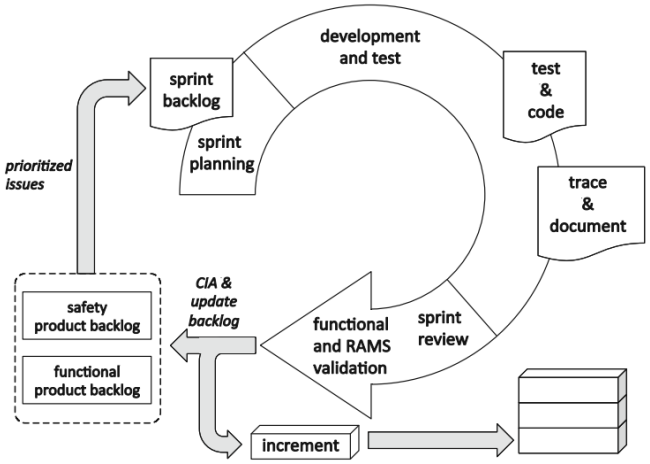

Fig.2. SafeScrum Model.

4. Methodology

Even though online surveys are the norm these days, researchers have a wide range of choices for gathering data. Once the survey's goals and sample strategy have been established, the following step is to choose which data gathering method is ideal for the project. Often, we presume a single means of data gathering, such as internet data collection for some and phone data for others. In certain situations, it may be preferable to use more than one mode, however this is not always the case.

4.1. Data Collection

Determining the research topic helps researchers focus their efforts on narrowing down the scope of their study to specific areas of interest. Researchers must first identify their research topics in order to choose the appropriate approach, methodologies and sample size for data collecting as well as the appropriate tools for data analysis. As researchers go through the design cycle, evaluating literature, integrating theory, and establishing a conceptual framework, they must constantly adjust and improve their research questions. The establishment of a specific research goal and target is impossible without a well-defined research issue.

To design the questionnaire in this study, the PICOT methodology was applied. It was initially presented in 1995 as PICO format, and was then enlarged to PICOT format [56], which is now a generally recommended technique for framing research questions. Notably, the PICOT format is often useful for comparison studies or examinations of the association between exposure and result (s). The PICOT approach is often recommended by evidence-based medicine for framing research questions. That is why we used it. Research demonstrates that the use of PICOT improves the results of clinical information searches in PubMed.

4.2. Data Analysis

A. Qualitative research

While the goal of any interview is to learn more about the subject being interviewed, the specific goal depends on the research topic and the researcher's academic viewpoint. There are two kinds of studies: those that are aimed to test pre-existing hypotheses and those that are designed to investigate meaning and perceptions in order to get a better understanding and/or produce hypotheses. Qualitative interviewing is often required for this kind of study, which encourages the interviewee to provide detailed descriptions of events while letting the investigators to interpret or analyze them. There are many different ways to conduct a qualitative research interview, but the goal is to get

information that is both conceptual and theoretical, and is based on personal experiences.

Qualitative research highlights study participants' viewpoints to "illuminate subjective meaning, acts, and situations" [57]. Quality qualitative research is defined by how well participants' viewpoints are reflected and understood throughout the research process and in interpretations derived from the material acquired (authenticity). In qualitative research, the researcher-researched interaction and the need for transparency (openness and honesty) are crucial [58].

B. Quantitative research

Research using statistical or numerical data to study social issues is known as quantitative research. Quantitative research is based on the assumption that events can be quantified, and so requires measurement. To uncover patterns and linkages, it examines data and evaluates the results. Quantitative research is an approach to inquiry. Measurements, evaluations, and inferences are all part of the process of deduction. It is possible to conduct quantitative research using hypotheses and statistical analysis to investigate concepts (Figure 3).

The numerical data generated by quantitative research must be analyzed. Enter, store, and analyze data electronically with the help of a database management system for example, enter data into an Excel or Word spreadsheet. Simple data analysis may be accomplished with Excel, but more complex analysis may be better served by SPSS [59]. Transcribing survey results or observation schedules into a computer is a necessary step in the data entry process. This has to be thoroughly investigated. Surveys are increasingly delivered online, and data may be immediately loaded into an analytical tool such as SPSS.

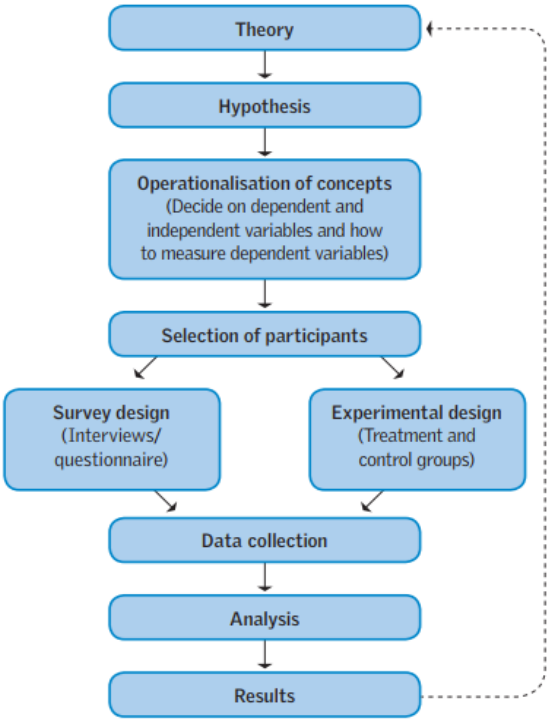

Fig.3. Quantitative Research Design [60]

While cross-sectional surveys are the most often used, longitudinal surveys provide the greatest insight into the evolution of individuals and communities because of their ability to track changes over time. The collection and storage of quantitative data in electronic databases is critical in quantitative research, as is the analysis of quantitative data using proper statistical techniques.

C. Mixed-mode research

Flat or declining survey response rates have plagued researchers in recent times. In the internet and telecommunications age, they must choose the best way to target and communicate with their survey group. Mixed-mode surveys are used to solve these problems. The mode selection dictates who may be interviewed and how to approach the target group most effectively. The method used by researchers is crucial to the study's success and credibility. Changing modalities may affect people' reactions and involvement.

Using both quantitative (such as surveys) and qualitative (such as in-depth interviews and focus groups) research methods, mixed mode research is a method that combines these two forms of information to provide more thorough, practical, and lucrative market research findings. Both quantitative and qualitative research may be done at the same time, but they don't have to be done simultaneously. As an example, when researchers are doing field experiments (quantitative data) and conducting focus groups (qualitative data), they don't necessarily need to be doing both at once.

Although all data collection methods must be part of the same overarching study topic, they must both take place in close proximity to one another.

D. Why mixed-mode research?

Because mixed-mode surveys provide researchers a wide range of possibilities, they are becoming popular. Survey usage has grown because to widespread internet access and e-mail. Using simple approaches that are more able to elicit a respond, such as telephone or cell phones, "snail mail" questionnaires, emails or online surveys, may help researchers better target their clients. Previously, researchers were unable to tailor a survey to the specific needs of each region.

Not only may researchers increase their coverage of their chosen demographic by addressing them in a variety of methods, but they can also aim for a greater response rate. Participants who consistently decline online surveys may be more receptive if reached through phone. Additionally, providing survey respondents with many options for providing feedback often enhances response rates. Additionally, it's worth noting that mixed-mode surveys may give cost savings in certain circumstances. Everyone is aware that research may be costly. Surveys that were formerly conducted entirely over the phone may now be conducted partly online, resulting in significant cost and time savings for the customer.

E. Benefits of mixed-mode research

Mixed mode research has several advantages, particularly in an era of tighter deadlines and reduced prices. The Bunker has compiled a list of mixed mode research advantages and how we use them in our daily research:

- **Increasing Response Rate:** People desire choice, flexibility, and alternatives. This is among the key reasons we use Mixed-Mode with our customers. Providing other locations for participants to reply increases response rates. The data collecting does not follow one path.
- **Expanding Reach:** Each technique of data collecting has a certain audience in mind. The youthful 20-year-olds would be ideal for a fast pulse survey. It's a poor idea to get a generic landline phone sample. A new independent living complex may want to conduct focus groups with elderly residents. Your best strategy is not to send out online survey invitations. Don't limit your audience's options for responding if you're running a survey with a wide range of demographics. This will prevent you from overlooking a group.
- **Increased Data Quality as a Result of Increased Representivity:** Mixed-modal survey research may improve comparability by reaching out to those who are difficult to reach through a single mode. For instance, surveying a particular target group using a mix of email/online, telephone, and snail mail may provide a better coverage of the researcher's target demographic than online alone.
- **Accelerating Fieldwork:** Utilizing Mixed-Mode might be an excellent technique to adhere to tough deadlines. When combined with improved response rates, their increased flexibility in responding leads in more responsive initiatives. Are you about to send out a mail survey? Why not send out an e-mail invitation as well, so they may complete the survey at work if they left it on the kitchen table? Are you still unresponsive? How about making follow-up reminder calls and, if they agree, completing the survey over the phone?

In this research, we have used Mixed-Mode research for data collection. Mixed mode research combines quantitative (surveys) and qualitative (in-depth interviews and focus groups) research approaches to deliver more detailed, useful, and profitable market research insights.

4.3. Result Analysis

An initial survey question was aimed to gather information about the respondents' roles, knowledge and expertise with safety critical systems engineering standards and agile approaches. The survey's primary section consisted of a series of comments about the possible integration of safety engineering tasks with agile development. To prevent replies from being skewed, both positive and negative comments on the possibility of integration were provided. In addition, some statements were created with intentional overlap and/or contradiction to test the respondent's consistency.

The vast majority of those who took the time to answer worked for companies with at least 25 employees. More than 80% of those surveyed have worked on safety-critical systems in the past. 90% of those surveyed have used agile development approaches in the workplace.

The study began with 80% of respondents agreeing that agile techniques can be integrated with the creation of safety-critical systems, while 10% said that agile and safety procedures are in compatibility, and 10% believed that certain agile and safety practices are in harmonious relationship while others are in conflict.

In this study, it is not feasible to offer a systematic presentation of the findings collected for the survey's more specific questions. These findings are instead illustrated:

- Over 95% of respondents agreed that consumer safety assurance and consumer safety must be provided to consumers. While developing, it's important to consider the potential risks.
- 75% of respondents agreed that engineers with system-level expertise and experience should be involved in software safety assurance.
- 70% of respondents also agreed that regulators and evaluators should be involved in the software

development process. A safety-critical systems development environment may use this information to better define the typical agile 'client' role.

- Almost all of the participants (90%) agreed that the safety case should be regularly evaluated throughout the development process. Safety risk management construction should be seen as an "improvement" rather than an "outcome."
- Requirements and Documentation were ranked highly in terms of importance (77% and 84%, respectively). When all of this is looked at together, it shows that an incremental safety case building might be a good way to guide a software development process that is done in steps.

5. Discussion

In this study, we gathered common ideas, practices, and constraints from the literature on safety-critical agile methodologies and compared them to the insights of industry professionals. We discover an expanded and polished list of guiding principles and obstacles, as well as a number of potential solutions. A majority of the answers show a favorable outlook on the use of agile and safety measures together.

The responders feel that in agile development for safety-critical systems, the safety documentation should assist safety-related communication to the utmost degree with a minimum quantity. The minimal amount of safety documentation should also be tailored to diverse scenarios. By restricting and exploring the proper set and type of safety documentation in agile development, both the formal safety-related communication and the informal communication might be enhanced. The standard that we wish to fulfill from the system and the definition of the safety level should also be understood.

The literature shows that formal approaches are still failing to acquire broad acceptance and application in ordinary software development but have seen use in validating important features of safety-critical systems. Due to the absence of rigorous validation methodologies, agile approaches have received widespread industry adoption in a short period of time. It is hoped that combining the two methods of software development would make the advantages of formal specification techniques more obvious as well as allow the adoption of agile development techniques for critical systems. The introduction of formal specifications into a widely used industrial process like Scrum will make formal specifications more approachable and hence enhance their industrial adoption. This will shift the emphasis of agile approaches, which have historically been focused on keeping projects on schedule and within budget, to concentrate more on system quality; this will get us closer to the holy grail of error-free code being built on schedule and under budget.

To sum it up: the safety narrative and epic patterns are highly recommended for decreasing communication issues in Scrum-based safety-critical systems, based on our study. In addition to well-documented, well-structured requirement elicitation and further testing, it is necessary to fulfill regulatory criteria. However, greater research is needed to examine the impact of an agile safety strategy in many situations, particularly in industry.

References

[1] Nuclear Regulatory Commission, Information Notice 96-29, Washington, DC (May 1996).
[2] European Space Agency, Ariane 501 Inquiry Board Report (July 1996) http:/ /ravel.esrin.esa. it/docs/esa-x-1819eng.pdf.
[3] Leveson, N. G. (1995) Safeware: System Safety and Computers. Addison-Wesley.
[4] MC Software, Inc. http://www.mscsoftware.com/products/index.cfm? S=85.
[5] N. Wirth, Towards discipline of real time programming, Commun. ACM 20, 1977.
[6] D. Harel and A. Pnueli, On the development of reactive systems. *Logics and Models of Concurrent Systems*, (K. R. Apt, ed.), Springer-Verlag, 1985, pp. 477-498.
[7] J. F. Cassidy, T. Z. Chu, Kutcher, S. B. Gershwin, and Y. Ho, Research needs in manufacturing systems, *IEEE Control Syst*. 5, 11-13(1985).
[8] W. J. Quirk, *Verification and Validation of Real-Time Software*, Springer-Verlag, Berlin, 1985.
[9] A. H. Levis, Challenges to control: a collective view, *IEEE Trans. Auto Control* AC-32, 1987.
[10] J. A. Stankovic, Misconceptions about real-time computing: serious problem for next generation systems, *Computer* 21, 10-19 (1988).
[11] S.-T Levi and A. K. Agrawala, *Real Time System Design*, McGraw-Hill Publishing Company, 1990.
[12] D. L. Parnas, A. J. van Schouwen, and S. P. Kwan Evaluation Standards for Safety Critical Software, Technical Report TR 88-220, Queen's University, Kingston Ontario, Canada, 1988.
[13] Marshall, P., NII Found Problems in 50%-Plus of Sizewell-B PPS Computer Tests, Nucleonics Week, 34, 43 (October 28, 1993).
[14] National Aeronautics and Space Administration, Mars Climate Orbiter Mishap Investigation Report, Washington, DC (November 1999) ftp://ftp.hq.nasa.gov/pub/pao/reports/2000/ MCO_MIB_Report.pdf.
[15] Jet Propulsion Laboratory, Report on the Loss of the Mars Polar Lander and Deep Space 2 Missions, JPL D-18709 (March 2000).
[16] Hatcliff, J., Wassyng, A., Kelly, T., Comar, C., and Jones, P. (2014). Certifiably safe software dependent systems: challenges and directions. In *Proceedings of the on Future of Software Engineering - FOSE*, (pp. 182–200).
[17] M Heimdahl, M. P. E. (2007). Safety and Software Intensive Systems: Challenges Old and New. In FoSE 2007: *Future of Software Engineering* (pp. 137–152)..

[18] Day, J.W., The Reliability of the Sizewell 'B' Primary Protection System, Reactor protection Group, Nuclear Electric (January, 1990).
[19] Neumann, P. Computer Related Risks, Addison Wesley (1995)
[20] Spence JW (2005) There has to be a better way! In: Proceedings of the Agile Development Conference (ADC'05), Denver, USA, IEEE, pp 272-278.
[21] Mehrfard H, Hamou-Lhadj A (2013) The impact of regulatory compliance on Agile software processes with a focus on the FDA guidelines for medical device software. International Journal of Information System Modeling and Design, 2(2): p. 67-81.
[22] Del Bianco V, Stosic D, Kiniry JR (2010) Agile Formality: A Mole of Software Engineering Practices. In: Proceedings of the 2nd International Workshop on Formal Methods and Agile Methods, Pisa, Italy, pp 29-48.
[23] Mehrfard H, Pirzadeh H, Hamou-Lhadj A (2010) Investigating the capability of agile processes to support life-science regulations: the case of XP and FDA regulations with a focus on human factor requirements. In: Proceedings of the 8th Conference of Software Engineering Research, Management and Applications (SERA), Montreal, Canada, Springer, pp 241-255.
[24] Boehm, B.W. and Ross, R. (1989), "Theory-W software project management principles and examples", *IEEE Transactions on Software Engineering*, Vol. 15 No. 7, pp. 902-916.
[25] Sommerville, I. (2015), *Software Engineering*, Addison-Wesley, Harlow, UK.
[26] Zultner, R.E. (1993), "TQM for technical teams", *Communications of the ACM*, Vol. 36 No. 10, pp. 79-91.
[27] Weiguo L., Xiaomin F.: *Software Development Practice for FDA-Compliant Medical Devices*. Proc. of the 2009 International Joint Conference on Computational Sciences and Optimization, 2009.
[28] Royce, W., 1970, „Managing the Development of Large Software Systems", *Proceedings of IEEE WESCON*, 26, pp. 328-388.
[29] N. Storey, Safety-Critical Computer Systems. Addison-Wesley, 1996.
[30] http://agilemanifesto.org/. [retrieved: September, 2012].
[31] http://agilemanifesto.org/principles.html. [retrieved: September, 2012].
[32] Roman Pichler: „Scrum – Using Agile Project Management Successfully" (Transl.), 2008, pp. 7-123.
[33] Results from Scott Ambler's March 2006 'Agile Adoption Rate Survey' posted at www.ambysoft.com/surveys/. [retrieved: September, 2012].
[34] Lukasiewicz K. assessment of risks introduced to safety critical software by agile practice: a software engineer's perspective, 2012.
[35] Paige R. et al Towards Agile Engineering of High Integrity Systems. Proc. of 27th International Conference on Computer Safety, Reliability and Security (SAFECOMP) 2008.
[36] Gary et al. Agile methods for open-source safety critical software, 2012.
[37] Boehm B. Get ready for agile methods with care.,2002
[38] Pikkarainen M. An approach for assessing suitability of agile solutions: A case study, 2005.
[39] Sidky, A. and Arthur, J. Determining the applicability of agile practices to mission and life-critical systems. In Proceedings of the 31st IEEE Software Engineering Workshop, SEW '07, pages 3–12, Washington, DC, USA. IEEE Computer Society,2007.
[40] Hajou A, Batenburg R, Jansen S (2015a) An Insight into the Difficulties of Software Development Projects in the Pharmaceutical Industry. Lecture Notes on Software Engineering 3 (4):267.
[41] Drobka J, Noftz D, Raghu R (2004) Piloting XP on four mission-critical projects. IEEE software 21 (6): 70-75
[42] Ge X, Paige RF, McDermid JA (2010) An iterative approach for development of safety-critical software and safety arguments. In: Proceedings for the Agile Conference (AGILE), Orloando, Florida, USA, IEEE, pp 35-43.
[43] Bedoll R (2003), A Tale of Two Projects: How 'Agile' Methods Succeeded after 'Traditional' Methods Had Failed in a Critical System-Development Project In: Proceedings of the 3rd Conference on Extreme Programming and Agile Methods, New Orleans, LA, USA, Springer, pp 25-34.
[44] Beznosov K (2003), Extreme Security Engineering: On Employing XP Practices to Achieve 'Good Enough Security' without Defining It X. In: Proceedings of the 1st ACM Workshop on Business-Driven Security Engineering (BizSec), Fairfax, USA, Citesee, pp 1-7.
[45] Abdelaziz AA, El-Tahir Y, Osman R (2015) Adaptive Software Development for developing safety critical software. In: Proceedings of the 1st International Conference on Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE), Khartoum, Sudan, IEEE, pp 41-46
[46] Boström G, Wäyrynen J, Bodén M, Beznosov K, Kruchten P (2006), Extending XP practices to support security requirements engineering. In: Proceedings of the 28th International Workshop on Software Engineering for Secure Systems (SESS), Shanghai, China, ACM, pp 11-17
[47] Del Bianco V, Stosic D, Kiniry JR (2010) Agile Formality: A Mole of Software Engineering Practices. In: Proceedings of the 2nd International Workshop on Formal Methods and Agile Methods, Pisa, Italy, pp 29-48.
[48] Beznosov K, Kruchten P (2004), Towards agile security assurance. In: Proceedings of the 17th Workshop on New Security Paradigms, Victoria, Canada, ACM. pp 47-54.
[49] Górski J, Łukasiewicz K (2012) Assessment of risks introduced to safety critical software by agile practices-a software engineer's perspective. Computer Science 13 (4):165-182.
[50] Heeager L (2012) Introducing Agile Practices in a Documentation-Driven Software Development Practice: A Case Study. Journal of Information Technology Case and Application Research 14 (1):3-24.
[51] Grenning J (2001) Launching extreme programming at a process-intensive company. IEEE Software 18 (6):27.
[52] Górski J, Łukasiewicz K (2013) Towards Agile Development of Critical Software. In: Proceedings of the 3rd International Workshop on Software Engineering for Resilient Systems, Kiev, Ukraine, Springer, pp 48-55.
[53] Stålhane, T., Myklebust, T., Hanssen, G.K.: The application of Scrum IEC 61508 certifiable software. In: Proceedings of ESREL. Helsinki, Finland (2012).
[54] Myklebust, T., Stålhane, T., Hanssen, G.K., Wien, T., Haugset, B.: Scrum, documentation and the IEC 61508-3:2010 software standard. In: Proceedings of Probabilistic Safety Assessment & Management conference (PSAM12). Self-published, Oahu, USA (2014).
[55] Stålhane, T., Hanssen, G.K., Myklebust, T., Haugset, B.: Agile change impact analysis of safety critical software. In:

Bondavalli, A., Ceccarelli, A., Ortmeier, F. (eds.) SAFECOMP 2014. LNCS, vol. 8696, pp. 444–454. Springer, Heidelberg (2014).
[56] Haynes RB, Sackett DL, Guyatt GH, Tugwell PS. *Clinical epidemiology: how to do clinical practice research*. 3rd ed. Philadelphia, PA: Lippincott Williams & Wilkins; 2006.
Popay J, Rogers A, Williams G. *Rationale and standards for the systematic review of qualitative literature in health services research*. Qualitative Health Research 1998; 8:341–351.
[57] Lincoln YS. *Emerging criteria for quality in qualitative and interpretive research*. Qualitative Inquiry 1995; 1:275–289.
[58] Pallant J (2007) SPSS Survival Manual: *A Step-by-Step Guide to Data Analysis Using SPSS for Windows Version 15*. Third edition. Open University Press, Maidenhead.
[59] Bryman A, Cramer D (2005) Quantitative Data Analysis with SPSS 12 and 13: *A Guide for Social Scientists*. Taylor and Francis, London.

Authors' Profiles

Mustafizur Rahman is born in Bangladesh on 13th June 1997. He did his BSc in Computer Science and Software Engineering from American International University-Bangladesh in 2022. His research focus is on Software Quality and Testing. His future target is to become a Software Test Engineer.

Shusmita Islam born in Bangladesh on 4th November 1998. She is a student of American International University of Bangladesh. Her general research interest is in the area of information technology and software engineering.

Rubiyet Fardous is a Computer Science graduate from American International University-Bangladesh. His general research interest is in the area of machine learning algorithms and web development.

Lamisa Yesmin Lorin is born in Bangladesh on 16th February 1999. Currently she has completed her Bachelor's degree in Computer Science and Software Engineering under the Faculty of Science and Technology at American International University- Bangladesh.

Prof. Dr. Dip Nandi is the Director of Faculty of Science & Technology (FST), at American International University- Bangladesh (AIUB). His research interest includes the concept of Software Engineering Model & Process, Data Science, E-Learning, Machine Learning etc.

How to cite this paper: Mustafizur Rahman, Shusmita Islam, Rubiyet Fardous, Lamisa Yesmin, Dip Nandi, "Applying Scrum Development on Safety Critical Systems", International Journal of Information Technology and Computer Science(IJITCS), Vol.14, No.5, pp.44-57, 2022. DOI:10.5815/ijitcs.2022.05.04