GDAŃSK UNIVERSITY OF TECHNOLOGY
FACULTY OF ELECTRONICS, TELECOMMUNICATIONS AND INFORMATICS

The author of the PhD dissertation: Katarzyna Łukasiewicz
Scientific discipline: Computer science

DOCTORAL DISSERTATION

Title of PhD dissertation: Method of selecting programming practices for the safety-critical software development projects.

Title of PhD dissertation (in Polish): Metoda doboru praktyk programistycznych w wytwarzaniu oprogramowania związanego z bezpieczeństwem.

Supervisor *signature*	Second supervisor *signature*
prof. dr hab. inż. Janusz Górski	
Auxiliary supervisor *signature*	Cosupervisor *signature*

Gdańsk, year 2018

ACKNOWLEGEMENTS

First and foremost, I would like to thank my supervisor prof. Janusz Górski, who has not only guided me through researching and writing this doctoral thesis but has also been an inspiration and a mentor in my scientific development.

I would like to thank the Argevide NOR-STA team for the support and for the chance to work with their product.

I would like to thank the Protégé team for sharing their product. *This work was conducted using the Protégé resource, which is supported by grant GM10331601 from the National Institute of General Medical Sciences of the United States National Institutes of Health.*

I would like to thank the students I had pleasure to work with during the experiments, for their commitment and enthusiasm.

I would like to thank Thor Myklebust, Geir Kjetil Hanssen, Tor Stålhane and Stig Ole Johnsen very much for their support and the time they dedicated during my visits in Trondheim and beyond

I would like to thank Frank Aakvik and Ingar Kulbrandstad for their time and honesty.

I would like to thank Jakub Miler for his advice and remarks he gave me during the completion phase of this doctoral thesis.

I would like thank Teresa Zawadzka and Wojciech Waloszek for their help and support with the subject of knowledge bases

Na końcu, ale też najmocniej, chciałabym podziękować mojej rodzinie. Moim rodzicom za cały czas, jaki poświęcili, abym ja mogła mój czas poświęcić na przygotowywanie tej pracy. Mojej siostrze, za wsparcie mentalne i podtrzymywanie na duchu w chwilach zwątpienia. Oraz mojemu mężowi Michałowi - byłeś cały czas w pierwszej linii wsparcia, dzięki Tobie ta praca w ogóle mogła powstać. Dziękuję.

Dla Jasia i Adasia

Abstract

In recent years a plan-driven approach traditionally used in safety-critical software development has been put to a test by rapidly changing technologies, more diverse group of clients and volatile market requirements. The need to deliver good quality systems, faster and at lower cost in comparison to competitors encouraged companies to look for more efficient solutions. Agile methodologies are known to successfully address these issues for small, non-critical projects. Presumably agile practices can reduce both cost and time to market when applied to safety-critical projects as well. While benefits can be significant, the main concern are quality and safety assurance. Plan-driven methodologies provide tools for such purpose, which agile methodologies in their pure form lack. The challenge that arises is to elaborate a more easily available and ready-to-use solution that would help safety-critical organizations to streamline their processes with agile practices and to maintain accordance with safety standards and certifications.

The goal of the research described in this work was to develop an approach aimed at facilitating the introduction of a more agile approach to the software development process, depending on the characteristics of the project, while maintaining compliance with the required safety standards and regulations, and the AgileSafe method presented in this thesis is the main result of this research.

The information about project and about the regulatory context constraining the project and its product are the inputs to the method. User is guided through two main processes of AgileSafe: process which selects the specifications of software development practices to be applied in the Project and a process which results in the set of assurance arguments corresponding to the regulations included in the regulatory context.

The two main processes of AgileSafe reflect the main objectives of AgileSafe: to support a hybrid approach to software development based on the tailored practices and to support continuous monitoring of conformance to the mandatory regulatory requirements.

In order to further improve the method and tailor its advice to the User's needs more accurately, the knowledge stored in the method should be reviewed and updated regularly.

To validate the proposed AgileSafe method, in the course of the research, three case studies have been conducted in addition to interviews and questionnaires with participation of experts.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. CONTEXT

The ultimate goal of the software development process is to produce high quality software that would please the customers and bring a satisfactory income. The success of the project, however, is a combination of multiple factors and the right choice of a software development method is a crucial one.

The software development methods *"provide the technical how-to's for building software"* (Pressman, 2009). They offer guidelines on applying specific activities and tasks and organising them into a software development process. Presently there are numerous methods available, ranging from rigid, plan-driven, process-oriented ones to more flexible, people-oriented agile approaches.

Plan-driven methodologies, as the name itself suggests, are concentrated on the planning aspects of the software development. These "*methods approach development in a requirements/design/build paradigm with standard, well-defined processes that organizations improve continuously*" (Boehm & Turner, 2003). Examples of such methodologies include waterfall process methodologies and SW-CMM (Software Engineering Institute, 1993). On the other end of the spectrum lie *agile methodologies*. As stated in the Agile Manifesto (Agile Manifesto, 2001), which outlines the principles behind this group of methodologies, in the agile approach working software, collaboration and flexible responding to change are valued more than the documentation, processes and following plans. Scrum (Schwaber, Beedle, 2001) and eXtreme Programming (eXtreme Programming, 1999) are examples of such methodologies. In between the two extremes, *hybrid methodologies* can be identified. Looking at methodologies as a collection of practices, activities and guidelines, they identify the potentially beneficial elements in both plan-driven and agile approaches and combine then in a new way.

In this thesis, a term *practice* is used to describe "*a collection of concepts, principles, methods, and tools*" (Pressman, 2009), which may origin from a software development method, combining its suggested activities into a practical purpose. More precisely, following Päivärinta's and Smolander's conclusions (Päivärinta and Smolander, 2015), "*in the software development context, practices may include both thoroughly organized use of predefined development methodologies and loosely organized and even emergent activities that may use individual tools or techniques at hand*". What is important, a practice "*populates a software process model with the necessary technical and management how-to's to get the job done*" (Pressman, 2009).

A *process model* of software development reflects the methodology chosen for a given project, organizing the project in a more rigid or a more flexible way. The core traditional disciplined model is called *the waterfall* (Royce, 1970). Its name stems from the waterfall-like flow of the activities performed in a sequential way. It has been introduced in the 1970's and for decades has been treated as a state of the art. However, it has become infamous over the years, as its rigidness and heavyweight documentation was becoming increasingly inadequate for the industry needs (Pressman, 2009). A need for an updated approach to the software development process resulted in new models being proposed. More modern models include *the V-model* (Forsberg & Mooz, 1991) and *the spiral model* (Boehm & Hansen, 2000). The former is an updated version of the waterfall model, with an additional stream for quality assurance actions, forming a V-like shape when presented on a diagram. The latter, is an evolutionary process model, combining the iterative development with a more disciplined approach, trying to balance the two. Finally, the agile process model focuses on the incremental and iterative development, reducing the initial planning and analysis stages known from the other process models.

It is widely accepted that there is no silver bullet as far as software development is concerned and no one-suits-all development methodology (Brooks, 1987). Both, agile and plan-driven approaches have their strengths and weaknesses depending on the specific project domain.

The foreseen application domain of a system is an important factor influencing the decision on the choice of software development method for a given project. Some domains, such as safety-critical systems domain, have such important and specific constraints and requirements, that they become crucial in deciding, which of the software development methods to use. Safety-critical systems are systems "*whose failure might endanger human life, lead to substantial economic loss, or cause extensive environmental damage"* (Knight, 2002). Such systems can be found in transportation, health care, energy industry and many other fields where certain critical responsibilities are transferred to technological solutions. Potential harm in case of the system malfunction can have a profound impact on its environment and context of use. For this reason, safety-critical systems are subject to numerous regulations and standards, which advise what to do to ensure that the final software is acceptably safe. Some standards also regulate the development process and define its necessary actions and artefacts.

In recent years a growing competition in the IT market has also influenced the safety-critical domain. With widespread use of software, for example in cars or planes, everyone is a potential user of safety-critical software, which naturally changed the perspective for companies providing such software. For example, when it comes to

medical software companies, they have evolved from supporting only hospitals and doctors to providing personalized e-health software solutions and equipment for individual patients. It has become crucial to offer better software, more appealing to a user while keeping cost as low as possible in order to compete in this fast changing and growing market. Consequently, there is a strong demand for increasing efficiency of software development processes (in terms of effort and time) while still respecting the safety requirements imposed by relevant standards and regulations.

1.2. PROBLEM STATEMENT AND THESIS PROPOSITION

The goal of the research described in this work was to develop an approach aimed at facilitating the introduction of a more agile approach to the software development process, depending on the characteristics of the project, while maintaining compliance with the required safety standards and regulations, and the AgileSafe method presented in this thesis is the main result of this research.

The thesis proposition is formulated as follows:

The proposed method makes it possible to reduce the effort devoted to software development processes in safety related projects, without compromising the requirements of applicable norms and standards.

1.3. RESEARCH APPROACH

The research presented in this dissertation included the following steps:

- ***Step I.*** Analysis of the recommended and currently applied (rigorous) practices of developing software for critical applications.
- ***Step II.*** Analysis of the recommended and currently applied agile practices for software development and their impact on software development performance
- ***Step III.*** Analysis of the constraints resulting from the present regulations and standards related to the safety-critical software domain (with a particular focus on medical devices)
- ***Step IV.*** Development of a new, hybrid approach – the AgileSafe method - which is the main contribution of this research.
- ***Step V.*** Experimental identification of candidate agile practices to be included in the scope of the proposed method.
- ***Step VI.*** Development and integration of tools supporting AgileSafe in order to demonstrate the method.
- ***Step VII.*** Selection of criteria for the evaluation of the proposed method.

Step VIII. Evaluation of AgileSafe – by case studies and assessments with active expert involvement.

1.4. THESIS OUTLINE

The thesis is organised as follows:

In Section 2 the domain of safety-critical software has been analysed. The context has been presented as well as the problems this domain is facing and some attempts to answer these problems have been investigated.

In Section 3 the works related to this research have been analysed, outlining the motivations and proposed solutions.

In Section 4 an overview of the AgileSafe method, the outcome of this research, has been presented. Main elements and algorithms have been described along with the ideas for ensuring safety level while introducing some new optimising practices.

In Section 5 the Project Analysis and Practices Selection Processes, the elements of AgileSafe method, has been described in more detail. It was explained what are the basic incentives for the project analysis and for the selection process.

In Section 6 a more detailed description of the AgileSafe Knowledge Base has been presented. A structure of the Knowledge Base was outlined along with the specification of its elements and the Suggestion Algorithm.

In Section 7 Assurance Arguments have been introduced as well as their use in the AgileSafe method. They form a crucial part of the monitoring of safety and conformance of the project.

In Section 8 the process of Evaluation of the method and its results have been described.

.

2. SOFTWARE DEVELOPMENT OF SAFETY-CRITICAL SYSTEMS

The key to distinguish a safety-critical system from a non-critical one is the consequences of failure (Knight, 2002). A safety-critical system can be defined as "*computer, electronic or electromechanical system whose failure may cause injury or death to human beings*" (Palanque et al., 1998). Such systems can be found in medical devices, aircrafts, military equipment, nuclear plants – in domains, where human health or life depends on the correct operations of a system.

2.1. SAFETY-CRITICAL SYSTEMS DOMAIN

With an intensive progress and expansion of technology in the 20th and 21st centuries, devices and solutions in many domains have become increasingly software reliant. Computers have been able to execute tasks that used to rely on humans or were not possible before due to the human limitations. However, the responsibility for the outcome of these computerised actions have been transferred to another human beings – the creators of such systems.

In order to ensure that the system is safe to use in its destined environment, the concern about its safety begins at its conception and lasts throughout its lifecycle. The system should be analysed in terms of potential risks, with risk being understood as "*a possibility of loss, damage or disadvantage*" in terms of the software development and final operating in its environment (Miler and Górski, 2001). There are activities and techniques, presented through the years that can be performed in order to analyse the potential failures thus increasing chances of avoiding them. For example, an assessment of hazards can be implemented, traditionally using methods such as Fault Tree Analysis (FTA), Failure Mode Effect Analysis (FMEA) etc. (Smith and Simpson, 2016). Nevertheless, the sole fact of using such methods does not ensure that the required level of safety will be met.

As with many engineering domains (Kelly, 1998), it has been the failures that provided the incentive for more thorough software safety policies. Procedures for airborne software are still subject to updates, especially after dangerous malfunctions such as Airbus A340-642 accident with fuel control and monitoring computer and its warning system (AAIB, 2005) or Boeing 777-200 software component analysing data from a known faulty accelometer (ATSB, 2005). A very explicit example of a disaster caused by a software error is the case of Therac-25, a radiation therapy machine (Leveson, 1995). Due to the malfunction of its software several people have died or been disabled. Policies concerning medical software in the United States have been updated based on this incident (Leveson, 1995). Nevertheless, software accounts for

approximately 25% of adverse events (Jones, Górski, 2017) recorded in the Manufacturer and User facility Device Experience database (MAUDE, 2017). Still, if an algorithm implemented in the software for insulin infusion pump used by a diabetic miscalculated the patient's insulin dose, which is then successfully administered to the patient's body it could lead to loss of health or even death (Chen, Lawford, Wang, 2014). The computers and software are more commonly used in medicine than most people realize (Knight, 2002). It is of great importance to ensure that the solutions the software companies deliver are safe enough to operate in the target environment without causing harm.

The safety-critical software domain will likely expand in the future, with dropping cost of hardware and new possibilities presented by the software (Knight, 2002). As such, the importance of safety assuring solutions will grow, especially with time and cost playing an increasingly important role in this domain.

2.2. SAFETY REQUIREMENTS OF SAFETY-CRITICAL SYSTEMS

The increasingly widespread use of computer systems in safety-critical domains has led to the analysis and research concerning the safety assurance for such systems. This section outlines the requirements imposed on safety-critical systems by selected guidelines, norms and standards, which aim at assuring the necessary level of safety.

2.2.1. Regulations and standards in safety-critical domain

With the technological progress some new domains have been born, such as nuclear or automotive industries and the new areas became a catalyst for researching advanced computer-based solutions. Such solutions needed to increase the safety of the products rather than pose another potential threat. For this reason, nuclear industry was one of the first advocates for safety assurance solutions (Bloomfield, Bishop, 2010).

In 1970's the European Working Group on Industrial Computer Systems (EWICS) started to work on a series of guidelines and a collection of best practices for development of safety related systems (Bloomfield, Bishop, 2010). This work contributed to the IEC 880 standard on software for nuclear systems (IEC, 1986). Since then, an increasing number of domains adapted the idea of a common safety regulation and many standards have been issued.

The International Organization for Standardization (ISO) defines standard as *"a document that provides requirements, specifications, guidelines or characteristics that can be used consistently to ensure that materials, products, processes and services are fit for their purpose"* (International Organization for Standardization, 2017).

Standards are issued by internationally recognized bodies, for example the aforementioned ISO, International Electrotechnical Commission (IEC), Institute of Electrical and Electronics Engineers (IEEE), European Committee for Electrotechnical Standardisation (CENELEC). International standards have often their counterparts at the national level, for instance UK British Standards Institute (BSI), Polish Committee for Standardization (PKN) and others.

The subjects of standardization range from generic concepts such as quality management i.e. ISO 9000 family (International Organization for Standardization, 2015c), which are applicable in various domains, to domain-specific, presenting more detailed requirements like ISO/TS 16949 (International Organization for Standardization, 2009), which concerns quality management for automotive production. The IEC 61508 standard (International Electrotechnical Commission, 2010), concerning functional safety, has also been the generic base for more domain specific standards. Standards may also address specific issues considered particularly important, an example is ISO 14971, the standard on risk management for medical devices (International Organization for Standardization, 2007). Other standards can be more product-oriented, examples include the IEC 60601-1-11 (International Organization for Standardization, 2015) standard, which concerns medical electrical equipment.

A complementary form of control over safety-critical systems are the criteria imposed by the national or regional organisations, which declare the conditions of admission to the local market. If a manufacturer wishes to introduce his/her product to the specific market and the product is a subject to local regulations, she/he needs to prove compliance with these regulations. In European Union such regulations function as European Norms (EN) and, in essence, are often standards interpreted into norms by the CENELEC. In the United States the Food and Drug Administration (FDA) provides a list of guidance and regulatory requirements for specific products (Fda.gov, 2017).

2.2.2. ***Regulations and safety requirements for medical devices***

E-health is a domain of high potential of bringing added value to the stakeholders. The ecosystem of health-related electronic devices prospers in hospitals, homes, and pharmaceutical companies and has safety relevance in many contexts. Software is becoming an indispensable component of medical devices and in most cases, they can be considered as being software intensive. The market of the suppliers of medical devices and e-health services is growing, as is the demand for the related software. Most of such products need to be compliant with appropriate standards e.g. GAMP 5 (International Society for Pharmaceutical Engineering, 2008), ISO 13485 (International Organization for Standardization, 2016), IEC 62304 (International Organization for

Standardization, 2006), and others. For this reason, the medical safety-critical software domain was chosen for demonstration and calibration of the AgileSafe method proposed in this thesis.

Medical certificates and standards concern both hardware and software parts of the product. This thesis concentrates on the standards that apply to the software components of medical devices, specifically ISO 14971:2007 *Medical devices -- Application of risk management to medical devices* (International Organization for Standardization, 2007) and IEC 62304:2006 *Medical device software -- Software life cycle processes* (International Organization for Standardization, 2006). All of the further mentions of ISO 14971 and IEC 62304 will refer to these versions. While the latest version of ISO 14971 is EN ISO 14971:2012, it applies only to European market and what is more, the core text of the standard stayed unchanged with only three added annexes. For this reason, the version of reference in this thesis is the ISO 14971:2007. When it comes to IEC 62304, in 2015 an amendment to the standard was issued, with details concerning legacy systems (International Organization for Standardization, 2015b). Due to its limited impact on the core text of the standard, in this thesis the IEC 62304 version of reference is IEC 62304:2006.

2.2.3. ***Assurance arguments***

Along with the evolution of safety guidelines and standards in 1970's and 1980's a need for a method to demonstrate conformance with these requirements emerged. The idea of presenting logically organised declarations of compliance with a supporting evidence material has been developed and eventually a concept of safety case has been introduced (Bishop, Bloomfield, 1995). Among various methods of representation of such safety cases the argument representation is the one, which has gained most popularity, hence the use of the term safety arguments in some of the later works (Ge, Paige, McDermid, 2010; Greenwell, Knight, Holloway, Pease, 2006) and more general assurance arguments (Ankrum and Kromholz, 2005; Stephenson, McDermid, Ward, 2006; Weinstock and Goodenough, 2009), used as a wider term.

Such arguments are built using a specific notation. Some of the most acknowledged approaches to argument structure are based on the ideas proposed by Stephen Toulmin in 1958 (Toulmin, 2003). Toulmin's model of argument organises reasoning supporting a good argument using a set of claims, warrants and data. Examples of structures based on this model are Trust-IT (Górski, 2005; Górski et al., 2005; Górski, 2007), Goal Structuring Notation (GSN) (Kelly, 1998) and Claims-Arguments-Evidence (Bishop, Bloomfield, 1998).

3. RELATED WORK

While plan-driven methodologies have proven their value and usefulness in safety-critical projects, the evolving market of software products of the last few years puts this approach to the test (Ge, Paige, McDermid, 2010). Growing competition, ever-changing technologies and more diverse groups of customers have changed the expectations towards software development methods. The need to deliver systems of acceptable quality, faster and at lower cost in comparison to competitors evoked seeking an alternative (Petersen, Wohlin, 2010).

3.1. MOTIVATIONS FOR NEW APPROACHES IN DEVELOPMENT OF SAFETY-CRITICAL SOFTWARE

Medical devices are nowadays highly software intensive and the suppliers of such devices may need to be compliant with various standards and certification programs. At the same time medical software is used in diverse environments extending the range of potential stakeholders from hospitals and medical experts to regular patients and e-health services users. With growing competition in the domain, fast paced changes in technology, and customers demanding innovations as well as highest safety standards, medical software companies are motivated to employ hybrid approaches where agility is combined with necessary safety assurance.

3.1.1. Agile practices and reduction of effort

Agile methodologies have grown in popularity since the presentation of the Agile Manifesto in 2001 (Agile Manifesto, 2001). They were introduced as an alternative to plan-driven and waterfall methodologies, which were considered as being too restrictive in some circumstances, in particular, while dealing with volatile requirements and ever-changing market demands. In such situation, heavyweight documentation and low flexibility associated with plan-driven approach could have an impeding effect on a software development process (Boehm, Turner, 2003; Ge, Paige, McDermid, 2010).
In response to these concerns agile methodologies have offered practices which value close relationship with customers, allow more relaxed approach towards documentation and provide a flexible development lifecycle based on short iterations (Abrahamsson et al., 2002). Successfully combined, agile practices can potentially reduce the cost of production as well as time to market (Drobka, Noftz, Raghu, 2004; Lindvall et al., 2004)

According to the CHAOS report (Standish Group, 2015) from years 2011-2015, projects, which followed agile methods, resulted in 39% success rate as opposed to 11% with projects following waterfall approach. Moreover, only 9% of the agile projects failed,

while the waterfall projects failed in 29% of the cases. Interestingly, the higher success rate for agile projects is even more remarkable when it comes to medium and large projects than it is for smaller ones.

What is more, VersionOne in its 10th State of Agile Annual Report presents the benefits of agile according to their respondents, based on 3,880 responses collected (VersionOne, 2016). With respect to our research the most interesting results were those concerning reduction of effort: 85% of respondents noticed increased team productivity, 80% faster time to market and 79% enhanced software quality when they introduced agile in their projects.

3.1.2. ***Research on hybrid approaches***

While agile approach has gained an almost immediate acclaim among SMEs and companies involved in non-critical projects, they were received with much distrust by larger companies and software engineers engaged in long-term and/or safety-critical projects. In case of such projects, the predictability and stability of plan-driven methodologies can bring expected profits (Paige, Charalambous, Ge, Brooke, 2008) by facilitating the certification processes and establishing a repetitive quality. The up-front analysis and rigorous documentation provide valuable foundations for further risk management and process traceability (Boehm, 2002). What is more, some companies have years of experience in managing their projects following plan-driven practices and therefore they have acquired the know-how that increases the trust towards this approach (Ge, Paige, McDermid, 2010; Rottier, Rodrigues, 2008).

For many years it has been prejudicially believed that agile methodologies are at odds with maturity models and certification schemes (Glazer et al., 2008). This point of view might stem from the outdated notions based on inflexible models such as Capability Maturity Model (replaced with Capability Maturity Model Integration (Software Engineering Institute, 2006)) and an oversimplification of agile values (Glazer et al., 2008). In fact, present-day maturity models provide more freedom than their predecessors while agile methodologies are not about the lack of documentation and recklessness.

Attempts to combine the best of the two approaches, the agile and the disciplined one, have been presented as early as in 2003 (Boehm, Turner, 2003) and reflected a global discussion about the need and applicability of such hybrid methodologies. As a result, some models adapting agile practices into maturity models such as CMMI have been introduced since then (Fritzsche, Keil, 2007; Marçal et al. 2008; Diaz, Garbajosa, Calvo-Manzano, 2009; Bulska, Miler, 2010) providing new possibilities for companies engaged in long-term and critical projects. What is more, case studies and evidences of

successful applications of such balanced approaches have been provided as well (Glazer et al., 2008; Lindvall et al., 2004; Poppendieck, Poppendieck, 2003; Babuscio, 2009; Potter, Sakry, 2009; Pikkarainen, Mantyniemi, 2006).

Agile methods are not immediately associated with risk management (Siddique, Hussein, 2016). One of the foundations of the Agile Manifesto is to reduce the documentation only to the necessary scope which seems to imply that there is no place for extensive analysis and risk assessment reports. However, a lack of separate risk analysis stage in the agile methods does not mean that the idea of risk itself is ignored. It is worth mentioning that the concepts of incremental software development and close cooperation with customers are in their essence the means to minimize the risk imposed by volatile requirements (Nyfjord & Kajko-Mattsson, 2008; Siddique, Hussein, 2016). Many agile methods were in fact proposed as mechanisms reducing the risk that burdened projects led with plan-driven methodologies and thus claim to be risk-driven (Nyfjord, 2008).

While many risk management practices were primarily designed for plan-driven approach and cannot be fully transferred into the agile ground, agile methods are flexible enough to accommodate the necessary amount of additional ceremony and indeed can benefit from a more structured approach to risk (Nyfjord, 2008; Nyfjord & Kajko-Mattsson, 2008; Gary et al., 2011). Having said that, the word „necessary" is crucial here. Because in the agile approach there is no space for „just in case" or „maybe we will need this", the risk management practices have to be carefully chosen to cover just what is needed (Gary et al., 2011).

After years of plan-driven methods being used as default it may be difficult to notice that in fact most standards do not explicitly require the use of such methods. The language and scrutiny represented in guidelines indeed suggest that the plan-driven way is the smoothest way to comply, nonetheless it is not a requirement. In fact, in 2012 FDA recognized the AAMI TIR45:2012 - *Guidance on the use of AGILE practices in the development of medical device software* (Association for the Advancement of Medical Instrumentation, 2011). It concludes that agile practices can be successfully used in safety-critical software development and that such practices can be compliant with IEC 62304 standard. It also provides a mapping between agile methods and IEC 62304 activities (Association for the Advancement of Medical Instrumentation, 2011).

3.2. MODELS FOR ADOPTING AGILE APPROACH TO SAFETY-CRITICAL DEVLOPMENT

There is a growing body of evidence supporting the theory that incorporating agile practices into safety-critical projects is not only feasible but also potentially profitable. In 2003 Alleman et al. (Alleman, Henderson and Seggelke, 2003) presented an approach combining eXtreme Programming practices (eXtreme Programming, 1999) with Earned Value Management (Earned Value Management, 1967) that was successfully implemented in a government contracted project. In their article they described their experiences and improvements obtained by using this approach, although the approach itself was not sufficiently illustrated and little was mentioned about which features of the product and its certificates had influenced the choice of practices.

Consecutive reports were more specific about successful implementations of their hybrid approaches. Rasmussen et al. (Rasmussen et al., 2009) described the application of a tailor-made agile approach in a Food and Drug Administration (FDA) regulated project in Abbott Company (Abbott, 2017). As a result of the rapid expansion of the market, which demanded responding to the changing requirements as well as reducing production cost, the Abbott Company decided to employ a software engineering organization AgileTek (AgileTek, 2011), which developed the Agile+ solution. The Abbott Company managed to improve their processes and achieved the financial goals by introducing more agile approach while still maintaining the acceptable level of FDA safety assurance. Unfortunately, because of a commercial aspect of the Agile+ it was only briefly described in the article, making it difficult for other companies to benefit from Abbott experiences without external help.

Another interesting case study was presented in an article by Petersen and Wohlin (Petersen, Wohlin, 2010) in which they described the application of agile practices at Ericsson AB, which is certified with ISO 9001:2000. While the project in question was not safety-critical, the need to follow the requirements of the standard imposed significant constraints. They focused on comparing how the perceived impediments have changed owing to the introduction of agile practices. The improvements were noticeable as most of the concerns raised in relation to the plan-driven methodologies were alleviated as well as the number of perceived impediments was reduced (Petersen, Wohlin, 2010).

The increasing number of reports suggesting that adapting agile practices to suit safety-critical processes can bring measurable profits provoked the need for a model of such adaptation. In the literature some attempts to propose such models can be found. Weiguo and Xiaomin (Weiguo, Xiaomin, 2009) presented an approach suitable for FDA

compliant medical devices projects. Their method was based on an idea of combining an incremental character of code development with a classic, waterfall-like way of preparing project documentation. Unfortunately, the model was insufficiently described, and we know very little about its possible implementations.

Stephenson, McDermid and Ward (Stephenson, McDermid, Ward, 2006) presented another model for tailoring agile practices to suit safety-critical systems. They called it the Agile Health Model and it was built around the idea of modular structures and risk management techniques known from plan-driven methodologies. However, the model was in the preliminary stage, introduced mainly in order to prove the possibility of applying agile practices into safety-critical projects, and no case study was provided as well.

A more complex and well-described model was presented by Paige et al. (Paige et al., 2008). They collated agile and safety principles and demonstrated the key challenges that need to be addressed when formulating a hybrid approach. Their main concerns were the different approaches towards communication, documentation, customer participation, multiple-domain engineering, testing, and incremental development. Indeed, as much as agile methodologies value an active customer participation, in safety-critical systems development it is often impossible to keep in touch with every group of stakeholders, in particular if we take external certification organizations into consideration as well. Testing and incremental development, both crucial to the agile approach, are also difficult to reconcile with safety requirements. Agile testing strategy is based on unit and black-box tests while in order to satisfy certification bodies it is important to incorporate costly and time consuming white-box and acceptance tests. What is more, the incremental product development, one of the key attributes of all agile methodologies, can impede the process of certification and preparation of safety arguments as they should be addressed up-front with all of the requirements and risks known beforehand.

Paige et al.'s solution concentrates on the following ideas: pair-programming of software and system engineers, the introduction of risk management techniques, usage of tools for generating documentation from source code and tackling the incremental development by using "pipelined iterations" consisted of the minor and major iterations with acceptance tests at the end of every major one. Their model was implemented in a case study in which an Integrated Altitude Data Display System (IADDS) for aeroplanes was developed. As a result, their approach was put to the test as their solutions proved to be insufficient in some aspects. They concluded that although *"XP and Aps [agile practices] in general were not designed with safety-critical systems development in mind, they can be adapted to that sort of development*, [...] *it is rather unlikely that level A*

software can be produced in the near future with the modifications made to the process so far" (Paige et al., 2008).

Another relevant approach is AV-Model (McHugh, McCaffery and Coady, 2014) combining the traditional V-Model with Scrum and focusing on medical device software development and the IEC 62304 standard. While the AV-Model present some promising solution, its focus as well as potential applications are restrained and as such cannot be universally recommended.

A more comprehensible and practical solution has been proposed by a joint research group of SINTEF (SINTEF, 2017) and the Norwegian University of Science and Technology (NTNU, 2017). They proposed a method called SafeScrum (Mycklebust, Stålhane and Hanssen, 2016), which concentrates on adapting Scrum into safety-critical software development. The method has been already applied in real life projects (Stålhane, Myklebust and Hanssen, 2013). The standard that has been mainly used in conjunction with SafeScrum is IEC 61508 (International Electrotechnical Commission, 2010) (Hanssen, Myklebust and Stålhane, 2012), which is focused on functional safety. The safety-oriented SafeScrum practices include Backlog Splitting, Backlog Refinement and Quality Assurance Role (Mycklebust, Stålhane and Hanssen, 2016). While the whole method is very promising, it may need various adjustments when applied to other standards. What is more, it does not provide any actual tools to control the conformance with standards.

3.3. RECONCILING SAFETY REQUIREMENTS AND THE STREAMLINING OF PROCESSES

While process optimization is vital to the business and economical part of a software development project, in the safety-critical software domain its profits will not be visible unless a company is able to conform to standards and guidelines, which regulate a particular industry. Clients demand their products to be of high quality, on time and within a reasonable budget but at the same time the software need to be certified by an appropriate authority in order to be approved for use in its destined environment. For this reason, in safety-critical software domain it is not enough to streamline the process and make people work in a more efficient way, it is not enough to prove the financial profits to the company – a mechanism needs to be employed to ensure that the safety level did not suffer in the process.

Attempts to provide a hybrid, disciplined-agile, approaches bringing together best of the two worlds are already in effect for several years. A growing body of evidence, including industrial reports, shows that obtaining the right balance is doable and

profitable especially if the companies decide to employ competent experts to develop a custom-made approach.

While these models of adapting agile practices to suit safety-critical projects are valuable sources of knowledge there is still a need to develop an easier to use and thorough set of guidelines for safety-critical software companies, that would like to adapt agile practices into their project development. AgileSafe method, which is the result of this doctoral research, is an attempt to provide such solution, a more comprehensible one than the analysed approaches and enabling both safety assurance and process effectiveness at the same time.

In order to ensure that the required (by the selected regulatory documents) safety assurance has been built into the Project, AgileSafe uses assurance cases. The main idea is to provide assurance cases for both, the software development process and for the end product itself. While the latter is the essence of demonstrating product conformance with a given standard or a guideline, the former is a mean to obtain it. It is a technique that will allow a company to ensure the practices they're choosing are suitable for this particular project with its safety requirements imposed by standards.

The method has been designed to be applicable to various safety-critical domains. In this thesis we concentrate on the medical software domain, but it can easily cater for other fields. In order to keep it independent from the type of product being produced in the analysed process, the assurance arguments are based on the standards structure. It means that instead of treating specific product-related list of risks as a base for an assurance argument, it is being built upon the applicable standard and then specified for the given project. The motivations behind this are explained more in-depth in the section 7.

4. AGILESAFE OVERVIEW

In the previous chapter the motivations behind seeking new solutions for development of safety-critical software have been described. With growing competition in the domain, fast paced changes in technology and clients demanding innovations as well as the highest safety standards, safety-critical software companies are tempted to employ hybrid approaches where agility is combined with necessary safety assurance. In this research we attempt to answer their needs by introducing AgileSafe method.

4.1. AGILESAFE MAIN CONCEPTS

AgileSafe is applied with respect to a chosen software development project (the Project). Most of the activities related to the method are performed in the preparation stage of the Project. The additional workload imposed on the development stage is kept to minimum and focuses on the regulatory requirements as well as the practices introduced to the Project. Most of the elements of the method, once prepared, can be reused or adjusted later for other projects.

There are two main uses of AgileSafe. The first and fundamental one is applying AgileSafe and obtaining an advice on software development process, with suggestions on which practices to use and how to assert conformance with selected standards. The second way is improving the method by updating the knowledge stored in the method, providing the feedback and information about new practices.

The high-level use case diagram of AgileSafe is presented in Figure 1

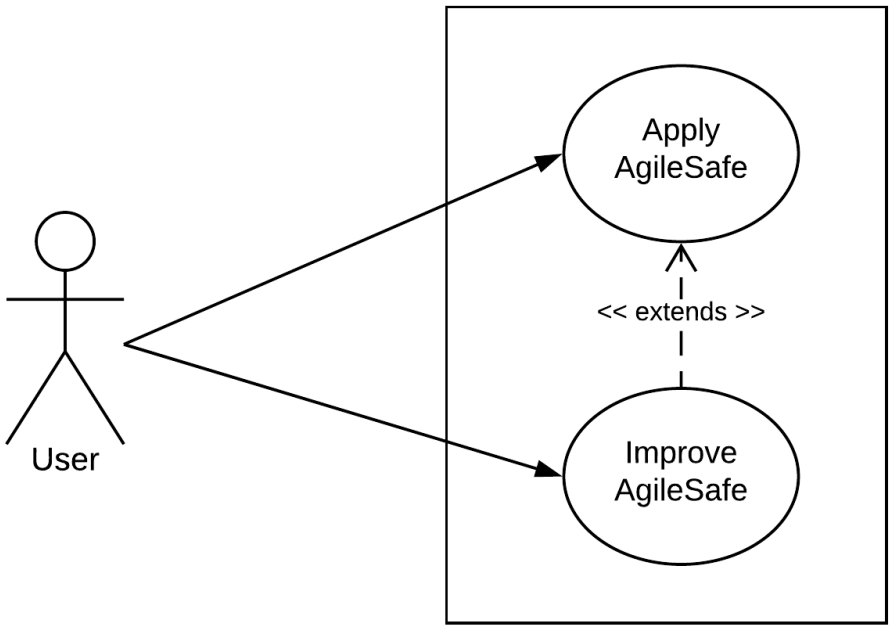


Figure 1 The high-level use case diagram of AgileSafe

It is assumed that User is a person or a team with a good knowledge of the Project, in which the selected practices are to be implemented. He or she needs to be able to specify the characteristics of the Project as well as to decide upon the final set of practices. The suggested practices will carry some information that should facilitate the decision but in order to increase a chance of success of the AgileSafe implementation the User need to have a good knowledge of the software development process as well.

In order to follow the *Improve AgileSafe* use case, User should be a person with good knowledge of the standards and the safety aspects of software development.

Taking all of the above into consideration, the User could be a Project Manager, Process Engineer, Scrum Master (or the whole Team), RAMS Engineer or similar roles, depending on the company applying AgileSafe method.

The use cases are introduced in the tables below:

Table 1.
Apply AgileSafe use case: general description

1.	**Apply AgileSafe**
Description	The information about Project and about the regulatory context constraining the Project and its product are the inputs to the method. User needs to specify the characteristics of the Project and the regulations the Project needs to comply with. Based on these, the User is guided through the two main processes of AgileSafe: process which selects the specifications of software development practices to be applied in the Project and a process which results in the set of assurance arguments corresponding to the regulations included in the regulatory context. The two main processes of AgileSafe reflect the main objectives of AgileSafe: to support a hybrid approach to software development based on the tailored practices and to support continuous monitoring of conformance to the mandatory regulatory requirements.

Table 2.
Improve AgileSafe use case: general description

2.	**Improve AgileSafe**
Description	In order to further improve the method and tailor its advice to the User's needs more accurately, the knowledge stored in the method should be reviewed and updated regularly. User can introduce new software development practices to the pool of the practices from which the candidate practices are selected. They can also be added to the AgileSafe assurance arguments. User can also add another standard to the assurance arguments.

A more detailed description of these use cases is given in the following paragraphs.

4.1.1. ***Use case: Apply AgileSafe***

A more detailed structure of the *Apply AgileSafe* use case is presented in Figure 2, following the UML use case notation.

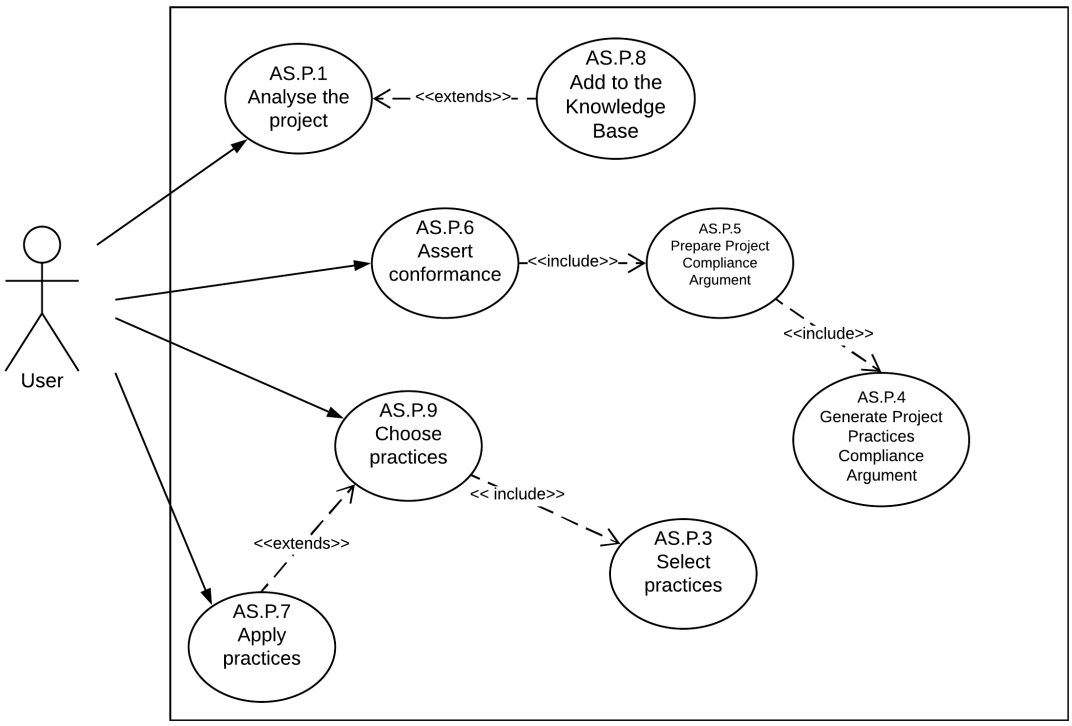


Figure 2 A detailed diagram of *Apply AgileSafe* use case

For the use cases shown in the following figures we apply the naming convention that each use case is unambiguously identified by the identifier AS.P<n>, where n is a natural number.

The use cases presented in the Figure 2 are the core processes of AgileSafe method. In order to apply AgileSafe in a Project, the User should initialize the use cases in the following order:

AS.P.1 Analyse the Project

AS.P.6 Assert Conformance

AS.P.9 Choose practices

AS.P.7 Apply practices.

Further explanation of their application is presented in the data flow diagram of Figure 4

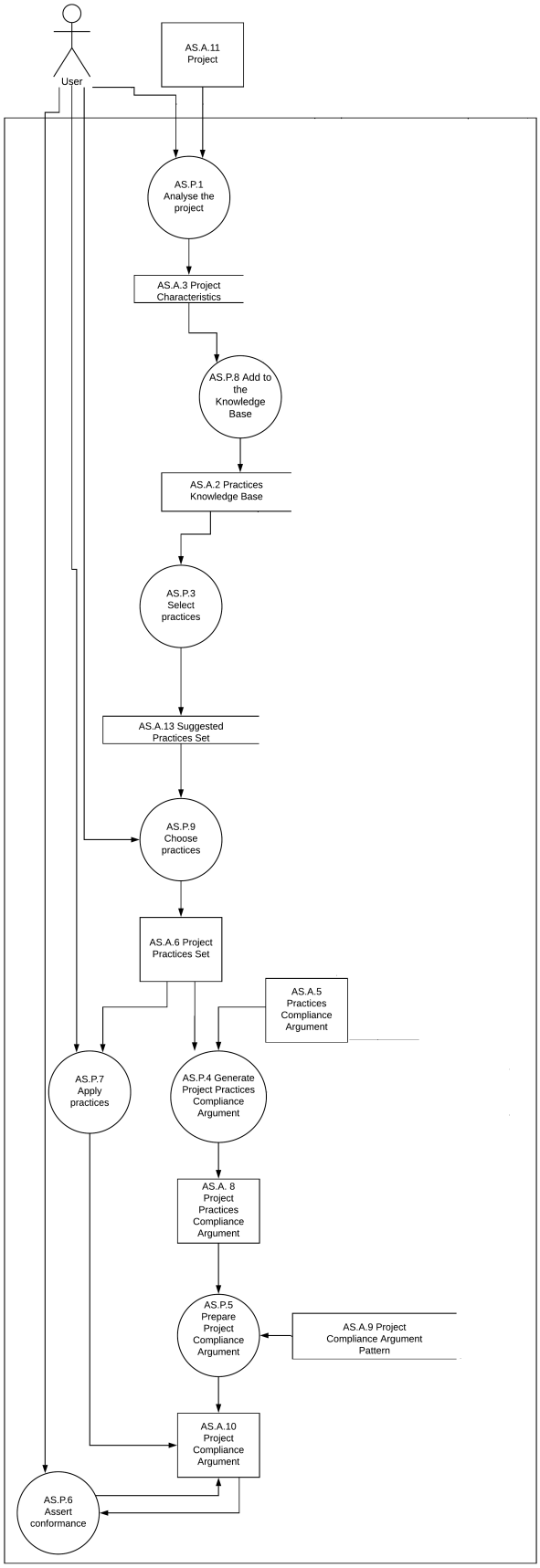


Figure 3 Data flow diagram of *Apply AgileSafe* use case

In the dataflow diagram of Figure 3, the User first analyses the project (*AS.P.1*) which results in *AS.A.3 Project Characteristics*. These characteristics are then added (*AS.P.8*) to the *AS.A.2 Practices Knowledge Base*. The *Practices Knowledge Base* provides input to the *AS.P.3 Select practices* process which presents to the User as *AS.A.13 Suggested Practices Set*. From these suggested practices the User can *AS.P.9 Choose practices* for his *AS.A.6 Project Practices Set*. And the resulting *Project Practices Set* is being implemented in the software development process (*AS.P.7 Apply Practices*).

AS.A.5 Practices Compliance Arguments should be adapted (*AS.P.4*), depending on the *AS.A.6 PPS,* into *AS.A.8 Project Practices Compliance Arguments*. Based on them, the *AS.A.10 Project Compliance Arguments* are prepared (*AS.P.5*) and they are the end products of the method allowing the User to *AS.P.6 Assert conformance*, using the evidence prepared during the *AS.P.7 Apply practices process*.

The artefacts used in AgileSafe have been organised into three categories:

- Input – the artefacts, which are supplied by the user and are the main variables in the method
- Method framework – the artefacts maintained by the method
- Output – the artefacts prepared as a direct result of AgileSafe application

The specific elements of a detailed diagram of *Apply AgileSafe* use case (Figure 2) are described in the tables below.

Artefacts:

Table 3.
AS.A.2 Practices Knowledge Base

AS.A.2	**Practices Knowledge Base**
Type	Method framework
Description	A knowledge base of software development and management practices, both agile and disciplined, created for the needs of projects that use AgileSafe.

Table 4.
AS.A.9 Project Compliance Argument Pattern

AS.A.9	**Project Compliance Argument Pattern**
Type	Method framework
Description	A template for developing AS.A.10 Project Compliance Argument, the main argument to be used for monitoring conformance with a given standard.

Table 5.
AS.A.3 Project characteristics

AS.A.3	**Project characteristics**
Type	Method framework
Description	The characteristics, which contain sufficient knowledge to decide on the most suitable software development practices for the project.

Table 6.
AS.A.6 Project Practices Set

AS.A.6	**Project Practices Set**
Type	Output
Description	The resulting custom set of software development practices for a given project.

Table 7.
AS.A.8 Project Practices Compliance Argument

AS.A.8	**Project Practices Compliance Argument**
Type	Output
Description	A tailored version of *AS.A.5 Practices Compliance Argument* containing only practices relevant to a given project.

Table 8.
AS.A.10 Project Compliance Argument

AS.A.10	**Project Compliance Argument**
Type	Output
Description	The main argument to be used for monitoring conformance with a given standard. Contains the evidence produced in the process of *AS.P.7 Applying practices* presented in the *AS.A.8 Project Practices Compliance Argument* for the given project.

Table 9.
AS.A.11 Project

AS.A.11	**Project**
Type	Input
Description	The software development project which is a subject to the AgileSafe practices selection and safety monitoring.

Table 10.
AS.A.13 Suggested Project Practices Set

AS.A.13	**Suggested Project Practices Set**
Type	Output
Description	The list of practices suggested by the method algorithms for the given project.

Processes:

Table 11.
AS.P.1 Analyse the project

AS.P.1	**Analyse the project**
Description	Investigating the project characteristics.
Steps	1. Identify the project that will be deployed throughout the process. 2. Gather all the available information about the project, concentrating on the standards the project needs to comply with and the circumstances in which the project will be developed. 3. Document the gathered information using the *AS.A.3 Project characteristics* form.

Table 12.
AS.P.3 Select practices

AS.P.3	**Select practices**
Description	Selecting the most suitable practices for a given project.
Steps	1. The *AS.A.3 Project characteristics* for a given project are presented as input for the method practices selection algorithm. 2. [OPTIONAL] If you wish to introduce a new Practice to the *AS.A.2 Practices Knowledge Base*, follow the *AS.P.3.1 Update Practices* sub process. 3. The practices selection algorithm analyses the *AS.A.3 Project characteristics* and as a result an *AS.A.13 Suggested Practices Set* is presented.

Table 13.
AS.P.4 Generate Project Practices Compliance Argument

AS.P.4	**Generate Project Practices Compliance Argument**
Description	Customizing the appropriate *AS.A.5 Practices Compliance Argument* to the needs of a given project. This process should be carried out for each standard the project is expected to conform with.
Steps	1. Select the appropriate *AS.A.5 Practices Compliance Argument*, depending on the standard you want to conform with. 2. Narrow down the list of practices in the *AS.A.5 Practices Compliance Argument* to those selected for the *AS.A.6 Project Practices Set*. 3. Based on *AS.A.7 Project Practices Compliance Pattern* and the edited *AS.A.5 Practices Compliance Argument* prepare the *AS.A.8 Project Practices Compliance Argument.*

Table 14.
AS.P.5 Prepare Project Compliance Argument

AS.P.5	**Prepare Project Compliance Argument**
Description	Developing an *AS.A.10 Project Compliance Argument*, which will demonstrate project's conformance with a specific standard. This process should be carried out for each standard the project is expected to conform with
Steps	1. Select the appropriate *AS.A.8 Project Practices Compliance Argument*. 2. Based on the *AS.A.8 Project Practices Compliance Argument*, for each standard demand adapt the structure as indicated in the *AS.A.9 Project Compliance Pattern*. The evidence nodes should be prepared but left empty at this stage.

Table 15.
AS.P.6 Assert conformance

AS.P.6	**Assert conformance**
Description	Monitoring conformance using *AS.A.10 Project Compliance Argument*
Steps	1. Collect the artefacts from the project development as they are produced. 2. Fill the *AS.A.10 Project Compliance Argument* with required *AS.A.12 Evidence* as it is being collected. 3. Update the *AS.A.10 Project Compliance Argument* if required along the process.

Table 16.
AS.P.7 Apply practices

AS.P.8	**Add to the Knowledge Base**
Description	Adding an artefact to the *AS.A.2 Practices Knowledge Base*
Steps	1. Enter the information into the *AS.A.2 Practices Knowledge Base* structure 2. Use a reasoner to immerse the information

Table 17 AS.P.9 Choose practices

AS.P.9	**Choose Practices**
Description	Deciding which of the practices from *AS.A.13 Suggested Practices Set* will be implemented in the *AS.A.11 Project*
Steps	1. Based on the method suggestions and your own experience decide upon a list of practices that will be used during the deployment of the project.

4.1.2. ***Use case: Improve AgileSafe***

A more detailed overview of the *Improve AgileSafe* use case is presented in Figure 4:

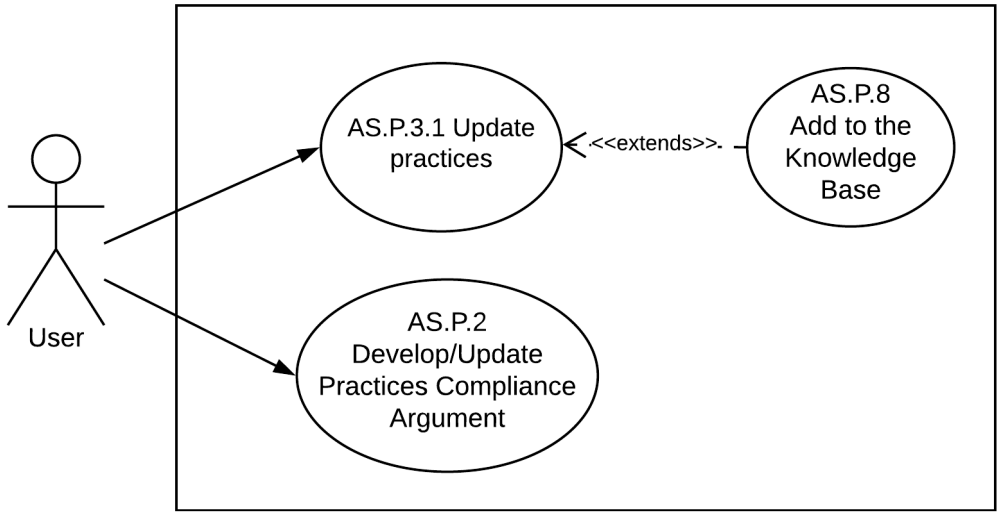


Figure 4 A detailed diagram of *Improve AgileSafe* use case

In order to improve AgileSafe, the User can update the *AS.A.2 Practices Knowledge Base* with new practices. User follows the *AS.P.3.1 Update practices* process and the new practice can be added (*AS.P.8 Add to the Knowledge Base*) to the existing resources.

A further explanation of the *Improve AgileSafe* use case is shown in the data flow diagram in the Figure 5.

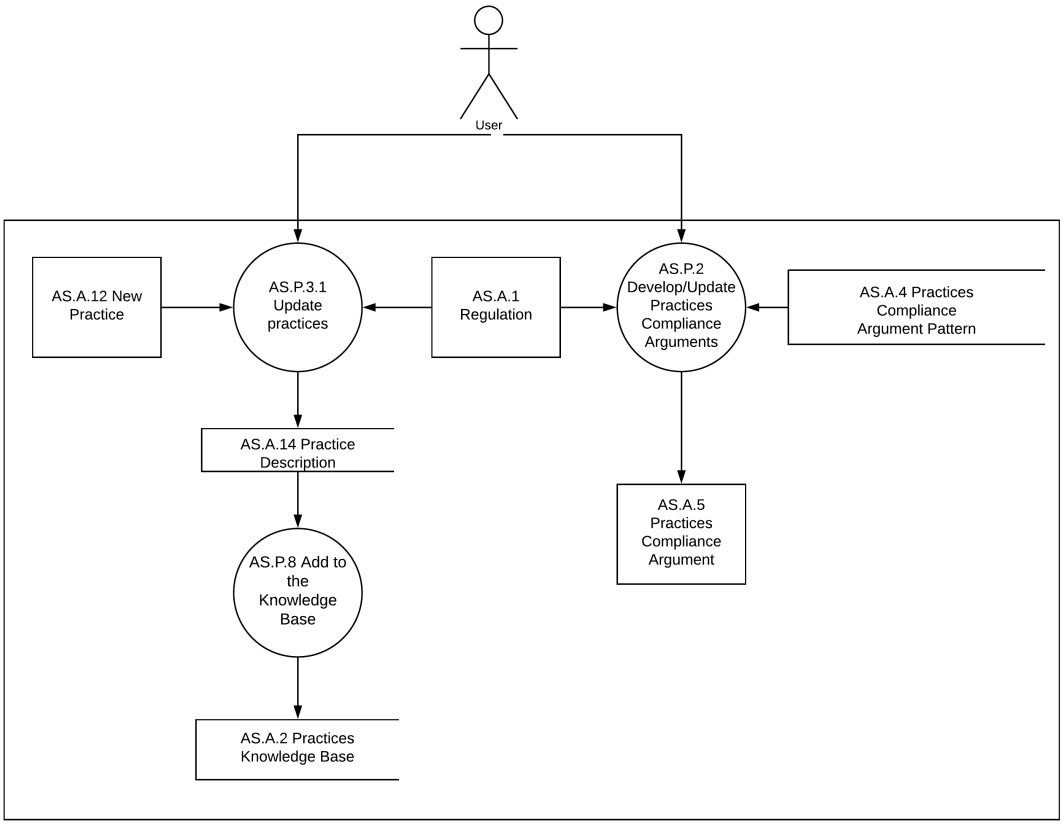


Figure 5 Data flow diagram of *Improve AgileSafe* use case

User introduces (*AS.P.3.1 Update practices*) the *AS.A.12 New Practice* or edits the information about a Practice, based on the *AS.A.1 Regulation* needs, by filling in the *AS.A.14 Practice Description*. In the next step the practice is added (*AS.P.8 Add to the Knowledge Base*) to the *AS.A.2 Practices Knowledge Base*.

User can also *AS.P.2 Develop or Update Practices Compliance Argument* for a given *AS.A.1 Regulation*, based on the *AS.A.4 Practices Compliance Argument Pattern* and using the Practices from the *AS.A.2 Practices Knowledge Base*.

The specific elements of a detailed diagram of *Improve AgileSafe* use case are described in the tables below (description of the assets already presented in section 4.1.1 are not repeated):

Table 18.
AS.A.1 Regulation

AS.A.1	**Regulation**
Type	Input
Description	An applicable standard (ISO, IEC etc.), guideline or other source of regulatory demands.

Table 19.
AS.A.12 New Practice

AS.A.12	**New Practice**
Type	Input
Description	A software development practice that might be relevant to the requirements of regulations and standards.

Table 20.
AS.A.14 Practice description

AS.A.14	**Practice description**
Type	Method framework
Description	A representation of the *AS.A.12 New Practice* in a form, in which it can be added to the *AS.A.2 Practices Knowledge Base*.

Table 21.
AS.A.4 Practices Compliance Argument Pattern

AS.A.4	**Practices Compliance Argument Pattern**
Type	Method framework
Description	A single template for developing *AS.A.5 Practices Compliance Arguments*.

Table 22.
AS.A.5 Practices Compliance Argument

AS.A.5	**Practices Compliance Argument**
Type	Output
Description	An argument that is developed based on *AS.A.4 Practices Compliance Pattern*, a given regulation and applicable practices from *AS.A.2 Practices Knowledge Base*.

Table 23.
AS.P.3.1 Update practices

AS.P.3.1	**Update practices**
Description	Editing or adding new practices to the *AS.A.2 Practices Knowledge Base*
Steps	1. Make sure that the practice you are willing to add has not yet been introduced to the *AS.A.2 Practices Knowledge Base*. If it does, edit the information you wish to change 2. Complete or edit the *AS.A.14 Practice description* with information about the practice. The more thorough the description, the better the future suggestions. 3. Add the practice in the *AS.A.2 Practices Knowledge Base*

Table 24.
AS.P.2 Develop/Update Practices Compliance Argument

AS.P.2	**Develop/Update Practices Compliance Argument**
Description	Creating or editing an *AS.A.5 Practices Compliance Argument* for a given standard.
Steps	1. [IF Develop] Prepare a structure for *AS.A.5 Practices Compliance Argument* based on a chosen standard and *AS.A.4 Practices Compliance Pattern*. 2. Decide, which practices from the *AS.A.2 Practices Knowledge Base* are able to answer the particular requirements of the structure and update the information about these practices in the *AS.A.2 Practices Knowledge Base*. 3. Arrange the Practices into the *AS.A.5 Practices Compliance Argument.*

4.2. AGILESAFE PROCESS MODEL

Figure 6 presents a complete data flow diagram of AgileSafe, including both *Apply AgileSafe* and *Improve AgileSafe* use cases:

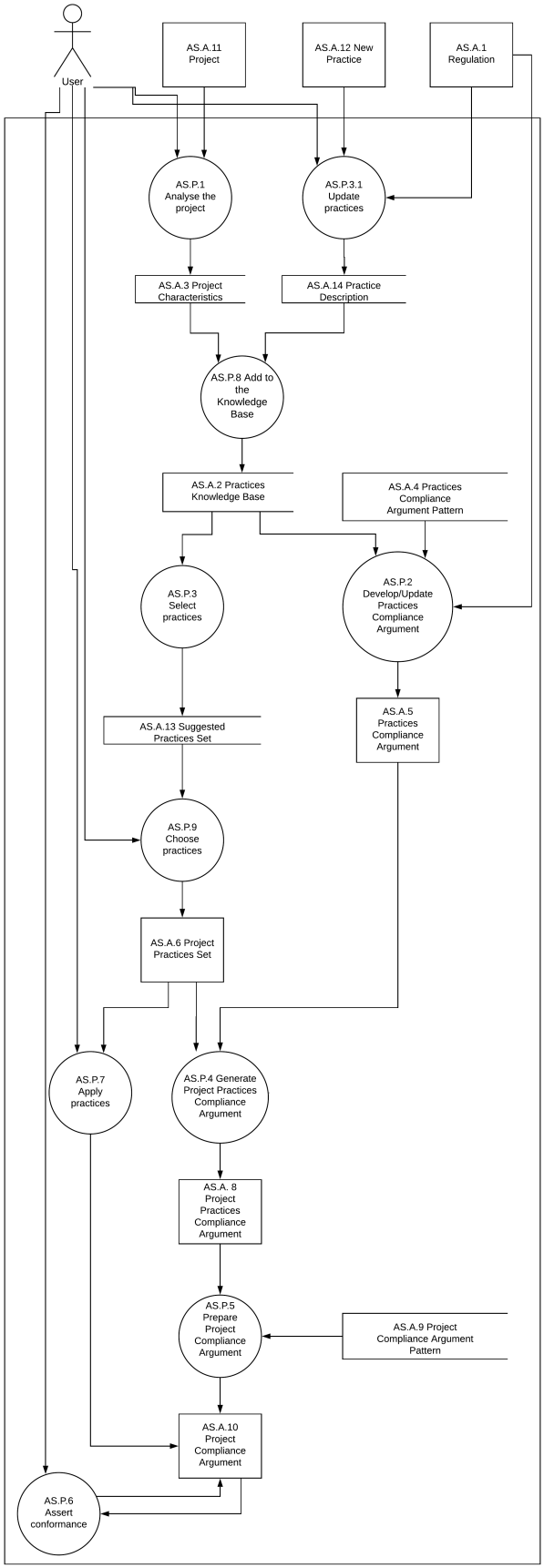


Figure 6: Data flow diagram of AgileSafe method

Application of the *AS.P.7 Apply practices* process shown in the Figure 6 results in software and other artefacts which are the evidence that can be referred to form the *AS.A.10 Project Compliance Argument*.

4.3. AGILESAFE IN SOFTWARE DEVELOPMENT PROCESS CONTEXT

AgileSafe method does not recommend any particular software development process model by default. The choice of process model might stem from the resulting *Project Practices Set* and is the responsibility of the User. Thus, AgileSafe has been designed to be adaptable to most process models, be it lightweight or more disciplined. If the User wishes to base his or her decision which process model to deploy on the *Project Practices Set*, the AgileSafe processes performed up to the obtaining this artefact (*AS.P.1 Analyse the project, AS.P.2 Develop/Update AgileSafe Practices Compliance Argument* and *AS.P.3 Select practices*) are performed without being rooted in any process model. Upon deciding which model to implement, these processes can be retroactively assigned to specific phases.

In the Table 25 below, a list of AgileSafe processes is collated with the examples of process models' phases, in which given Agile Safe process might be executed. For the demonstration purposes, a V-model and Scrum model have been chosen.

Table 25. AgileSafe processes in software development model's context

	Process model phases	
AgileSafe process	**V-model**	**Scrum**
AS.P.1 Analyse the project	System Requirements Analysis	Product Backlog Planning or with modification i.e. Sprint 0
AS.P.2 Develop/Update AgileSafe Practices Compliance Argument	System Requirements Analysis	Product Backlog Planning or with modification i.e. Sprint 0
AS.P.3 Select practices	System Requirements Analysis	Product Backlog Planning or with modification i.e. Sprint 0

AS.P.4 Generate AgileSafe Project Practices Compliance Argument	Software Requirements Analysis	Product Backlog Planning or with modification i.e. Sprint 0
AS.P.5 Prepare Project Compliance Argument	Software Requirements Analysis	Product Backlog Planning or with modification i.e. Sprint 0
AS.P.6 Assert conformance	Testing and Evaluation	Sprint, Sprint Review
AS.P.7 Apply practices	Coding and Testing	Sprint

4.4. TOOL SUPPORT

The processes of AgileSafe are currently supported by two specialist tools:

a. NOR-STA Argevide (Argevide, 2017)

It is a software solution for managing conformance with regulations using assurance arguments. It has been based on a research project called NOR-STA (NOR-STA, 2012) and Trust-IT (Górski, 2005; Górski et al., 2005; Górski, 2007) methodology, both developed by Gdańsk University of Technology.

It will be described in more detail in further chapters.

NOR-STA tool has been used for preparing AgileSafe Assurance Arguments Set patterns, as well as for developing all of the assurance arguments in the evaluation process.

It is recommended to use NOR-STA in the following AgileSafe processes: *AS.P.2 Develop/Update AgileSafe Practices Compliance Argument, AS.P.4 Generate AgileSafe Project Practices Compliance Argument, AS.P.5 Prepare Project Compliance Argument, AS.P.6 Assert conformance*.

b. Protégé (Musen, 2015)

It is a free and open-source tool for ontologies management, developed by Stanford Center for Biomedical Informatics Research.

In AgileSafe it has been used for creating and editing *AS.A.2 Practices Knowledge Base* ontology and SWRL rules.

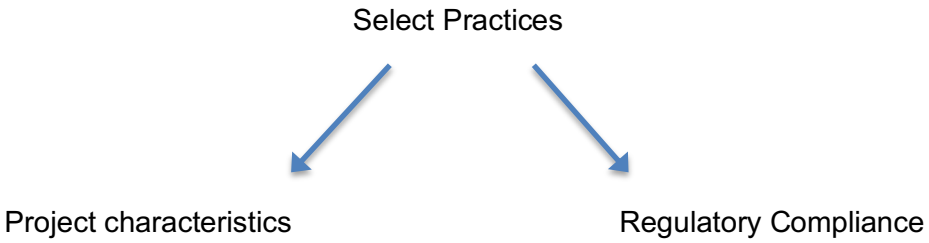
Protégé has been chosen because of its support of the newest versions of OWL and SWRL as well as for its user-friendly interface. As the AgileSafe method is not directed to the OWL professionals, editing and using the *AS.A.2 Practices*

Knowledge Base need to be as intuitive and clear as possible. Compared to other available free ontology editors, i.e. The NeOn Toolkit and Vitro, it provided a more suitable graphic interface for this project.

It is recommended to use Protégé with Pellet reasoner in the following AgileSafe processes: *AS.P.9 Choose practices, AS.P.8 Add to the Knowledge Base*.

5. PROJECT ANALYSIS AND PRACTICES SELECTION PROCESS IN AGILESAFE

The AgileSafe *AS.P.3 Select Practices* process has two main aspects it takes into consideration:



It analyses the Project in these two aspects and in the end compares the resulting recommendations and collates an approach with appropriate practices.

5.1. PROJECT CHARACTERISTICS

In order to introduce agile practices into a software development project, safety-critical or not, it is crucial to know the characteristics of the project. The first step of AgileSafe is to *AS.P.1 Analyse the project* and as a result gain a knowledge of *AS.A.3 Project Characteristics*.

These characteristics have to be relevant to the *AS.P.3 Select practices* algorithm, which suggests how agile this particular project can be. The algorithm uses the classification proposed by Scott W. Ambler for scaling agile (Ambler, 2010) which is illustrated in Figure 7. This classification focuses on context of the project. As Kruchten noted in (Kruchten, 2011), the context is vital in deciding how agile the software development and management in a given project can be. Ambler's scaling factors represent a broad spectrum of circumstances, both company and project related.

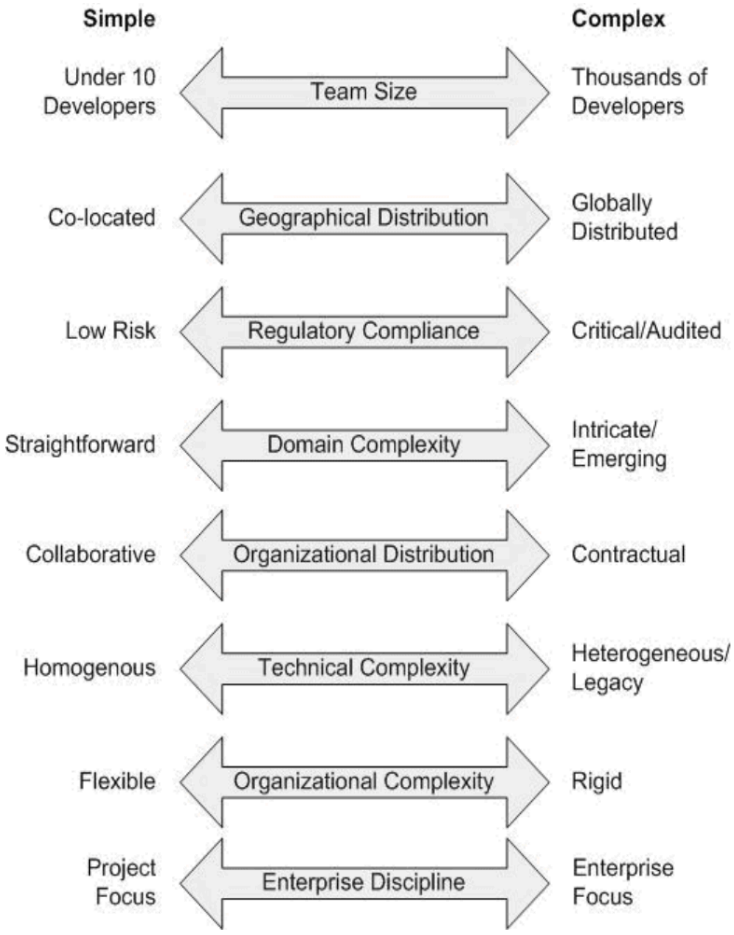


Figure 7: Potential scaling factors for software development (Ambler, 2010)

In AgileSafe the Regulatory Compliance factor of Amber's classification is omitted at this stage because the method provides a different and more sophisticated mechanism in the *AS.P.3* Select practices process to cope with this aspect. The remaining seven factors are represented in the *AS.A.2 Practices Knowledge Base* as Factors. Each Factor can be evaluated in a 5-point scale [A, B. C, D, E] meaning:

1. Team Size (based on Ambler's survey (Ambler, 2012))
   (*What is the number of developers working in the project?)*
   A – Under 10 developers; B – From 10 to 50 developers; C – From 50 to 100 developers; D – 100's of developers; E – 1000's of developers
2. Geographical Distribution (based on Ambler's survey (Ambler, 2012)) (*Where are the team members located physically?)*
   A – Co-located; B – Same building; C – Within driving distance; D – some working from home; E – Globally distributed
3. Domain Complexity
   (*How complicated is the target domain of the product?)*
   A – Straightforward; B - Predictable; C – Quickly changing; D – Complicated; E – Intricate/Emerging

4. Organisational Distribution
   (*What is the affiliation of the people working in the project, how is the work organised?*)
   A – Collaborative; B – Different teams; C – Different departments; D – Different partner companies; E – Contractual
5. Technical Complexity
   (*How complicates is the technological side of the project?*)
   A – Homogenous; B - Multiple technology; C – New technology; D - System/embedded solutions; E – Heterogeneous/Legacy
6. Organisational Complexity
   (*What are the structures of the company, how are they managed?)*
   A – Flexible, intuitive; B – Flexible, structured; C – Stable, evolutionary; D – Stable, planned; E – Rigid
7. Enterprise Discipline
   (*What lies in the centre of attention of the company management?*)
   A – Project focus; B – Mostly project focused; C – Balanced; D – Mostly enterprise focused; E – Enterprise focus

In *AS.A.2 Knowledge Base* these scales are used for both, assessing *AS.A.3 Project Characteristics* and Practices Capability within given factor. These elements will be described in more detail in further sections.

5.2. DISCIPLINES

Additionally, the *AS.A.2 Practices Knowledge Base* stores information about software development disciplines with which Practices can be predominantly connected. The disciplines are based on the OpenUP (OpenUP, 2012) disciplines because of their potential applicability to more agile practices. The disciplines are as followed:

1. Architecture
   A discipline focused on preparing a vision of software architecture.
2. Deployment
   A discipline in which most of the solution planning and deploying takes place.
3. Development
   A discipline dedicated to designing and implementing “a technical solution” (OpenUP, 2012)
4. Environment
   A discipline aimed at preparing and managing project infrastructure and processes.

5. Project Management

   A discipline focused on managing and supporting the team while working efficiently on a solution.

6. Requirements

   A discipline tackling with activities aimed at specifying and managing requirements.

7. Test

   A discipline aimed at analysing and evaluating the technical solution.

This classification is used to group the recommended practices in the *AS.A.6 Project Practices Set*. The aim is to support the user's decisions concerning the final selection from the practices proposed by AgileSafe.

5.3. GUIDANCE BASED ON PROJECT CHARACTERISTICS

The first step of AgileSafe is to *AS.P.1 Analyse the project*. A User gathers available information about the project and based on this information evaluates the project against the seven Factors and thus compose its *AS.A.3 Project Characteristics*.

Table 26 presents the form used for composing *AS.A.3 Project Characteristics*:

Table 26.Project Characteristics Analysis

Id		
Name		
Description		
Regulatory Requirements		
Characteristics	Factor	Values
	Team size	A – Under 10 developers; B – From 10 to 50 developers; C – From 50 to 100 developers; D – 100's of developers; E – 1000's of developers
	Geographical Distribution	A – Co-located; B – Same building; C – Some working from home; D – Within driving distance; E – Globally distributed
	Domain Complexity	A – Straightforward; B - Predictable; C – Quickly changing; D – Complicated; E – Intricate/Emerging

	Organisational Distribution	A – Collaborative; B – Different teams; C – Different departments; D – Different partner companies; E – Contractual
	Technical Complexity	A – Homogenous; B - Multiple technology; C – New technology; D - System/embedded solutions; E – Heterogeneous/Legacy
	Organisational Complexity	A – Flexible, intuitive; B – Flexible, structured; C – Stable, evolutionary; D – Stable, planned; E – Rigid
	Enterprise Discipline	A – Project focus; B – Mostly project focused; C – Balanced; D – Mostly enterprise focused; E – Enterprise focus;

Where:

Id – an identifier of the project

Name – a short term describing the project

Description – a characterisation of the project, its domain, purpose, client etc.

Regulatory Requirements – a list of standards, guidelines etc. with which the project need to be compliant

Characteristics – for each factor, a list of predefined circumstances, which characterises the project best (one or more, for each factor)

The information about the *AS.A.3 Project Characteristics* is then stored in the *AS.A.2 Practices Knowledge Base*.

In a similar manner information is stored about every Practice's "sweet spot" for each Factor. This information is gathered during *AS.P.3.1 Adding New Practice* process.

When a user enters the *AS.P.3 Select practices* process, the *AS.A.3 Project Characteristics* are established and the *AS.A.2 Practices Knowledge Base* is considered complete for the needs of the *AS.A.11 Project*, the guidance process can begin.

The algorithm *AS.AL.1* for the guidance based on *AS.A.3 Project characteristics,* used in *AS.P.3 Select practices,* is presented in the Figure 8, using the Z-notation (Spivey, 1992).

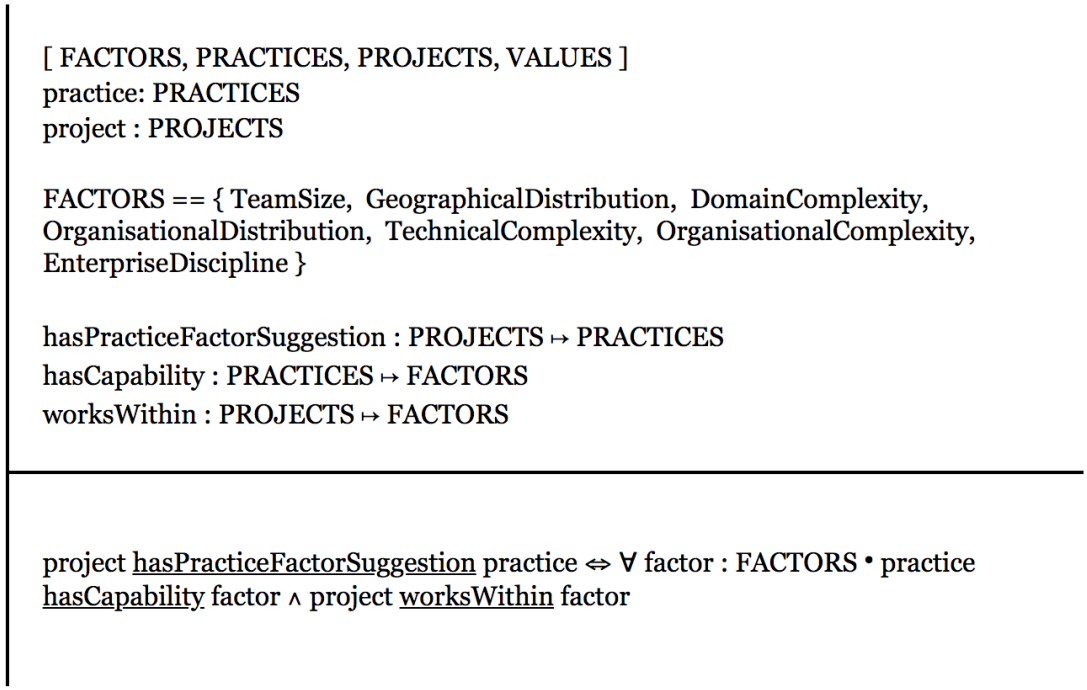


Figure 8 AS.AL.1 Algorithm for Practices suggestions based on the Project Characteristics

The *hasCapability* relation depicts at what values of a given factor a practice works best, as indicated in *AS.A.14 Practice Description*. The *worksWithin* relation depicts the *AS.A.3 Project characteristics* (values of each factor) for a given project. More information about implementation of the presented relations can be found in Section 6.

The output of the algorithm is then subject to *AS.AL.3 Algorithm for Suggested Practice Set.*

5.4. GUIDANCE BASED ON REGULATORY COMPLIANCE

Due to the importance of the regulatory aspect of the projects the AgileSafe method is addressed to, the Regulatory Compliance is considered separately to the other Project Characteristics. The *AS.AL.2* algorithm presented in the Figure 9 illustrates how the Regulatory Compliance is analysed in the *AS.P.3 Select practices* process.

[REGULATIONS, PRACTICES, PROJECTS]
practice : PRACTICES
project : PROJECTS

hasPracticeRegulationSuggestion : PROJECTS ↦ PRACTICES
fulfilsRegulation : PRACTICES ↦ REGULATION
requiresRegulation : PROJECTS ↦ REGULATION

---

project hasPracticeRegulationSuggestion project ⇔ ∀ regulation : REGULATIONS •
project requiresRegulation regulation ∧ practice fulfilsRegulation regulation

Figure 9 AS.AL.2 Algorithm for Practices suggestions based on the Regulatory Compliance

The *fulfilsRegulation* relation depicts for which regulations a given practice can provide evidence, as indicated in *AS.A.14 Practice Description*. The *requiresRegulation* relation depicts the regulations required by the project, as indicated in the *AS.A.3 Project characteristics*. More information about implementation of the presented relations can be found in Section 6.

The output of the algorithm is then subject to *AS.AL.3 Algorithm for Suggested Practice Set.*

5.5. PROJECT PRACTICES SET

An *AS.A.13 Suggested Practices Set* is a set of practices, which have a potential to respond to the Project's needs when it comes to introducing the new hybrid approach. The suggestion of the *AS.AL.3* algorithm is based on the algorithms presented in the previous paragraphs in the manner presented in the Figure 10:

$$
\begin{array}{l}
[\ PRACTICES, PROJECTS\ ] \\
practice : PRACTICES \\
project : PROJECTS \\
SuggestedProjectPracticesSet : \mathbb{P}\ PRACTICES \\
\\
hasPracticeSuggestion : PROJECTS \mapsto PRACTICES \\
hasPracticeFactorSuggestion : PROJECTS \mapsto PRACTICES \\
hasPracticeRegulationSuggestion : PROJECTS \mapsto PRACTICES \\
\hline
project\ \underline{hasPracticeSuggestion}\ practice \Leftrightarrow project\ \underline{hasPracticeFactorSuggestion}\ practice \wedge project\ \underline{hasPracticeRegulationSuggestion}\ practice \\
\\
project\ \underline{hasPracticeRegulationSuggestion}\ practice \Leftrightarrow practice \in SuggestedProjectPracticesSet
\end{array}
$$

Figure 10 AS.AL.3 Algorithm for Suggested Project Practices Set

The resulting set of Practices is then a subject to User evaluation. The User can decide which of the suggested practices she/he decides to implement in the new hybrid approach. The resulting *AS.A.6 Project Practices Set* is then used to build a customised set of AgileSafe assurance arguments.

The *AS.AL.1*, *AS.AL.2* and *AS.AL.3* algorithms presented in this chapter are implemented in the *AS.A.2 Knowledge Base* in the form of rules and will be presented in more detail in the following Section 6.

6. AGILESAFE PRACTICES KNOWLEDGE BASE

The intention behind *AS.A.2 Practices Knowledge Base* is to assemble practices for software development projects, which can provide a base for the custom-made approach. Additionally, the *AS.A.2 Practices Knowledge Base* stores information about the elements vital to the *AS.P.3 Select Practices* process such as *Project Characteristics* as well as the rules, which determine the suggested *AS.A.13 Suggested Practices Set*.

6.1. AGILESAFE PRACTICES KNOWLEDGE BASE STRUCTURE

In order to illustrate the *AS.A.2 Practices Knowledge Base* structure, in the diagram below (Figure 11) associations represent properties and classes represent concepts of the AgileSafe practices world. Class attributes in the diagram represent data properties, which should be specified for each individual of a given concept in the knowledge base.

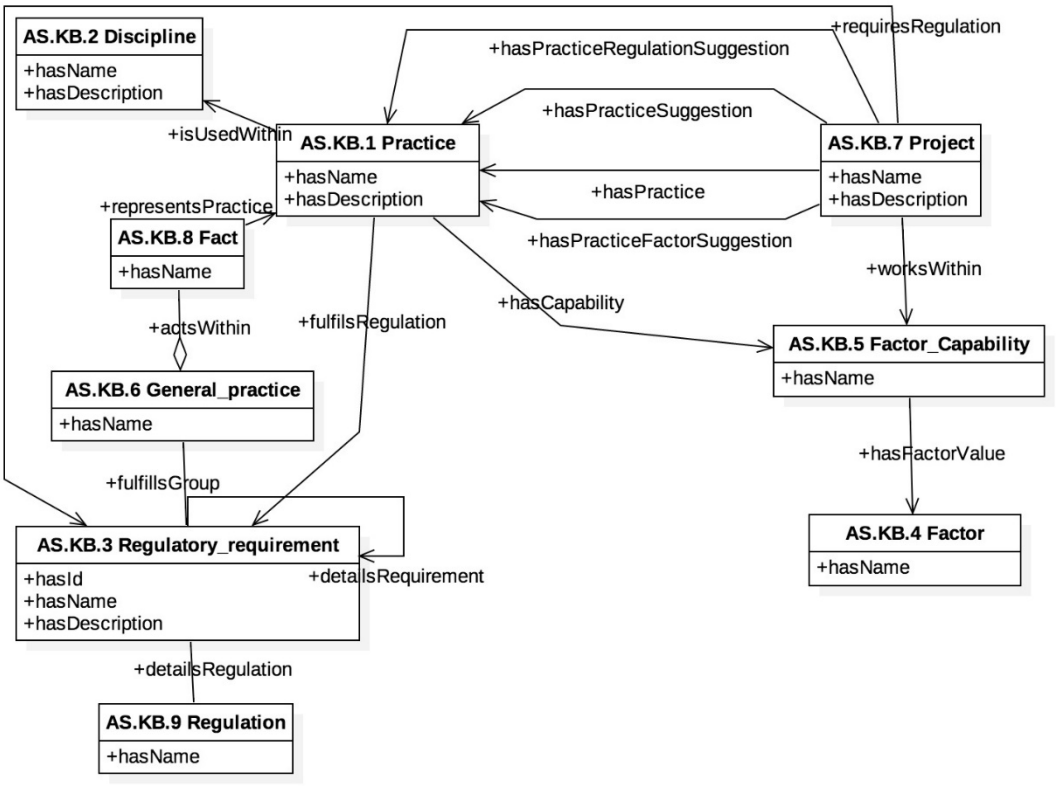


Figure 11: AgileSafe Knowledge Base class diagram

The details of the model of Figure 11 are explained in the following tables.

Data properties in the *AS.A.2 Practices Knowledge Base*:

Table 27.
hasName

	hasName
Description	Holds the name of a concept.

Table 28.
hasDescription

	hasDescription
Description	Stores a more elaborated characterisation of a concept.

Table 29.
hasId

	hasId
Description	Indicates an order number or letter (or a combination of both) of a specific concept (i.e. Regulatory_Requirement) position in the Regulation, for example: “7.1”, “9a” etc.

Concepts in the AgileSafe knowledge base:

Table 30.
AS.KB.1 Practice

AS.KB.1	**Practice**
Description	A software development practice.
Data properties	hasName; hasDescription

Table 31.
AS.KB.2 Discipline

AS.KB.2	**Discipline**
Description	Holds information about software development Disciplines. A practice can be connected with one or more of disciplines. The individuals of this concept are predefined in the Knowledge Base: 1. Architecture (*hasName* Architecture) 2. Deployment (*hasName* Deployment) 3. Development (*hasName* Development) 4. Environment (*hasName* Environment) 5. Project Management (*hasName* ProjectManagement) 6. Requirements (*hasName* Requirements) 7. Test (*hasName* Test)
Data properties	hasName

Table 32.
AS.KB.3 Regulatory Requirement

AS.KB.3	**Regulatory_Requirement**
Description	A regulatory requirement derived from a specific Regulation. It is a statement in the Regulation, which articulates some form of demand that needs to be met in order to gain conformance, for example: “4.4.a. A sequence of situations along with resulting hazard should be recorded”, where “4.4.a” should be held in the *hasId* property, “A sequence of situations along with resulting hazard should be recorded” in the *hasName* property and any additional information, if indicated in the Regulation, goes to the *hasDescription* property.
Data properties	hasName; hasId; hasDescription

Table 33.
AS.KB.4 Factor

AS.KB.4	Factor
Description	Concepts representing Factors, whose values are represented in Factor Capabilities, are used in the *AS.P.3 Select Practices* process. Each Practice has a Capability connected with each Factor. The individuals of this concept are predefined in the Knowledge Base: 1. Team Size (*hasName* TeamSize) 2. Geographical Distribution (*hasName* GeographicalDistribution) 3. Domain Complexity (*hasName* DomainComplexity) 4. Organisational Distribution (*hasName* OrganisationalDistribution) 5. Technical Complexity (*hasName* TechnicalComplexity) 6. Organisational Complexity (*hasName* OrganisationalComplexity) 7. Enterprise Discipline (*hasName* Enterprise Discipline)
Data properties	hasName

Table 34.
AS.KB.5 Factor_Capability

AS.KB.5	Factor_Capability
Description	The Practice's "sweet spot" for a given Factor. These are the values in which the Practice works best within given Factor. For example, "From 10 to 50 developers" (connected with Factor *Team Size*). This text will be stored in the *hasName* property.
Data properties	hasName

Table 35.
AS.KB.6 General Practice

AS.KB.6	**General_Practice**
Description	A Claim from the *AS.A.5 Practices Compliance Argument* concerning an approach or activity, which sums up a general goal of a group of Practices, of which given Practice is a part. Each Practice can be mentioned under many General Practices, for many standards.
Data properties	hasName

Table 36.
AS.KB.7 Project

AS.KB.7	**Project**
Description	Holds information about the *AS.A.11 Project* for which the *AS.A.13 Suggested Practices Set* is prepared.
Data properties	hasName; hasDescription

Table 37.
AS.KB.8 Fact

AS.KB.8	**Fact**
Description	An element of AgileSafe assurance arguments, which states the Practice's ability to answer the specific Regulatory Requirement demand, for example "Hazard Stories describe sequences of cause and effect for hazardous situations, in a natural language". The text of the Fact statement is written in the *hasName* property.
Data properties	hasName

Table 38.
AS.KB.9 Regulation

AS.KB.9	**Regulation**
Description	A source of requirements concerning safety or other aspect of the project or product, for example a standard, guideline, directive etc.
Data properties	hasName

The properties in the AgileSafe knowledge base:

Table 39.
AS.KB.10 fulfilsRegulation

AS.KB.10	**fulfilsRegulation**
Description	A given Practice can fulfil demands of a given Regulatory Requirement.
Usage	Practice *fulfilsRegulation* Regulatory_requirement

Table 40.
AS.KB. hasCapability

AS.KB.11	**hasCapability**
Description	A given Practice works best (hasCapability) within given values for a given Factor.
Usage	Practice *hasCapability* Factor_Capability

Table 41.
AS.KB. hasPracticeFactorSuggestion

AS.KB.12	**hasPracticeFactorSuggestion**
Description	A property connecting Project with a Practice suggested based on the Project Characteristics
Usage	Project *hasPracticeFactorSuggestion* Practice

Table 42.
AS.KB.13 hasPracticeRegulationSuggestion

AS.KB.13	**hasPracticeRegulationSuggestion**
Description	A property connecting Project with a Practice that fulfils a given Regulatory_Requirement within which the Project requiresRegulation.
Usage	Project *hasPracticeRegulationSuggestion* Practice

Table 43.
AS.KB.14 hasPracticeSuggestion

AS.KB.14	**hasPracticeSuggestion**
Description	A property connecting a Project with a Practice that is suggested to match Project needs of both Project Characteristics (hasPracticeFactorSuggestion) and Regulatory Requirements (hasPracticeRegulationSuggestion). These Practices form a *AS.A.13 Suggested Practices Set.*
Usage	Project *hasPracticeSuggestion* Practice

Table 44.
AS.KB.15 isUsedWithin

AS.KB.15	**isUsedWithin**
Description	A property connecting Practice with Disciplines, within which it is usually used.
Usage	Practice *isUsedWithin* Discipline

Table 45.
AS.KB.16 worksWithin

AS.KB.16	**worksWithin**
Description	A property connecting Project with its Project Characteristics values for each factor.
Usage	Project *worksWithin* Factor_Capability

Table 46.
AS.KB.17 requiresRegulation

AS.KB.17	**requiresRegulation**
Description	A property connecting Project with its Regulatory Requirements
Usage	Project *requiresRegulation* Regulatory_requirement

Table 47.
AS.KB.18 hasPractice

AS.KB.18	**hasPractice**
Description	A property, which connects Project with a Practice that has been chosen for *AS.A.6 Project Practices Set* for this specific project.
Usage	Project *hasPractice* Practice

Table 48.
AS.KB.19 hasFactorValue

AS.KB.19	**hasFactorValue**
Description	Each Factor can take specific values from the predefined range.
Usage	Factor_Capability *hasFactorValue* Factor

Table 49.
AS.KB.20 actsWithin

AS.KB.20	**actsWithin**
Description	In the AgileSafe assurance cases Practices are grouped into *AS.KB.6 General practices*.
Usage	Practice *actsWithin* General_practice

Table 50.
AS.KB.21 fulfilsGroup

AS.KB.21	**fulfilsGroup**
Description	Groups of Practices are placed in the appropriate nodes of the AgileSafe assurance arguments for each Regulation.
Usage	General_practice *fulfilsGroup* Regulatory_reguirement

Table 51.
AS.KB.22 detailsRequirement

AS.KB.22	**detailsRequirement**
Description	Forms the sub requirement of a specific Regulatory_requirement.
Usage	Regulatory_requirement *detailsRequirement* Regulatory_reguirement

Table 52.
AS.KB.23 detailsRegulation

AS.KB.23	**detailsRegulation**
Description	Connects the Regulatory_requirement with its source Regulation.
Usage	Regulatory_requirement *detailsRegulation* Regulation

6.2. KNOWLEDGE BASE RULES AND QUERIES

The implementation of the *AS.A.2 Practices Knowledge Base* uses a distinction between the world of OWL Concepts and the world of Semantic Web Rule Language (SWRL) Rules (SWRL, 2004). The chosen reasoner - Pellet (Pellet, 2011) supports both OWL and SWRL.

6.2.1. SWRL Rules

Due to the fact that OWL cannot express all of the AgileSafe relations, additional SWRL rules needed to be introduced for the more complex reasoning. This mainly concerns limitations in representation of property chains and relations between individuals in OWL. Following a recommended good practice in implementing ontologies, two approaches (DL and SWRL) are kept in separate base ontologies and then imported into one *AS.A.2 Practices Knowledge Base*.

The rules expressed in SWRL in *AS.A.2 Practices Knowledge Base* are presented in the screenshot from Protégé tool (Figure 12):

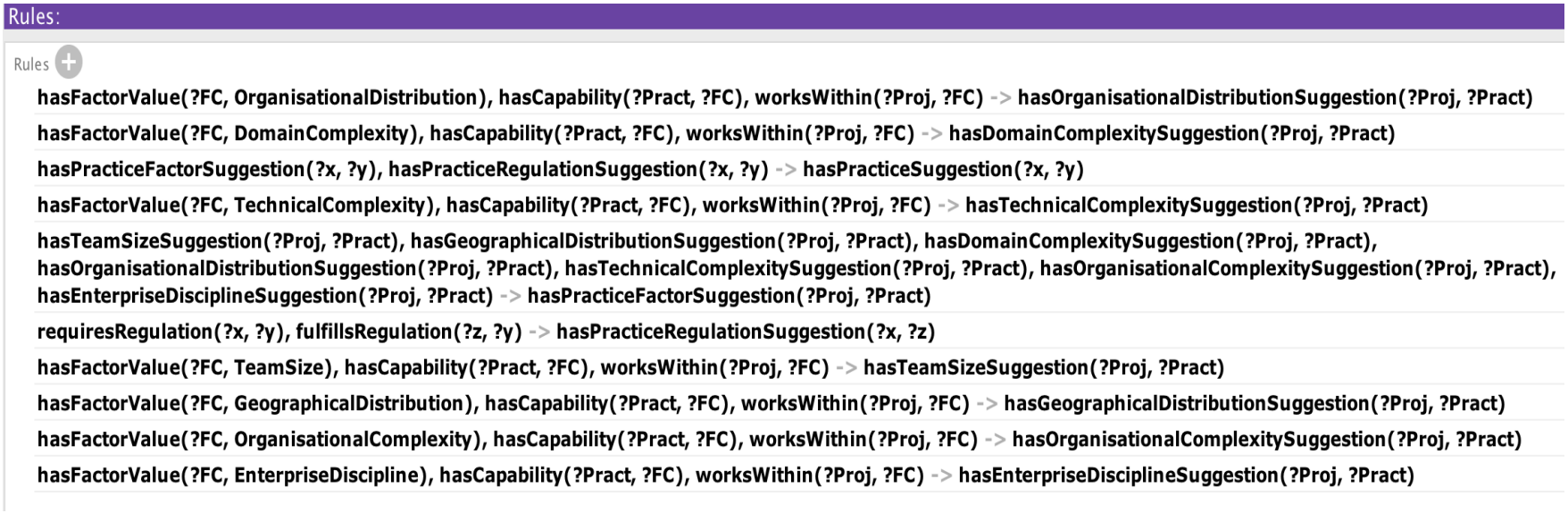


Figure 12 SWRL Rules in AgileSafe Knowledge Base (screen from Protege)

In the tables below, detailed descriptions of these rules are given:

Table 53.
AS.KB.S.1 hasTeamSizeSuggestion

AS.KB.S.1	**hasTeamSizeSuggestion**
SWRL	hasFactorValue(?FC, TeamSize) ^ hasCapability(?Pract, ?FC) ^ worksWithin(?Proj, ?FC) -> hasTeamSizeSuggestion(?Proj, ?Pract)
Variables	?FC – an individual of Factor Capability concept; ?Pract – an individual of Practice concept; ?Proj – an individual of Project concept.
Description	For a Factor TeamSize, if Practice's range of preferable values contains at least one of the values that the Project works within, the Practice is suggested for a given project within TeamSize Factor.

Table 54.
AS.KB.S.2 hasGeographicalDistributionSuggestion

AS.KB.S.2	**hasGeographicalDistributionSuggestion**
SWRL	hasFactorValue(?FC, GeographicalDistribution) ^ hasCapability(?Pract, ?FC) ^ worksWithin(?Proj, ?FC) -> hasGeographicalDistributionSuggestion(?Proj, ?Pract)
Variables	?FC – an individual of Factor Capability concept; ?Pract – an individual of Practice concept; ?Proj – an individual of Project concept.
Description	For a Factor Geographical Distribution, if Practice's range of preferable values contains at least one of the values that the Project works within, the Practice is suggested for a given project within Geographical Distribution Factor.

Table 55.
AS.KB.S.3 hasDomainComplexitySuggestion

AS.KB.S.3	**hasDomainComplexitySuggestion**
SWRL	hasFactorValue(?FC, DomainComplexity) ^ hasCapability(?Pract, ?FC) ^ worksWithin(?Proj, ?FC) -> hasDomainComplexitySuggestion(?Proj, ?Pract)
Variables	?FC – an individual of Factor Capability concept;

	?Pract – an individual of Practice concept; ?Proj – an individual of Project concept.
Description	For a Factor Domain Complexity, if Practice's range of preferable values contains at least one of the values that the Project works within, the Practice is suggested for a given project within Domain Complexity Factor.

Table 56.
AS.KB.S.4 hasOrganisationalDistributionSuggestion

AS.KB.S.4	**hasOrganisationalDistributionSuggestion**
SWRL	hasFactorValue(?FC, OrganisationalDistribution) ^ hasCapability(?Pract, ?FC) ^ worksWithin(?Proj, ?FC) -> hasOrganisationalDistributionSuggestion(?Proj, ?Pract)
Variables	?FC – an individual of Factor Capability concept; ?Pract – an individual of Practice concept; ?Proj – an individual of Project concept.
Description	For a Factor Organisational Distribution, if Practice's range of preferable values contains at least one of the values that the Project works within, the Practice is suggested for a given project within Organisational Distribution Factor.

Table 57.
AS.KB.S.4 hasTechnicalComplexitySuggestion

AS.KB.S.4	**hasTechnicalComplexitySuggestion**
SWRL	hasFactorValue(?FC, TechnicalComplexity) ^ hasCapability(?Pract, ?FC) ^ worksWithin(?Proj, ?FC) -> hasTechnicalComplexitySuggestion(?Proj, ?Pract)
Variables	?FC – an individual of Factor Capability concept; ?Pract – an individual of Practice concept; ?Proj – an individual of Project concept.
Description	For a Factor Technical Complexity, if Practice's range of preferable values contains at least one of the values that the Project works within, the Practice is suggested for a given project within Technical Complexity Factor.

Table 58.
AS.KB.S.4 hasOrganisationalComplexitySuggestion

AS.KB.S.4	**hasOrganisationalComplexitySuggestion**
SWRL	hasFactorValue(?FC, OrganisationalComplexity) ^ hasCapability(?Pract, ?FC) ^ worksWithin(?Proj, ?FC) -> hasOrganisationalComplexitySuggestion(?Proj, ?Pract)
Variables	?FC – an individual of Factor Capability concept; ?Pract – an individual of Practice concept; ?Proj – an individual of Project concept.
Description	For a Factor Organisational Complexity, if Practice's range of preferable values contains at least one of the values that the Project works within, the Practice is suggested for a given project within Organisational Complexity Factor.

Table 59.
AS.KB.S.4 hasEnterpriseDisciplineSuggestion

AS.KB.S.4	**hasEnterpriseDisciplineSuggestion**
SWRL	hasFactorValue(?FC, EnterpriseDiscipline) ^ hasCapability(?Pract, ?FC) ^ worksWithin(?Proj, ?FC) -> hasEnterpriseDisciplineSuggestion(?Proj, ?Pract)
Variables	?FC – an individual of Factor Capability concept; ?Pract – an individual of Practice concept; ?Proj – an individual of Project concept.
Description	For a Factor Enterprise Discipline, if Practice's range of preferable values contains at least one of the values that the Project works within, the Practice is suggested for a given project within Enterprise Discipline Factor.

Table 60.
AS.KB.S.4 hasPracticeFactorSuggestion

AS.KB.S.4	**hasPracticeFactorSuggestion**
SWRL	hasTeamSizeSuggestion(?Proj, ?Pract) ^ hasGeographicalDistributionSuggestion(?Proj, ?Pract) ^ hasDomainComplexitySuggestion(?Proj, ?Pract) ^ hasOrganisationalDistributionSuggestion(?Proj, ?Pract) ^ hasTechnicalComplexitySuggestion(?Proj, ?Pract) ^

	hasOrganisationalComplexitySuggestion(?Proj, ?Pract) ^ hasEnterpriseDisciplineSuggestion(?Proj, ?Pract) -> hasPracticeFactorSuggestion(?Proj, ?Pract)
Variables	?Pract – an individual of Practice concept; ?Proj – an individual of Project concept.
Description	If the Project has at least one suggestion within each Factor for a given Practice, then the Practice is suggested for this Project.

Table 61.
AS.KB.S.4 hasPracticeRegulationSuggestion

AS.KB.S.4	**hasPracticeRegulationSuggestion**
SWRL	requiresRegulation(?x, ?y) ^ fulfillsRegulation(?z, ?y) -> hasPracticeRegulationSuggestion(?x, ?z)
Variables	?x – an individual of Project concept; ?y – an individual of Regulation Requirement; ?z – an individual of Practice concept.
Description	If the Project's Regulation Requirement is met by a given Practice, then the Practice is suggested for this Project within Regulation Suggestion.

Table 62.
AS.KB.S.4 hasPracticeSuggestion

AS.KB.S.4	**hasPracticeSuggestion**
SWRL	hasPracticeFactorSuggestion(?x, ?y) ^ hasPracticeRegulationSuggestion(?x, ?y) -> hasPracticeSuggestion(?x, ?y)
Variables	?x – an individual of Project concept; ?y – an individual of Practice concept.
Description	If a given Practice is suggested for the Project both from the Project Characteristics and Regulation points, it is considered as a Practice Suggestion for the Project.

6.2.2. ***SWRL Queries***

To obtain suggestions from the *AS.A.2 Practices Knowledge Base*, the User needs to execute DL queries in the Protégé tool.

In order to obtain an *AS.A.13 Suggested Practices Set* for a specific Project, a following DL query should be executed, making sure that "Instances" checkbox is ticked - that means that instances of the Practice concept will be included in the result:

inverse (hasPracticeSuggestion) **value** <*Name of the Project instance*>

The resulting list of Practices form the *AS.A.13 Suggested Practices Set*.

6.3. ***INTRODUCING NEW PRACTICES***

There might be some company-specific practices already in place, practices that come from experience and that have already proved their value. The *AS.A.2 Practices Knowledge Base* allows users to introduce their own practices into the already existing set. In addition, software development methodologies evolve constantly and the *AS.A.2 Practices Knowledge Base* might need to be periodically updated just to stay relevant and of real value.

In order to add a new practice to the *AS.A.2 Practices Knowledge Base* a User completes an *AS.A.14 Practice Description* with information about the new practice:

Table 63. *AS.A.14 Practice Description* template

Id		
Name		
Description		
Discipline	Architecture	Yes / No
	Deployment	Yes / No
	Development	Yes / No
	Environment	Yes / No
	Project Management	Yes / No
	Requirements	Yes / No
	Test	Yes / No
Capability	Factor	Values
	Team Size	A – Under 10 developers; B – From 10 to 50 developers; C – From 50 to 100 developers; D – 100's of developers; E – 1000's of developers

	Geographical Distribution	A – Co-located; B – Same building; C – Some working from home; D – Within driving distance; E – Globally distributed	
	Domain Complexity	A – Straightforward; B - Predictable; C – Quickly changing; D – Complicated; E – Intricate/Emerging	
	Organisational Distribution	A – Collaborative; B – Different teams; C – Different departments; D – Different partner companies; E – Contractual	
	Technical Complexity	A – Homogenous; B - Multiple technology; C – New technology; D - System/embedded solutions; E – Heterogeneous/Legacy	
	Organisational Complexity	A – Flexible, intuitive; B – Flexible, structured; C – Stable, evolutionary; D – Stable, planned; E – Rigid	
	Enterprise Discipline	A – Project focus; B – Mostly project focused; C – Balanced; D – Mostly enterprise focused; E – Enterprise focus;	
Used in:	Name of the Regulation and regulatory requirement	General Practice	Fact

Where:

Id – an identifier of the practice

Name – a short term describing the practice

Description – a characterisation of the activities, artefacts etc. that form the practice

Discipline – a list of disciplines within which the practice operates (one or more)

Capability – for each factor, a list of predefined circumstances in which the practice works best (one or more, for each factor)

Used in – a link to the Regulatory Requirement that the Practice (under General Practice) complies with. The Fact is the statement of the Practice's contribution to the compliance.

An example of a completed description of a practice from *AS.A.2 Practices Knowledge Base* (excerpt):

Table 64.Hazard Stories description

Id	1	
Name	**Hazard Stories**	
Description	*Hazard stories are scenarios for possible safety violations, written in natural language much like User Stories.* *Procedure: Hazard Stories should be created at the planning stage and supplemented through the whole project development process. User Stories or Product Backlog features should be prepared before the Hazard Stories in order to outline the main objectives of the system. Methods for creating Hazard Stories can be similar to the methods known from writing User Stories; brainstorming should be useful. The team can assign roles, including "the devil's advocate". Where applicable Hazard Story should be linked to User Stories/Features it can violate or have impact on. This means that User Stories/Features ought to include non-functional requirements, especially safety-related. Based on this, priorities should be given to each Hazard Story – priorities can be taken straight from the linked User Stories/Features which means that the more important User Story/Feature it disrupts is, the more distress it can cause and should be dealt with sooner. They can provide a starting point for the safety related tests.*	
Discipline	Architecture	**No**
	Deployment	**No**
	Development	**No**
	Environment	**No**
	Project Management	**Yes**
	Requirements	**Yes**
	Test	**Yes**
Capability	Factor	Values
	Team Size	**A – Under 10 developers; B – From 10 to 50 developers;**

	Geographical Distribution	**A – Co-located; B – Same building; C – Some working from home;**	
	Domain Complexity	**C – Quickly changing; D – Complicated; E – Intricate/Emerging**	
	Organisational Distribution	**A – Collaborative; B – Different teams;**	
	Technical Complexity	**A – Homogenous; B - Multiple technology; C – New technology; D - System/embedded solutions; E – Heterogeneous/Legacy**	
	Organisational Complexity	**A – Flexible, intuitive; B – Flexible, structured; C – Stable, evolutionary;**	
	Enterprise Discipline	**A – Project focus; B – Mostly project focused; C – Balanced; D – Mostly enterprise focused; E – Enterprise focus;**	
Used in:	Name of the Regulation and regulatory requirement	General Practice	Fact
	ISO 14971 / 4.4.a. A sequence of situations along with resulting hazard should be recorded	A sequence of hazardous situations can be recorded using stories written in natural language	Hazard stories describe sequences of cause and effect for hazardous situations in a natural language
	ISO 14971 / 4.4.b Risk associated with each hazard should be estimated and recorded	Risk can be assessed in kanban style format as an annotation to each hazard description	Hazards stories can be managed in a kanban way and carry an additional annotation with information on associated risk
	IEC 62304 / 7.1.5 [IF Class B, C] Document sequences of events	A sequence of hazardous situations can be	Hazard stories describe sequences of

		recorded using stories written in natural language	cause and effect for hazardous situations in a natural language
	…		

The information collected in this form is then transferred to the *AS.A.2 Practices Knowledge Base*. In order to do that, in the present implementation the User uses the Protégé tool. As the first step, the User adds a new instance and sets its name (see Figure 13).

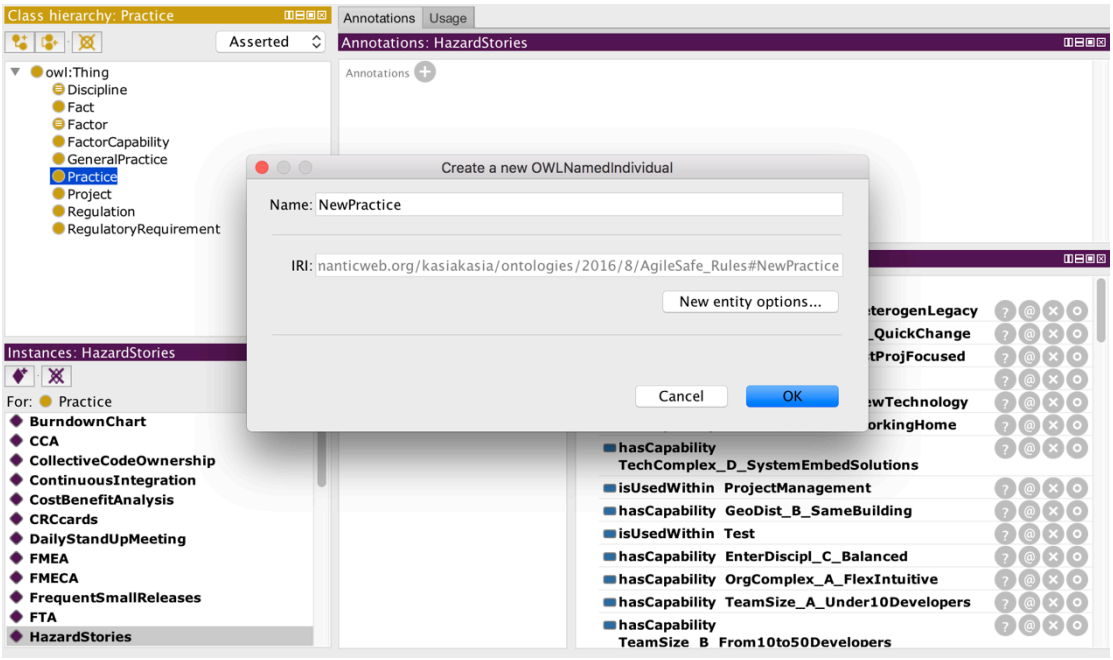


Figure 13 Adding new Practice in Protege tool - New instance

In the next step, the User specifies Factor Capabilities for the NewPractice by adding new Object property assertions in the Property assertions tab (see Figure 14).

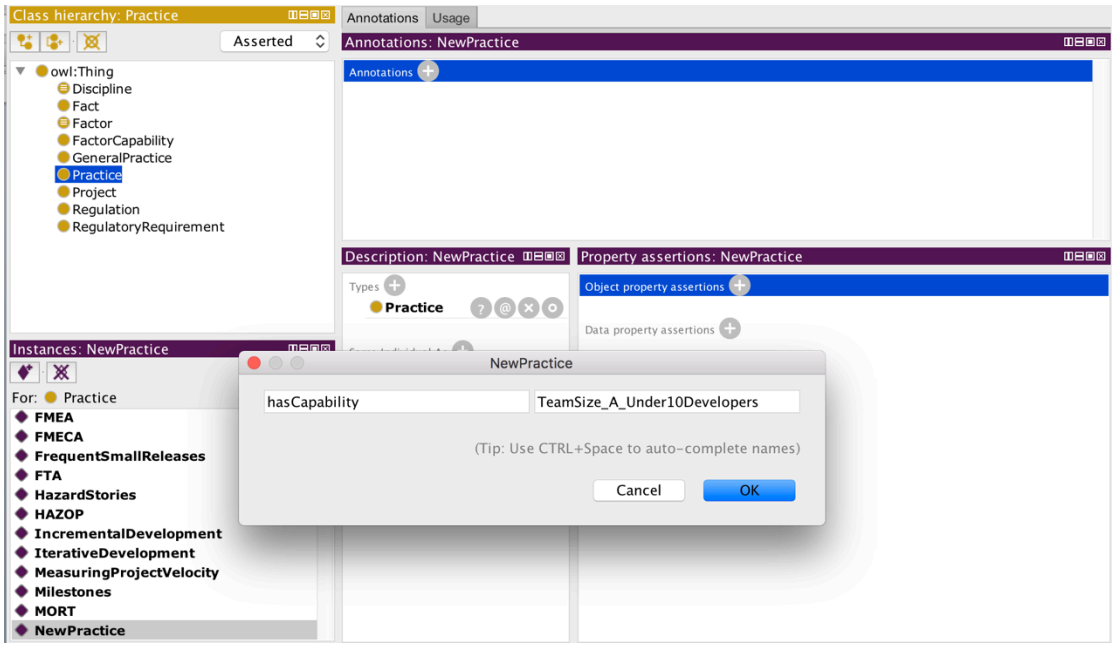


Figure 14 Adding new Practice in Protege tool - Object property assertions

On the left hand side of the new Object property assertions window the User gives a property name, hasCapability in this case, and follows with an individual name from the FactorCapability class, on the right hand side. The individuals specifying the capabilities for each Factor are already implemented in the *AS.A.2 Practices Knowledge Base* and their names reflect their connections to the Factor Capability values in the following manner: <NameOfTheFactor_OrderLetter_NameOfTheValue>.

This action needs to be repeated for all of the values for this Practice, for each Factor.

Next, the User adds Data property assertions – hasName and hasDescription (see Figure 15).

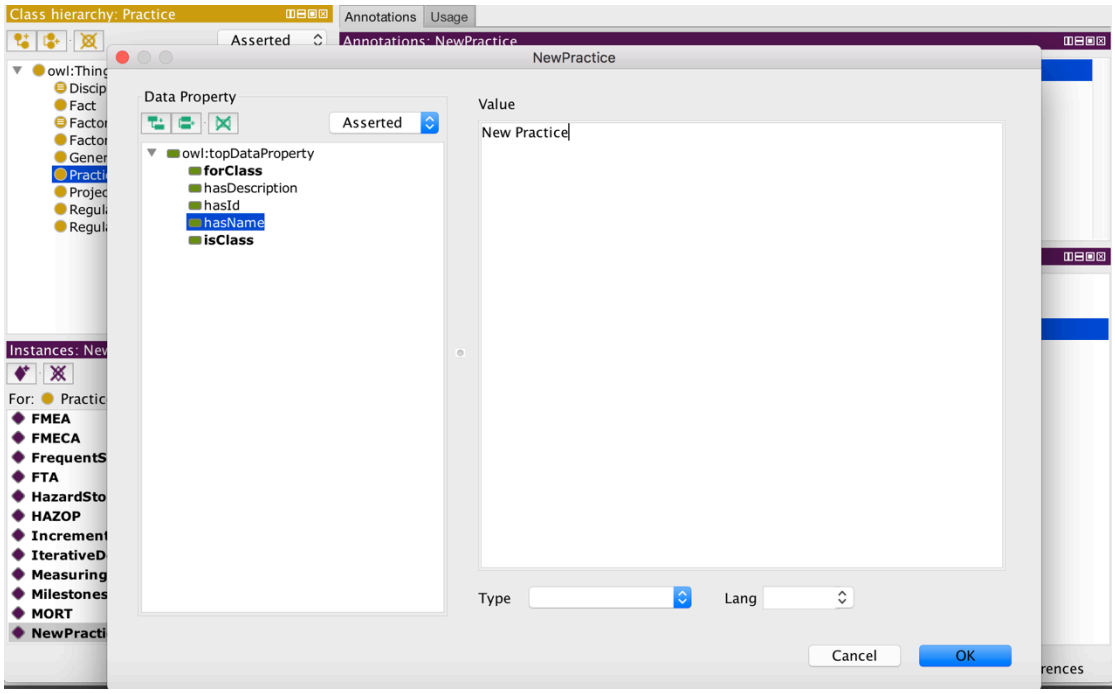


Figure 15 Adding new Practice in Protege tool - Data property assertions

In order to be fully incorporated into the *AS.AL.2* and *AS.AL.3* suggestion algorithms, the new Practice should also be connected with some Regulatory Requirements. In order to present them in the *AS.A.2 Practices* Knowledge Base, the User adds Data property assertions with property name fulfillsRegulation and values according to the Regulatory Requirements that the Practice is able to respond to (this is illustrated in Figure 16).

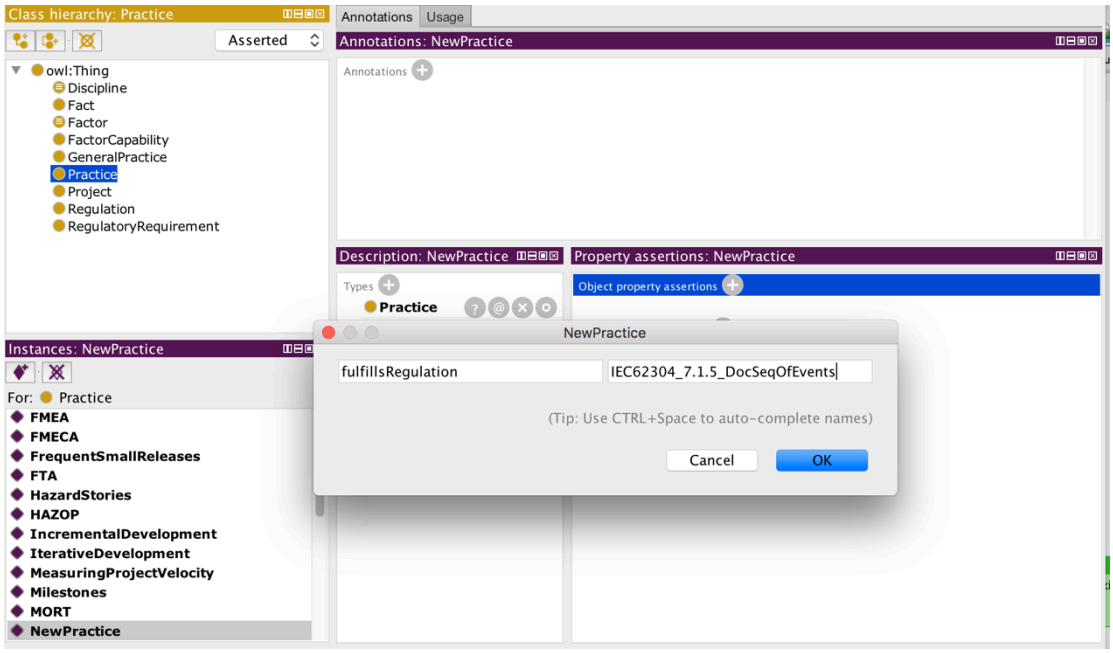


Figure 16 Adding new Practice in Protege tool - Regulatory requirements

A Practice added with such information can be then used by the *AS.AL.3* suggestion algorithm.

6.4. SELECTING PRACTICES FOR PROJECT PRACTICES SET

The result of the *AS.AL.3* algorithm is the *AS.A.13 Suggested Practices Set*. As explained in previous sections, it contains a list of Practices that might be suitable for a given Project, based on its *AS.A.3 Project Characteristics* and regulatory requirements. This list though is not ready to be implemented in the Project as it is. Depending on the Project, this list may contain several Practices that concern the same aspect of software development or produce similar artefacts, making them redundant. A User makes the final choice which Practices should be used in a given Project and thus create the *AS.A.6 Project Practices Set*. These chosen Practices are then used to build *AS.A.8 Project Practices Compliance Argument* and their artefacts fill the *AS.A.10 Project Compliance Argument*.

In order to help the User to make the best choice we propose the following recommendations:

1. Each Practice is connected with one of the specified Project development disciplines. It is recommended to check whether all of the disciplines are sufficiently covered in the resulting *Project Practices Set*.
2. The User should be careful when rejecting Practices from *AS.A.13 Suggested Practices Set*, which might result in loosing potential conformance
3. Each Practice has a Description field, which carries information about the details of its application, so the User should analyse and compare which of the Practices seem more suitable for this specific Project.

7. ASSURANCE ARGUMENTS IN AGILESAFE

7.1. ASSURANCE ARGUMENTS IN STANDARDS CONFORMANCE

When releasing a piece of safety-critical software, the company responsible for it needs to be able to prove its safety in its target environment. Proving in this context means being able to convince the licencing bodies as well as the potential users that the software is acceptably safe and will not cause harm. In order to do that a sufficient evidence to back claims about safety should be presented. This is where assurance arguments can be of great use.

7.1.1. Trust-IT

In this research assurance arguments patterns are used to guide the software developers in building explicit and incremental assurance arguments in parallel with the software development project. The patterns are derived from the relevant standards, regulations and guidelines. They follow the Trust-IT approach of applying argument structures to support application of standards (NOR-STA, 2012; Cyra, Górski, 2011a), in particular for the medical domain (Górski, Jarzębowicz, Miler, 2012).

Trust-IT (Górski, 2005; Górski et al., 2005; Górski, 2007) is an approach to promoting trust by presenting in the cyberspace 'live' arguments integrated with the supporting evidence and providing means for assessing and visualizing the compelling power of the arguments. Evidence is a document in any form: text, graphics, image, web page, video, audio etc. which is used to demonstrate the facts referred to in the argument. Integrating an argument with supporting evidence helps to make it more convincing. Trust-IT introduces a model of an argument, a graphical language for expressing arguments and a technique for integrating arguments with the evidence. It also offers a general-purpose argument appraisal mechanism based on Dempster-Shafer belief functions (Sentez K., Ferson S., 2002) among others and the corresponding mechanism of visualisation of the argument compelling power (Cyra, Górski, 2011b). Trust-IT arguments were already applied to analyse safety, privacy and security issues of personalized health and lifestyle-oriented services (Górski, Jarzębowicz, Miler, 2008), monitoring of environmental risks (ERM, 2009) and support of standards conformance (Cyra, Górski, 2011a; Górski, Jarzębowicz, Miler, 2012). Trust-IT is offered to its users by means of software services deployed in accordance with the SaaS (Software-as-a-Service) cloud-computing model. The approach is generic and can be applied in any context were evidence-based argumentation brings added value to decision making processes and disputes.

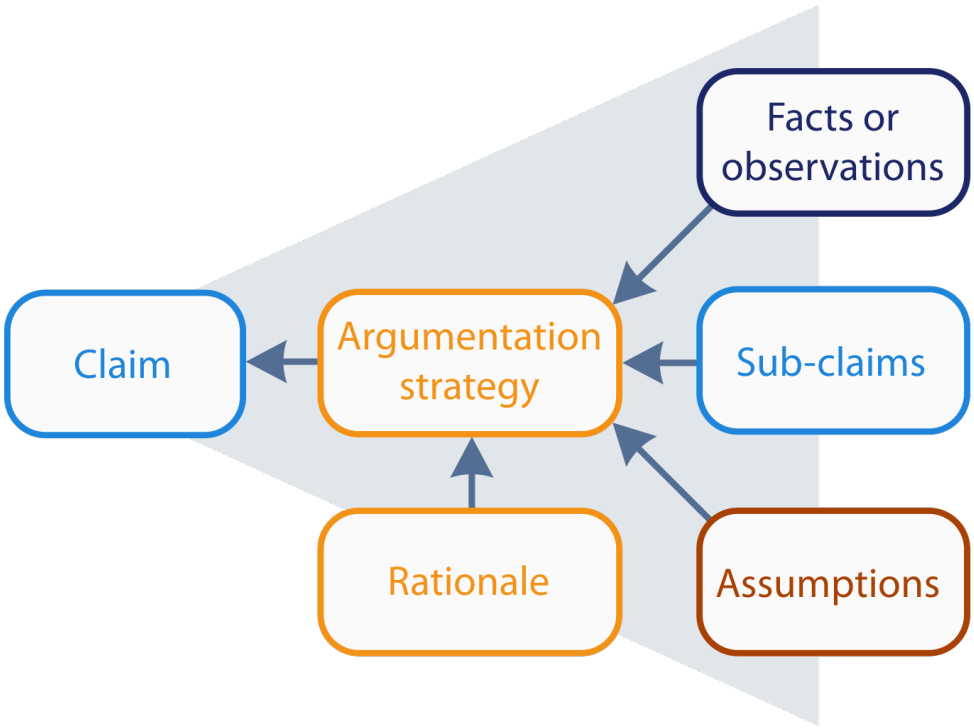


Figure 17: The structure of the arguments used in AgileSafe (Argevide, 2017)

The structure of the arguments used in AgileSafe, following the Trust-IT approach, is presented in Figure 17. The postulates about safety are expressed in the Claim nodes and if needed Sub-claims. In order to defend these postulates an Argumentation strategy is used and it can be further justified by expressing its Rationale. The Argumentation strategy is then referring to Facts, Assumptions and/or more specific sub-Claims. The Facts and Assumptions are demonstrated by evidence which is linked to them by means of references.

A generic assurance argument structure is presented in the TCL notation in Figure 18.

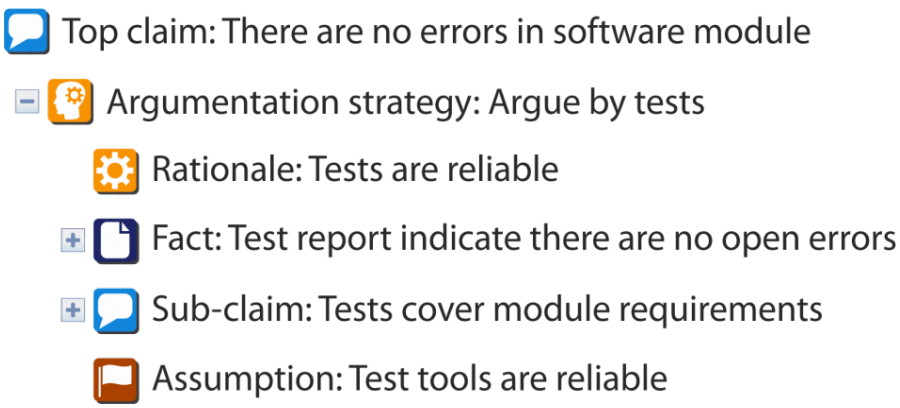


Figure 18: The assurance argument tree structure (Argevide, 2017)

7.1.2. ***Demonstrating conformance with assurance cases***

Since 2005 the idea of safety assurance cases has been analysed in depth by both FDA and Software Engineering Institute (SEI) (Weinstock and Goodenough, 2009). This partnership resulted in series of documents presenting potential uses of assurance cases in FDA certification process (Weinstock and Goodenough, 2009), (Food and Drug Administration, 2014). With FDA currently recommending the use of assurance arguments in order to present compliance with safety regulations this method is gaining more and more recognition. On the other hand, as the Health Foundation report (Bloomfield, Chozos and Cleland, 2012) states, there is little experience in the industry when it comes to the preparation of assurance cases. A need for special training and developing new methodologies in the matter is emerging. Templates and methods facilitating the use of arguments which the manufacturers can relate to can be of great value.

Another important step towards popularisation of assurance arguments was the introduction of tools allowing users to create and maintain the arguments in a user-friendly manner. The popular tools, such as Adelard SafetyCase Editor (ASCE) (Emmet and Cleland, 2002) and Astah GSN (Astah.net, 2017), are graphical editors allowing more intuitive operations on assurance cases. Other tools, such as AdvoCATE (Denney, Pai and Pohl, 2012), function as plug-ins or toolsets based on development environments i.e. Eclipse. While such tools provide substantial support for building cases using GSN notation, another tool called NOR-STA Argevide (Argevide, 2017) seems to provide a more advanced solution. It tackles several issues, not covered by the previously mentioned tools. Most interesting features from the AgileSafe point of view are support for argument reuse, sharing arguments (teamwork), practical evidence management and sophisticated argument assessment (Górski et al., 2014).

In order to increase usability of the AgileSafe method NOR-STA Argevide tool has been chosen for managing the AgileSafe arguments set.

Another important matter is the question of confidence put in the assurance case itself. If the assurance case is to be the mean to validate trust, we need to trust the assurance case as well. The question of how to assess validity of an assurance case is increasingly attracting attention of the researchers (Bloomfield, Bishop, 2010), (Weinstock, Goodenough and Klein, 2013), (Hawkins et al., 2011), (Cyra and Gorski, 2011).

7.2. OVERVIEW OF ASSURANCE ARGUMENTS IN AGILESAFE

In order to support separate certification processes, in AgileSafe the assurance arguments are developed separately for each applicable standard. The structure of these arguments is based on the standard structure.

In 2005 Kromholz and Ankrum (Kromholz Ankrum, 2005) conducted an experiment in which they tried to represent three different standards (ISO/IEC 15408:1999, RTCA/DO-178B and ISO 14971) in the form of assurance cases and analysed their applicability. They noticed that with some care, standards requirements can be represented in such form and this fact presented potential benefits, such as better organisation of the evidence, a structure for presenting conformance and overall clarity. They recognized impediments and challenges as well, including some difficulties using the Goal Structure Notation (GSN) and Adelard Safety Claims Arguments Data (ASCAD) notations chosen for the experiment as well as imperfect representation of the cases in the Adelard SafetyCase Editor (ASCE) tool they chose. In this research we attempted to reduce these obstacles by using the more advanced Trust-IT approach and Argevide services, which allow more intuitive and thorough representation.

7.2.1. Assurance arguments and agile practices

The main advantages of using assurance arguments for monitoring conformance in AgileSafe are:

A. Support for incremental certification

The concept of incremental certification has already been researched, as the need for a more flexible stance on certification process has become an increasingly common concern. It is mainly due to Agile approach's rise in popularity but also due to potential optimization of cost and time (Paige, Charalambous, Ge, Brooke, 2008), (Elmqvist et al., 2008), (Ge, Paige, McDermid, 2010). The idea is that a system might be certified modularly, which would also allow easier re-certification in the event of change in the system, in which case only the affected modules should be re-certified. Assurance arguments are vital part of such modular process and modular safety cases have been the state-of-the-art in incremental certification research (Trapp, Schneider and Liggesmeyer, 2013), i.e. they have been used by the Industrial Avionics Working Group (IAWG) in the approach enabling modular and incremental certification (Banner et. al.,2007), they have been a base for the ISO 61508 Open Certification scheme (Faller and Goble, 2007). We assume that AgileSafe assurance arguments can be adapted to modular certification in the future, if it

becomes widely acknowledged by the certifying bodies. However, it is worth noting that incremental approach to certification is not completely at odds with current guidelines.

B. Reusability of parts or whole cases

The organised structure of assurance arguments allows reusing the arguments in other projects, provided the main idea behind the structure stays the same. The scope of reuse depends on the specific cases but for the standard-based cases it can be significant. Although the workload dedicated to developing such cases can be initially high, depending on similarities between systems they can be reused up to some level, for each project needing compliance with a given standard. It can decrease in total the amount of time spent on documentation, which is an important issue while following an agile approach.

C. Intuitive and logical representation as opposed to elaborated documents

The organisation of evidence provided by assurance arguments allows greater control over what exactly is needed for compliance with a standard and whether or not the necessary evidence is produced while following chosen practices. It can reduce the redundant documentation as well as support better use of the evidence material provided during the development process. What is more, such clear argumentation standing behind each of the needed evidence can serve as good motivation and explanation for the team working on the project why they need to follow some specific practices or produce particular artefacts.

7.3. PRACTICES COMPLIANCE ARGUMENT PATTERN AND PRACTICES COMPLIANCE ARGUMENT

AS.A.5 Practices Compliance Argument is an argument, which is developed separately for each relevant regulation or standard. Its structure is based on the requirements included in such document. To make such argument uniform, it follows the *AS.A.4 Practices Compliance Pattern*, as shown in Figure 19.

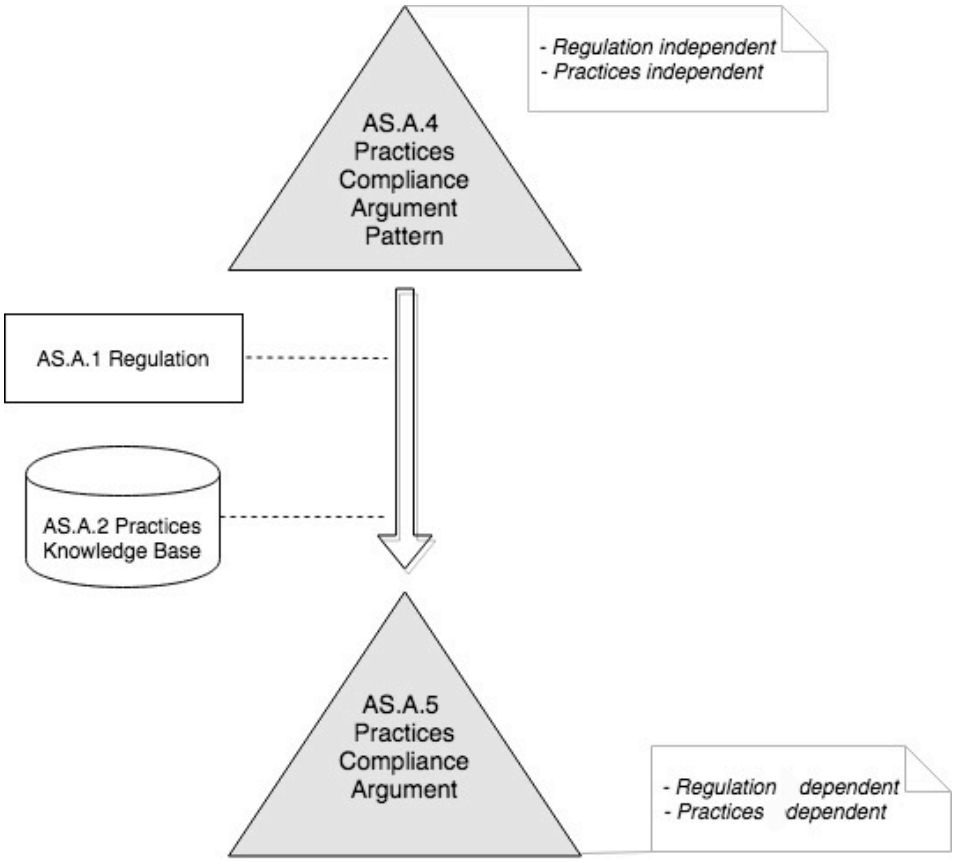


Figure 19 Practices Compliance Assurance Argument and its Pattern

The *patterns* for assurance arguments hold the information about the organization of an argument and provide a structure for creating specific arguments. They are generic and focus on the conformance of practices from *AS.A.2 Practices Knowledge Base* with particular regulatory requirements. For each requirement, it proposes an argumentation strategy and the range of software engineering practices used for collecting evidence demonstrating the compliance. It also contains explicit justification that the argumentation strategy is adequate on the condition that the evidence is collected and integrated with the argument. A list of claims concerning different types of practices, which may contribute to satisfying the requirement is presented, each claim postulating the potential of a given practice to generate the evidence needed to demonstrate compliance.

Below the *AS.A.4 Practices Compliance Argument Pattern* is presented using a GSN notation (Figure 20)

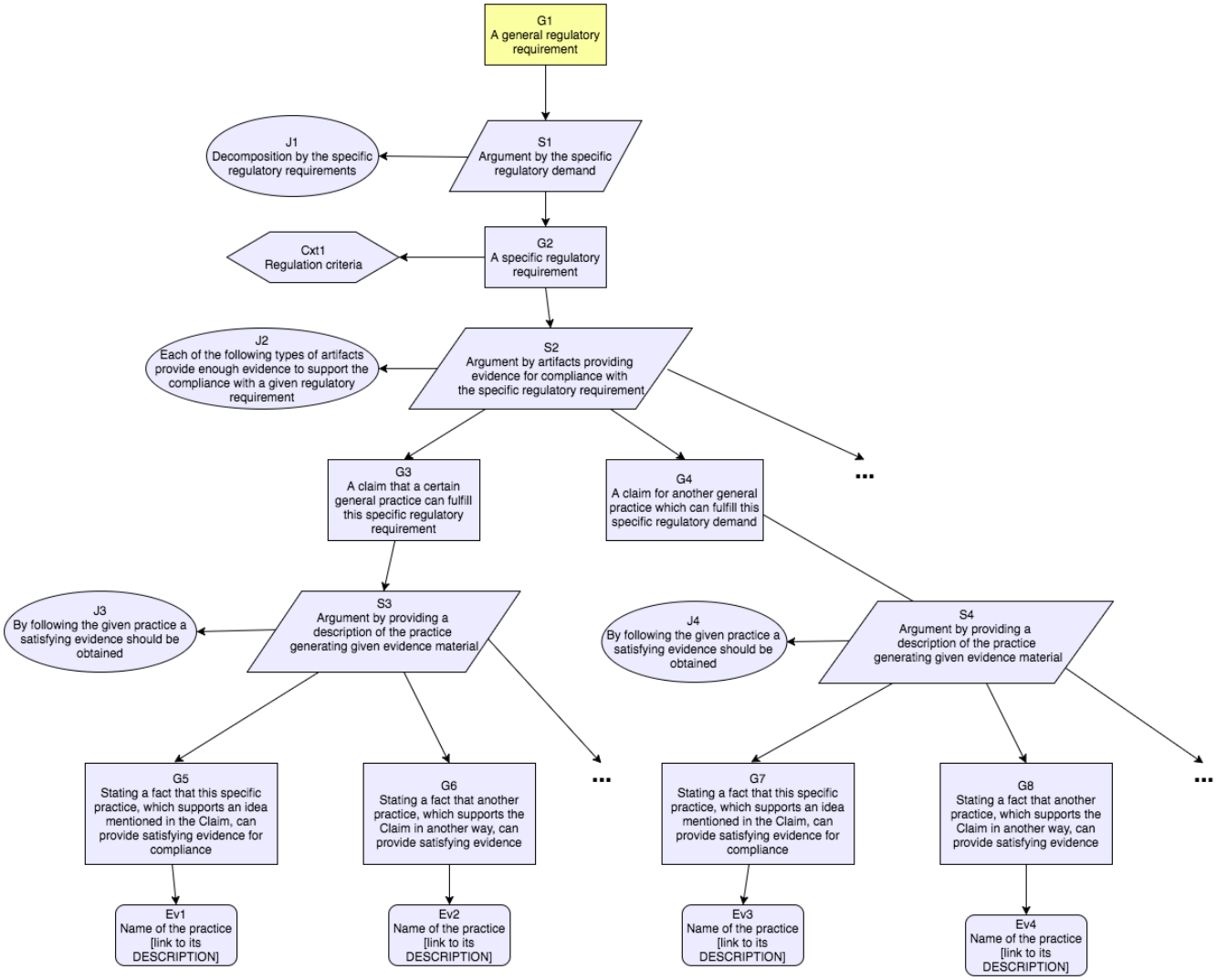


Figure 20: Presentation of Practices Compliance Argument Pattern in GSN notation

Figure 21 below presents a structure of *AS.A.4 Practices Compliance Argument Pattern* shown in Figure 21, following Trust-IT approach in NOR-STA tool:

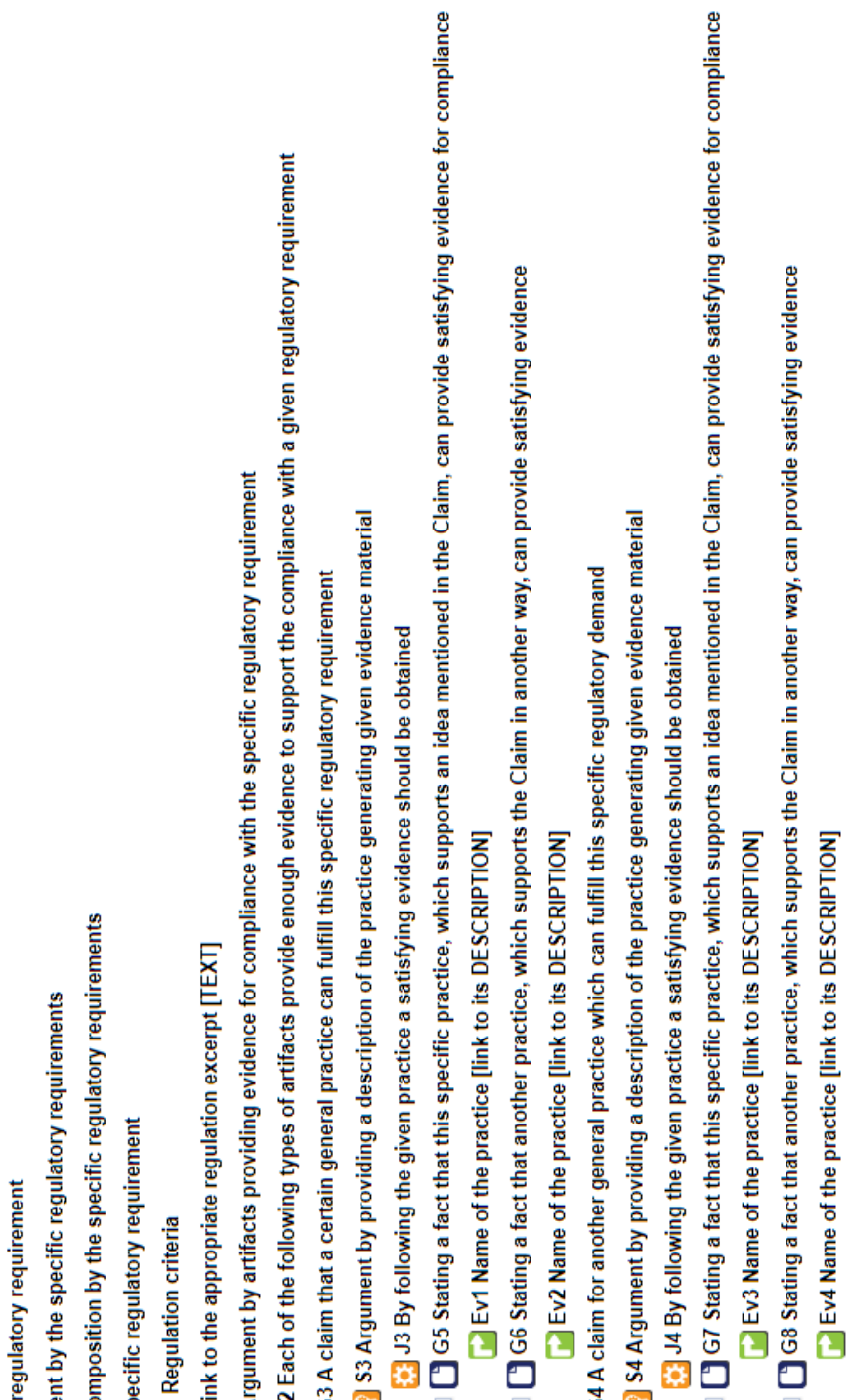


Figure 21: Practices Compliance Argument Pattern presented in NOR-STA tool

The pattern shown in Figure 21 requires that for each specific regulatory requirement (G2 in Figure 21), there is an explicit argumentation strategy: S2 *Argument by artefacts providing evidence for compliance with the specific regulatory requirement*. It is followed by the justification: J2 *Each of the following types of artefacts provide enough evidence to support the compliance with a given regulatory requirement*. The argumentation strategy S2 is supported by claims concerning different types of practices (G3 and G4 in Figure 21). Each such claim is justified by its own argumentation strategy (S3 for G3 and S4 for G4) which then refer to the specific facts (G5 and G6 for S3, and G7 and G8 for S4). These facts are demonstrated by the evidence collected directly from the descriptions of the related practices

The pattern as presented in Figure 21 can be used while building a *AS.A.5 Practices Compliance Argument* which refers to a specific standard and a specific set of practices. Figure 22 illustrate the top-level structure of the *AS.A.5 Practices Compliance Argument* for an excerpt of ISO 14971 standard (the requirements: 4*.4 Risk Analysis: Estimation of the risks for each hazardous situation).*

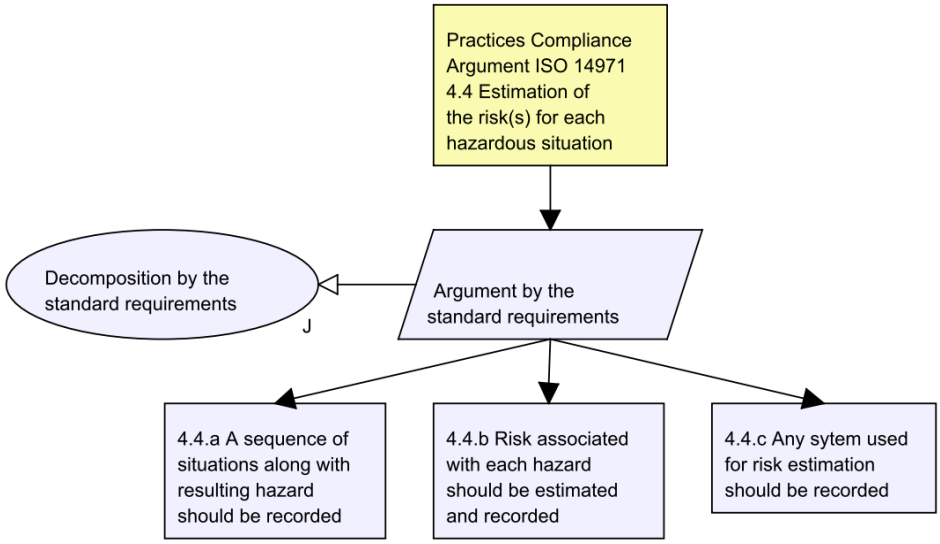


Figure 22: Fragment of Practices Compliance Argument for ISO 14971 presented using GSN notation

Figure 23 illustrates further decomposition of the ISO 14971 requirement *4.4a A sequence of situations along with resulting hazard should be recorded specifically* with reference to a set practices {Hazard stories, ZHA, FMEA}.

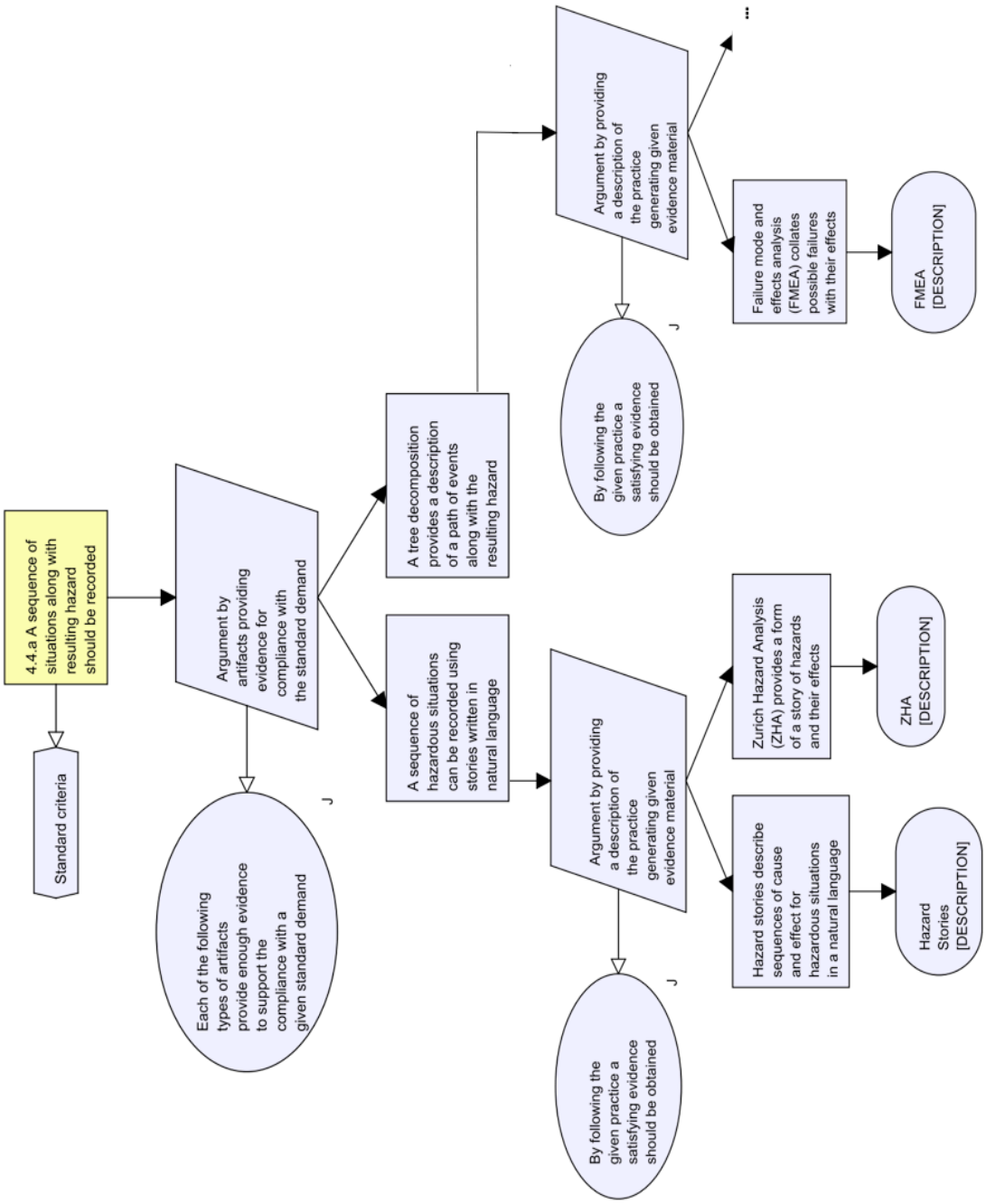


Figure 23: Fragment of Practices Compliance Argument for ISO 14971 4.4a presented in GSN notation

Figure 24 presents the argument of Figure 23 expressed with the help of NOR-STA tool.

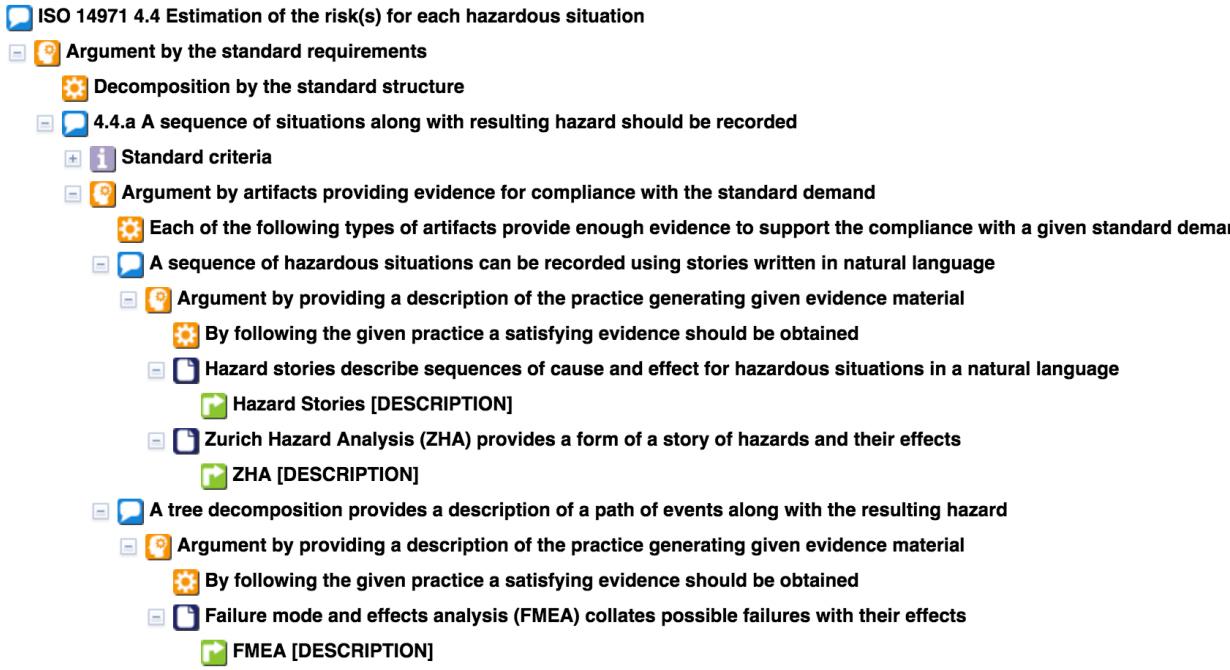


Figure 24: Fragment of Practices Compliance Argument for ISO 14971 4.4a presented in NOR-STA tool

Provided that the *AS.A.2 Practices Knowledge Base* remains unchanged, the *AS.A.5 Practices Compliance Argument* once prepared, can be reused by many software development projects which aim to be compliant with a given standard.

If the *AS.A.2 Practices Knowledge Base* gets modified (for instance, to accommodate new Practices or to include a new standard to the scope of AgileSafe) the *AS.A.5 Practices Compliance Arguments* need to be reviewed and updated accordingly. However, assuming that after some time the *AS.A.2 Practices Knowledge Base* gets 'saturated' and the frequency of such changes decreases, the *AS.A.5 Practices Compliance Arguments* become stable reusable elements supporting application of AgileSafe in many projects with the reference to many standards.

7.4. PROJECT PRACTICES COMPLIANCE ARGUMENT

AS.A.8 Project Practices Compliance Argument is an *AS.A.5 Practices Compliance Argument* adapted to a specific Project and is determined by the *AS.A.6 Project Practices Set* specific to this project. The *AS.A.8 Project Practices Compliance Argument* refers only to the practices used in the project along with the description of evidence they are providing, as presented in the Figure 25.

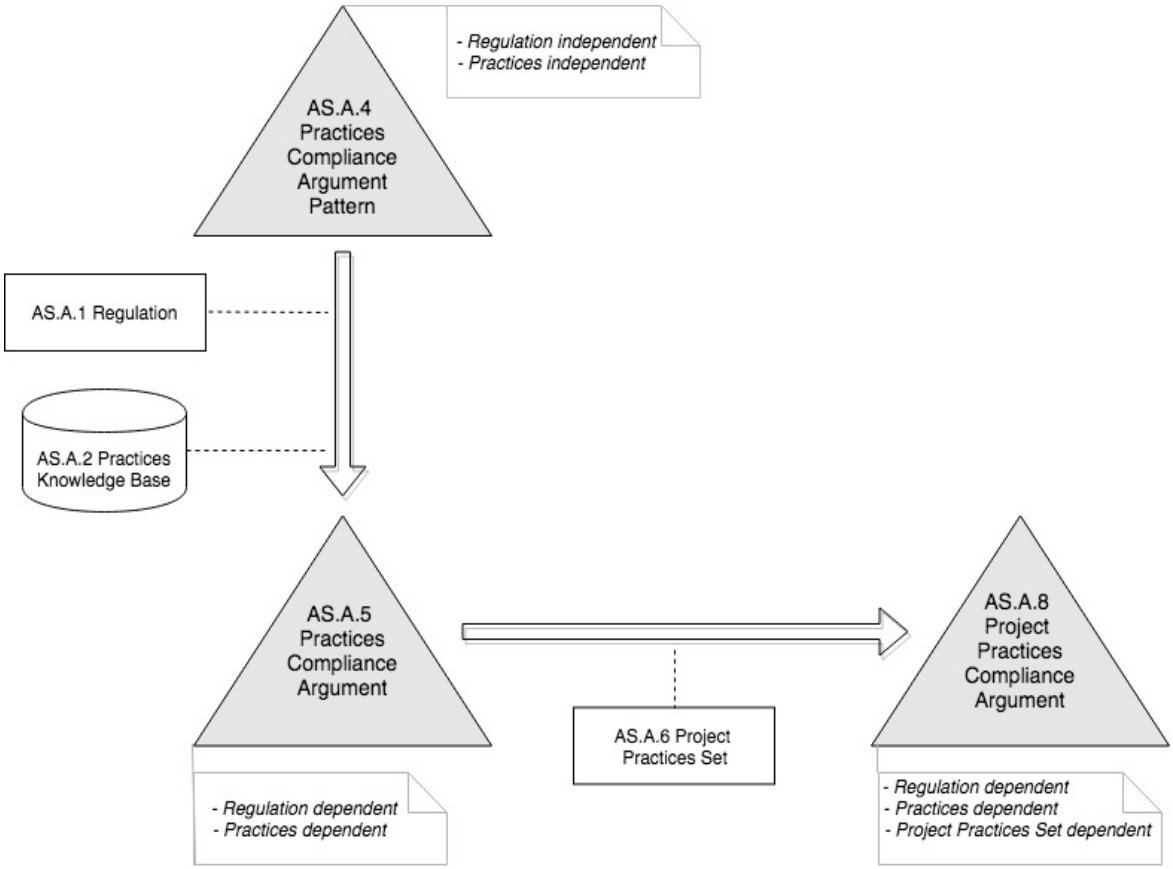


Figure 25 Project Practices Compliance Argument in the context

The structure is similar to the Practices Compliance Argument.

Below in the Figure 26 is shown an example of *AS.A.8 Project Practices Compliance Argument* for an excerpt of ISO 14971 standard– 4.4 Risk Analysis: Estimation of the risk(s) for each hazardous situation – 4.4a A sequence of situations along with resulting hazard should be recorded.

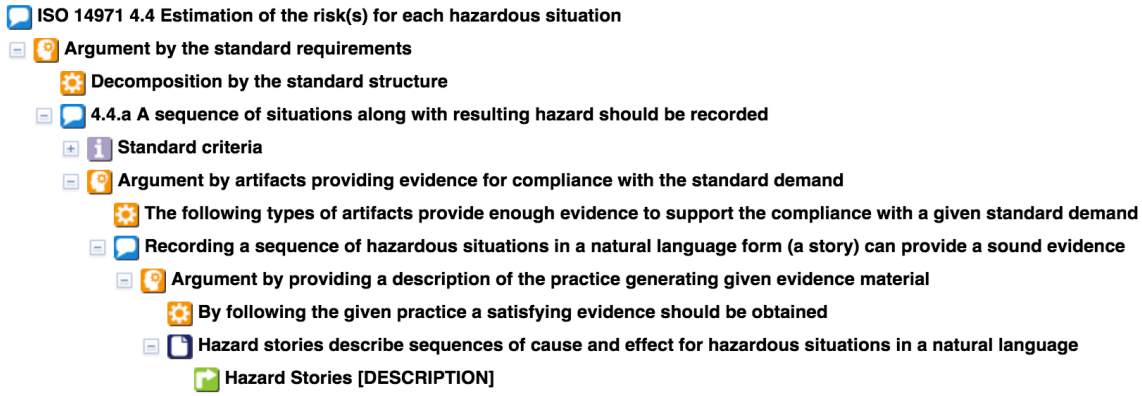


Figure 26: An excerpt of Project Practices Compliance Argument for SO 14971 standard– 4.4 Risk Analysis: Estimation of the risk(s) for each hazardous situation – 4.4a presented in NOR-STA tool

At this stage of tool support for the method, the User has to delete or hide manually the Practices from the *AS.A.5 Practices Compliance Argument* that do not apply to the Project and that are not included in its *AS.A.6 Project Practices Set* in order to create *AS.A.8 Project Practices Compliance Argument.*

It is possible that some nodes of the regulatory requirements will be left without connection to any Practice from the *AS.A.13 Suggested Practices Set*. The reason for this may be that:

a) The *AS.A.2 Practices Knowledge Base* lacks a Practice that could provide an Evidence for this particular Regulatory Requirement,
b) The Practices have not been fully analysed over their ability to respond to this particular regulatory requirement
c) The regulatory requirement is formulated in such a vague manner that it is not possible to connect it to any software development practice.

In this case, the User should analyse the regulatory requirements which are not covered by the *AS.A.13 Suggested Practices Set* and supplement the set with the needed Practices, adding them to the final *AS.A.6 Project Practices Set* and preferably to the *AS.A.2 Practices* Knowledge Base as well, for the future projects.

7.5. PROJECT COMPLIANCE ARGUMENT AND PROJECT COMPLIANCE ARGUMENT PATTERN

In AgileSafe, there are two categories of assurance arguments: (1) for development process assurance and (2) for assurance using the process artefacts, as shown in Figure 27.

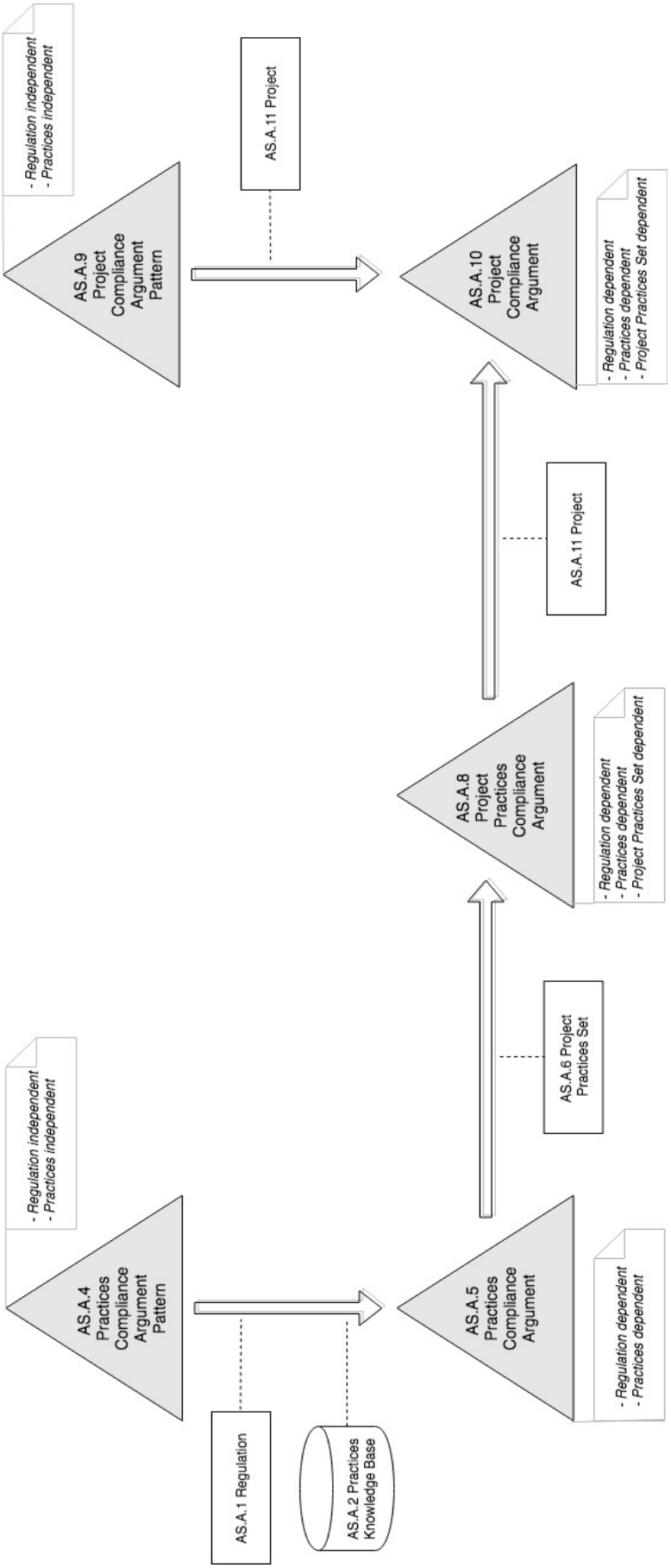


Figure 27 Project Compliance Argument in its context

While the concept of using assurance arguments for demonstrating conformity to the standards has already been discussed earlier in this chapter, introducing an additional level of process assurance argument is a more innovative concept. Both *AS.A.5 Practices Compliance Argument/AS.A.4 Pattern* and *AS.A.8 Project Practices Compliance Argument* are part of this level.

The main incentives for introducing this additional level are:

A. Traceability of the practices used in the process

While creating a hybrid approach it is important to be able to control the scope of the practices. It is even more important in the safety-critical domain, where replacing a practice with a new one can result in changes in evidence collected for certification. As such situation can cause inconsistency, an additional system of control in needed. In AgileSafe arguments, Practices are organized along with evidence they produce, according to their potential to conform to a given standard demand.

B. Supplementary conformance material

In general, Agile practices provide less documents and potential evidence than the disciplined ones. Therefore, it is important to collect and organize the available evidence as efficiently as possible. AgileSafe provides the additional assurance for the practices used in the process, as a part of the planning and management phase. For each evidence demonstrating the fulfilment of a specific standard requirement, there is a supplementary explanation justifying the choice of the evidence.

What is more, the set of AgileSafe assurance arguments provides a structure for the development process, which is often required by the certifying bodies.

C. Reusability between projects

Separating the AgileSafe assurance arguments into three arguments allows the user to reuse the higher-level arguments (mainly *AS.A.5 Practices Compliance Arguments* and *AS.A.8 Project Practices Compliance Arguments*) in other projects, which need to comply with a given standard (*AS.A.5 Practices Compliance Arguments*) or even follow the same or similar methodology (*AS.A.8 Project Practices Compliance Arguments*).

AS.A.10 Project Compliance Argument is an assurance argument where the evidence is supplied by the artefacts of the actual project development. It is structured around a particular standard and is used to collect the required product related evidence to demonstrate conformance with given standard. Each Practice has evidence that is

expected to be collected during application of this Practice. The evidence should be an effect of *AS.P.7 Apply practices* process.

The evidence is stored in the files, which are then linked with appropriate Link nodes in the argument structure.

A structure of *AS.A.9* Project Compliance Argument Pattern is presented in the Figure 28:

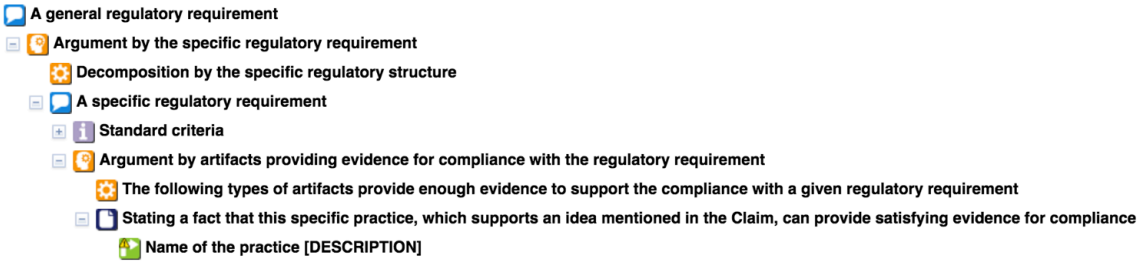


Figure 28 Project Compliance Argument Pattern presented in NOR-STA tool

Figure 29 shows an example of *AS.A.10 Project Compliance Argument* for an excerpt of ISO 14971 standard– 4.4 Risk Analysis: Estimation of the risk(s) for each hazardous situation – 4.4a A sequence of situations along with resulting hazard should be recorded.

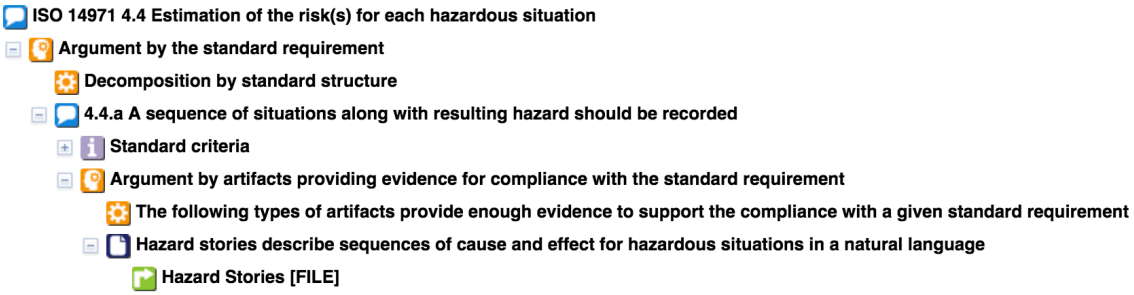


Figure 29: An example of Project Compliance Argument for an excerpt of ISO 14971 standard– 4.4 Risk Analysis: Estimation of the risk(s) for each hazardous situation – 4.4a presented in NOR-STA tool

7.6. MONITORING CONFORMANCE

In disciplined methodologies the process of monitoring conformance is rather monolithic and usually conducted by a designated team of analysts. Adapting this process to a more agile approach would mean changing this concept and incorporating different techniques that would allow the monitoring to be efficient and harmonious with a more lightweight project development.

One of the main characteristics of agile approach is incremental development, therefore the process of monitoring conformance in a hybrid method should reflect this idea as well.

The main element of monitoring conformance with chosen standards in AgileSafe is the set of assurance arguments. They were planned with the idea of incremental approach in mind.

First of all, assurance case's format allows incremental collection of evidence throughout the process. The structure chosen for AgileSafe arguments enables user to add the collected evidence as it is produced.

Secondly, the traceability of the process and of the conformance allows user to adapt the Practices and evidence. It is easier to make changes to the process, to improve procedures and quality of the work systematically when traceability of project practices along with their artefacts is provided.

The diagram below presents the processes of AgileSafe, which tackle the monitoring of conformance for each given standard (Figure 30).

Figure 30: Monitoring Conformance processes in AgileSafe

Most of these processes (*AS.P.3, AS.P.4 and AS.P.5*) are expected to be complete before the actual development stage (in Sprint 0, Planning stage or else). These are the processes aimed at preparing assurance arguments.

The core of processes, which allow conformance monitoring, is *AS.P.6 Assert Conformance*. It takes place in parallel with development activities. The User by following practices from *AS.A.6 Project Practices Set* should be able to obtain the evidence planned in *AS.A.8 Project Practices Compliance Argument* and place it in appropriate evidence nodes of the *AS.A.10 Project Compliance Argument*.

The collected evidence along with the reasoning behind it is subject to assessment of conformity with a given standard. This activity can be performed by an assessor. In AgileSafe it means that she/he analyses the *AS.A.8 Project Practices Compliance Argument* and *AS.A.10 Project Compliance Argument* for a given standard, assessing the collected evidence. She/he can use the argument appraisal mechanism as well to determine the strength of the argument, as a feedback for the team.

The assessed arguments can be used in the process of attestation. The certifying body can be presented with organised evidence for Project conformance as well as the additional evidence for the Practices used in the development process.

Once the certificate of conformance has been granted, the conformity can be maintained by continuing the work with AgileSafe Assurance Argument set and updating it as needed.

8. EVALUATION

In order to evaluate AgilSafe, the Goal-Question-Metric (GQM) method (Van Solingen, Berghout, 1999) has been followed.

8.1. EVALUATION OBJECTIVES

Based on the thesis of this doctoral research, two Goals have been declared:

(G1) Analyse whether the proposed method supports introduction of agile practices into a software development process while still maintaining the compliance with the software assurance requirements imposed by the application domain.

(G2) Analyse whether the proposed method makes it possible to reduce the effort devoted to software development processes in safety related projects.

With regard to these goals, the following research questions have been stated:

(Q1) What is the potential impact of introducing agile practice to a safety-critical software development process?

(Q2) Is AgileSafe method capable of reducing the negative impact agile practices might have on the safety-critical software development process?

(Q3) Does the method suggest practices which are relevant to a project and its environment?

(Q4) Does the method support the safety assurance aspect of a project while introducing new practices?

(Q5) Do an introduction of agile practices into a software development process carry a potential benefit of reducing effort needed for a project?

(Q6) Does AgileSafe method allow for an introduction of agile practices that can potentially reduce the effort needed for a project?

In order to find the answers for these questions, the following metrics were collected:

(M1) A list of reported benefits and problems for the introduction of agile practices - based on data from the literature concerning applying agile practices to safety-critical software development collected in Section 2 and Section 3.

(M2) Risk assessment for introduction of Scrum and eXtreme Programming practices to a safety-critical development process - based on data concerning a perceived impact agile practices might have on a safety-critical development process collected in Section 8.2

(M3) The number of additional practices needed for risk mitigation when introducing Scrum and eXtreme Programming practices - based on data concerning a perceived impact agile practices might have on a safety-critical development process collected in Section 8.2

(M4) The coverage of the Agile assurance arguments for ISO 14971 by the practices suggested in a case study - based on data concerning applicability of main elements of AgileSafe method in software development projects in a controlled limited environment collected in Section 8.3

(M5) The assessment of the New Practice template by the SafeScrum experts – based on the theoretical assessment of AgileSafe by the experts collected in Section 8.5

(M6) The list of agile practices currently used in two industry safety-critical projects – based on the theoretical assessment of AgileSafe by the experts collected in Section 8.5

(M7) The list of the most valuable aspects of AgileSafe method – based on the theoretical assessment of AgileSafe by the experts collected in Section 8.5

(M8) The list of the most troublesome aspects of AgileSafe method – based on the theoretical assessment of AgileSafe by the experts collected in Section 8.5

(M9) The relation between the number of *Suggested Practices (AS.A.13)* and the number of *Project Practices (AS.A.6)* – based on data collected in Section 8.4

(M10) The coverage of the AgileSafe assurance arguments for ISO 14971 by the GlucoMet *AS.A.6 Project Practices Set* – based on data collected in Section 8.4

The relations between specific goals, questions and measures are illustrated in Figure **31**.

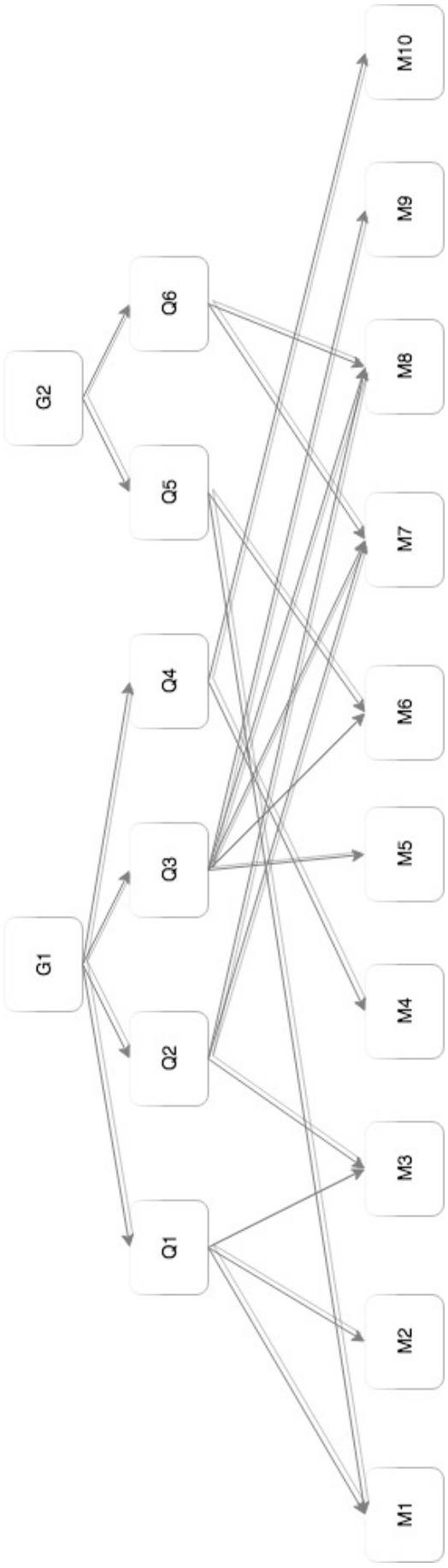


Figure 31 The GQM structure for AgileSafe evaluation

In the following paragraphs the evaluation process is described and data concerning metric M1 – M10 are presented and analysed.

8.2. CASE STUDY A: STUDENT'S ASSESSMENT

The domain of medical safety-critical software has been developing at an unprecedented speed over the past years. In addition to supplying hospitals and providing solutions for medical staff it now brings a wide variety of e-health technologies, including personal medical equipment and devices. In order to not be left behind and to reach bigger and more diverse customer groups, medical companies look for new software development processes to reduce costs, accelerate time to market and improve product quality. For this reason, we have decided to use that domain for our case study.

8.2.1. The case study objectives

The main goal of this case study was to compile a checklist of hazards as well as risk estimates for incorporating agile practices into safety-critical project development, based on risk analysis provided by the participants. Moreover, participants' suggestions for additional risk mitigation practices were investigated and reviewed as a valuable insight when preparing the model for agile practices application in safety-critical software development.

The participants were not expected to provide analysis on the expert level. They obviously based on their limited knowledge and their own working experience when completing the tasks. What was aimed for was an overview of opinions of young software engineers, their concerns regarding adapting agile practices into safety-critical software environment as well as their views on possible solutions and compromises. It was possible to determine which practices caused most uneasiness and distrust thus which practices should be revised in order to make them more compliant with safety standards. This experience helped in building better relations with prospective stakeholders.

The case study was carried out from March to the end of May 2012. It involved a group of 36 participants (students of the last semester of MSc course in software engineering). All participants had attended courses on plan-driven and agile methodologies and on high integrity systems. 67% of them had already been part-time employees in software companies and most had some prior experience with agile practices. During the case study, the participants worked in teams of 2-3.

While planning the case study the Goal-Question-Metrics (GQM) methodology was followed. The objective of the study was to investigate the participants' perception of the risks introduced by the agile practices and to collect their opinions on how these risks could be mitigated (A.G).

We have decomposed this goal into the following research questions:

(A.Q1) If and how the agile practices contribute to software hazards?

(A.Q2) Which agile practices are perceived as carrying most risk?

(A.Q3) How can the risks be mitigated?

The answers to these questions help in devising a checklist of hazards as well as risk estimates and the suggestions for additional risk mitigation practices for incorporating agile practices into safety-critical software development and will be reflected in the related software assurance argument patterns. In relation to the above questions, we selected the following metrics:

(A.M1) A complete list of agile practices associated with Scrum (Schwaber, Beedle, 2001) and eXtreme Programming (eXtreme Programming, 1999) methodologies;

(A.M2) A list of hazard scenarios explaining how the application of agile practices contributes to software hazards;

(A.M3) An assessment of risk connected with each enlisted agile practice;

(A.M4) A list of agile practices, which carry the highest risk;

(A.M5) A list of additional risk mitigation practices.

The participants were given a description of a fictional medical software company MediSoft along with a characterization of their flagship product – an insulin infusion pump. The device parameters were based on Animas One-Touch Ping pump (Animas Insulin Pumps, 2012). MediSoft wants to decide whether selected agile practices are suitable for their safety-critical projects. For this purpose, the groups were asked to assess the applicability of Scrum or alternatively eXtreme Programming to the insulin infusion pump software development project. For each group, its assignment consisted of the three tasks described in Section 8.2.2. Upon completion of the tasks we organized a workshop during which the participants were discussing their ideas, summarizing the results and concluding their work.

8.2.2. ***The case study domain description***

We devised a fictional company called MediSoft and specified its operational activity as producing software for insulin infusion pumps. After observing an increasing role of agile methodologies, the company's management team became interested in the possible benefits that might be gained by utilizing such methodologies in their workplace. As a way to investigate the effects of the introduction of agile approaches into software development, MediSoft chose to carry out a pilot project whose aim is to prepare software for an insulin infusion pump. They would like to employ eXtreme Programming and Scrum methodologies.

The participants were divided into 12 project groups, each group consisting of 3 members. Every group was given a short description of MediSoft company as well as product specification for standard infusion pump.

An insulin pump is a device for patients with diabetes who need to control their blood sugar level by administrating insulin. The pump is attached to the patient's body along with a small container filled with insulin. At the proper times, small and precisely calculated amounts of insulin are released from the container into the patient's bloodstream. It helps to keep blood glucose levels steady between meals and during sleep.

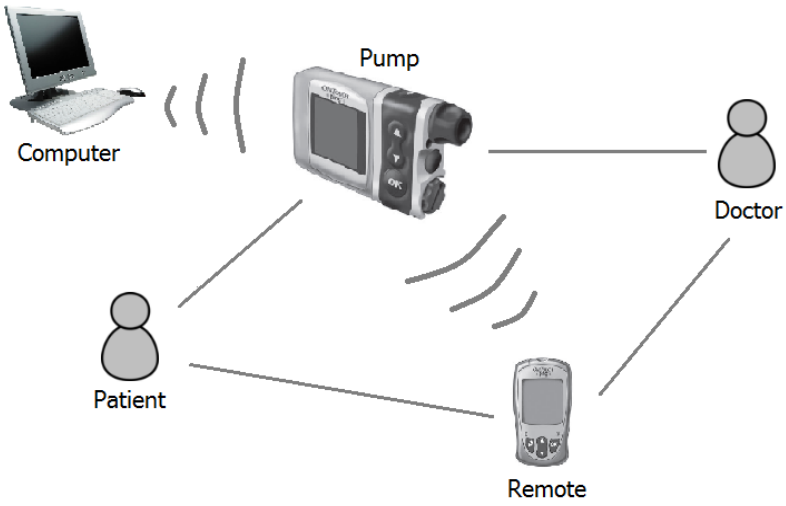


Figure 32 A context diagram of the insulin pump

The insulin pump description used in the project is based on the Animas OneTouch Ping, a real pump available on market (Animas Insulin Pumps, 2012).

8.2.3. *Participants' tasks*

The case study has been divided into three tasks. After completing each task, the participants sent in the results using Moodle course management system.

Task 1. Based on a documentation of the insulin pump and their own knowledge participants prepare a list of hazards connected with such a device. During a process of the domain analysis they are encouraged to use hazard identification techniques, like Preliminary Hazard Analysis and HAZOP adapted for this case. What is more, participants should identify and map how system hazards are related to development process hazards by building hazard scenarios documented as Fault Trees (FTA diagrams in MS Visio).

Task 2. Participants prepare a risk analysis for the introduction of agile methodologies (half of the groups for Scrum and the other half for eXtreme Programming) to a project developing the software for insulin pump. A support tool used in this task is called Designsafe (Designsafe, 2012) and it provides a platform for identifying and assessing project risks. Students are given framework project files and work on their analysis based on the content provided in these projects. Each file contains a set of project development phases and tasks according to the given methodology.

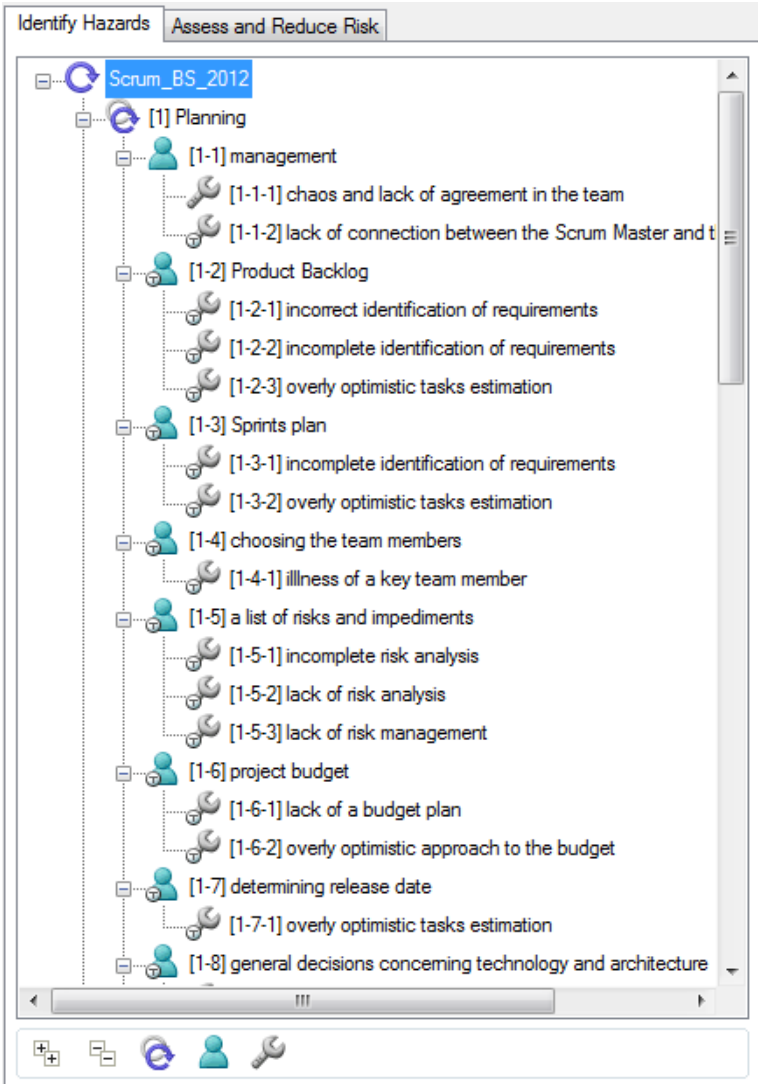


Figure 33 A tree of Scrum process stages along with tasks and possible impediments

The tree shown in Figure 33 A tree of Scrum process stages along with tasks and possible impediments consists of process stages (i.e. Planning) and tasks connected with each stage (i.e. management, Product Backlog etc.). Each task is associated with a set of typical impediments that may appear in the project during their realisation (for instance, management -> chaos and the lack of agreement in the team).

Based on the fault trees prepared in Task 1 participants analyse the connection between project development related impediments and the identified hazards. If the

trees prepared earlier by the team do not suggest the path between development process and the insulin pump hazards, they should be reworked in order to fully explain why each hazard is connected with given process impediment. The list of process impediments is very basic and should be extended where needed.

The next step is a risk analysis for the hazards and the hazard-impediment paths prepared using Designsafe tool. Severity should refer to the insulin pump hazard while Probability should refer to the whole hazard-impediment path.

An example of such a risk analysis is given below:

Table 65: An example of a simple hazard analysis (based on Designsafe layout)

Name	Insulin overdose
Description	The pump infuses a too high dosage of insulin due to insulin overflow or air pressure in lines.
Users	All
Connected Tasks	Product Backlog/Requirements Managements, Risk management, Technical analysis, Dose calculation algorithm design, Implementation, User Interface project, User's guide preparation
Cause	Faulty dose calculation algorithm, wrong requirements analysis, error in the implementation, misleading user interface or manual, database error
Severity	Catastrophic
Probability	Likely
Risk Level	High
Reduce Risk	Extensive testing, up-front technical analysis, good contact with the client representative, regular iterations, testing user interface and manual with real patients and doctors

Task 3. The participants present a list of additional practices that would mitigate the risks found in Task 2 and that could be used as an extension to the agile methodology they have been working with.

Table 66: A template for the additional practices description

NNo.	*Name of the practice*
Description	*A description of the proposed practice – what activities it includes, how should they be performed, by whom, at what stages of project.*
Related hazards	*Which hazards (from your risk analysis) the practice is expected to have influence on.*
Expected influence	*What is the expected result of implementing the practice, in what way it could reduce the risk, to what extent.*
Agility/discipline balance	*How the practice will affect the agility of the methodology ie. will it require some alterations in project roles or additional project stages etc.*

Upon completion of the tasks the participants present their ideas during a workshop, discussing their opinions on how incorporating agile practices into safety-critical projects can affect the project risk and in what manner they can be supplemented in order to be more compliant with safety standards.

8.2.4. ***The results: Hazards and hazard scenarios***

In total the participants have identified 124 hazards connected with the insulin infusion pump. Those hazards reflected different levels of detail and often represented synonymous situations. Overall, they could be grouped into 9 categories:

(1) User errors (adjusted dose, incorrect configuration, etc.).

(2) Error in measuring the level of insulin or sugar

(3) Physical / hardware errors

(4) Missing or incorrectly administered insulin doses

(5) Lack of measurement of insulin or sugar within a prescribed period

(6) Errors in alert system (sugar level, the needle slip, discharging, etc.)

(7) Unauthorized use of the device via radio waves

(8) Interruption of system normal activity

(9) Incorrect display of data.

The hazards were analysed by developing FTA diagrams and were anchored in the software development process if it was justified. The artefacts of this task were collected with respect to metric A.M2.

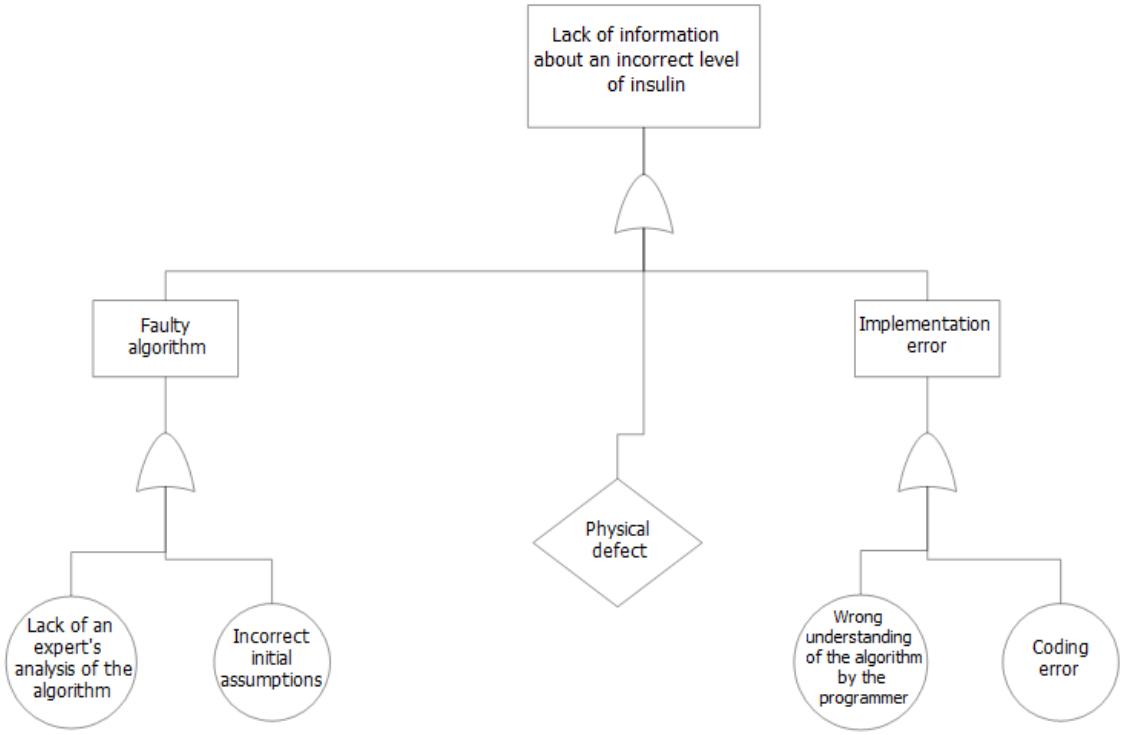


Figure 34: An example FTA analysis

8.2.5. *The results: Risk analysis and assessment*

The assessment of the hazards in relation to the agile process related impediments was performed using the Designsafe tool. The results of risk assessment were collected as metric A.M3. Figure 35 and Figure 34 present the distribution of risk levels assigned to the process impediment – hazard pairs, for Scrum and for eXtreme Programming.

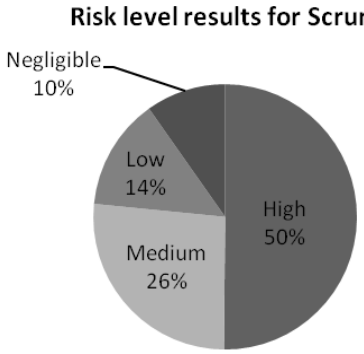


Figure 35: Risk assessment results for Scrum

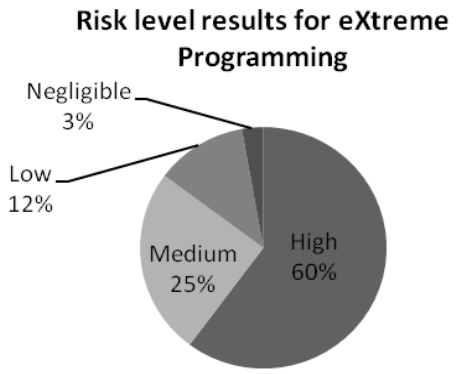


Figure 36: Risk assessment results for eXtreme Programming

Results for Scrum.

For the process impediment - hazard relationship (of type many-to-many) the groups considered 337 hazard-impediment pairs. Some of these 337 pairs were duplicated; nevertheless, they were treated separately if the groups assessed the corresponding Risk Level differently.

For the pairs associated with high risk, the following software development process stages along with their impediments were pointed most frequently (Scrum part of the A.M4 metric):

(1) Product Backlog - incorrect identification of requirements

(2) Sprints Plan - incomplete identification of requirements

(3) general decisions concerning technology and architecture - lack of architecture plan and crucial implementation decisions

(4) general decisions concerning technology and architecture - incomplete architecture plan and lacking crucial implementation decisions

(5) providing the requirements (client) - incorrect identification of requirements (6) providing the requirements (client) - incomplete identification of requirements

(7) Sprint implementation - incomplete set of tests

(8) product release - large number of detected errors.

As shown in Figure 35, 50% of the pairs were associated with high risk, 26% with medium risk, 14% with low risk while 10% were associated with negligible risk.

Results for eXtreme Programming.

Overall, the groups working with eXtreme Programming distinguished 669 hazard-impediment pairs. Out of these, 60% pairs were assessed with high risk, 25% with medium risk, 12% with low risk and 3% with negligible risk (see Figure 36)

The following process stages along with their impediments were assessed with high risk most frequently (eXtreme Programming part of the A.M4 metric):

(1) User Stories - incomplete identification of requirements

(2) Prototyping - too general plan for architecture and methods of implementing system

(3) Release scope: functionalities from previous iteration - large load on errors from the previous iteration

(4) Tests preparation - incomplete test plan

(5) Unit tests - low test coverage

(6) Acceptance tests - low test coverage

(7) Implementation of the product at the customer premise - large number of detected errors

(8) Product release - large number of detected errors.

8.2.6. ***The results: Risk mitigation practices***

This task resulted in lists of risk mitigation practices. These results were collected as metric A.M5. The most commonly proposed practices included:

(1) Introducing an expert knowledge into the project

(2) Extensive testing (i.e. enhanced acceptance tests, Test Driven Development)

(3) Introducing safety standards

(4) Improving quality assurance in relation to the artefacts different than the code (e.g. improving User Stories quality)

(5) Keeping high coding standards.

8.2.7. ***Limitations and validity***

While the results of the study brought interesting and valuable knowledge to this research, there were some limitations and possible threats to validity present.

First of all, the group selected for this study consisted of students, which have a limited experience and knowledge of the issue. However, as suggested in the paper from 2016 (Stalhane and Malm, 2016), people with very little experience in safety analysis can perform just as well as the experts during the risk identification.

The number of participants was 36, which is a limited sample when it comes to definitive conclusions. For resources limitations we had to accept this number as sufficient to get a generic overview of the young software engineers' perspective on the matter.

8.2.8. ***Conclusions***

The case study was concluded by the workshop, which took place on 24th of May 2012.

Based on the A.M4 metric, we tried to understand better why in the methodologies which flagship feature is incremental approach to the requirements, a change or a mistake in Product/Scrum Backlog or User Stories can have, according to the study, such detrimental effect on the end product. The discussion revealed that in the opinion of participants a flexible approach to change in requirements could narrow the scope and undermine rigor and discipline needed from the safety viewpoint.

One could expect to see higher risk associated with the team collaboration impediments, as interpersonal relations are crucial in agile projects. By contrast, the participants felt that agile practices, if implemented, effectively remove extreme cases of interpersonal problems.

The study revealed differences in the participants‘ perception of Scrum and eXtreme Programming, which is illustrated by the results of Task 2. While assessing Scrum, the highest risk was perceived in project management practices, whereas for eXtreme Programming the implementation and in particular coding practices were those seen as associated with the highest risk.

The participants were cautious when it comes to introducing agile methodologies into safety-critical projects. They concluded that neither Scrum nor eXtreme Programming were suitable for such projects in their strict form. The best way to meet safety-critical requirements would be to combine agile practices of different approaches (for instance, adding Test Driven Development tools or Pair Programming to Scrum) and to add risk management practices known from more disciplined approaches.

The above findings from the case study allowed an evaluation of the incentives, which led to the formulation of the PhD thesis. The results showed that although the use of selected agile methods in the safety-critical project might leave some areas of safety management insufficiently catered for, the idea of agility in such projects might be beneficial when used in conjunction with some more disciplined practices.

In the results collected as metric A.M5, using safety standards and some support from expert knowledge were pointed out as ones of the activities that might help improve the safety of agile methods.

Based on the collected data, the evaluation metric M2 was derived from A.M3 metric as well as metric M3 from A.M5 metric.

8.3. CASE STUDY B: GROUP PROJECT

In order to gain more experience with the AgileSafe at the early stages of the method, a case study has been conducted. A Goal Question Metric method has been used to assess the study.

8.3.1. Case study description

The goal (B.G) was to use AgileSafe to incorporate selected risk management practices into an agile project with safety requirements. The project was conducted by a group of students of Informatics and lasted for 2 semesters, from February 2015 till January 2016.

The project focused on health-related mobile applications. Of particular interest was how to follow the FDA guidelines being recently issued (Food and Drug Administration, 2015).

The project team cooperated with a software development studio Bright Inventions (Bright Inventions, 2017) who offered technical support. The task presented to the students was to implement an iBeacon (iBeacon, 2017) based clinic appointment & queue management system for iOS. Main features of the application were as follows:

- appointment reminders,
- queue management,
- patient arrival detection,
- detection of patient's entrance to the consulting room,
- safe handling of the patient's documentation,
- automatic update of documentation,
- navigation facilitation.

Three students with work experience in iOS were assigned for this project, which was called DocBeacon which was also the name of the resulting product. Figure 37 presents an excerpt from the Product Backlog.

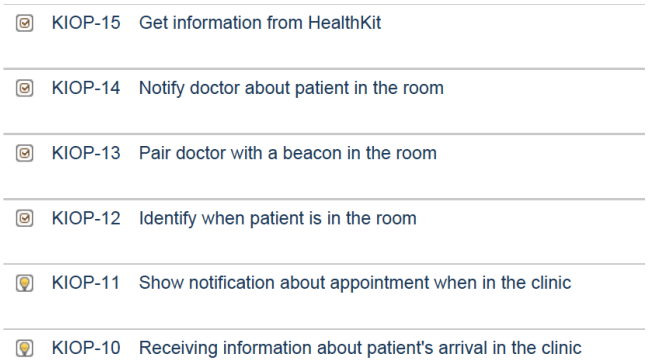


Figure 37 An excerpt from the Product Backlog for Doc Beacon (Miszczyszyn and Naliwajek, 2016)

DocBeacon was supposed to integrate with Apple Watch, the device that contains four sensors, which monitor heart rate and an accelerometer (Apple, 2015). The data stored in the application were classified as confidential. Because of the health monitoring aspect, DocBeacon could be placed somewhere between fitness-tracking/wellness-related applications and the applications used for diagnosis, treatment and prevention. As such, it might be in a "subset of mobile apps that are the focus of FDA's regulatory oversight" (Food and Drug Administration, 2015).

With this case study we hoped to answer the following questions:

(B.Q1) Can an agile project development process be complemented with risk management practices and still maintain its agility?

(B.Q2) Could this project be compliant with ISO 14971 while using the proposed set of practices?

In order to find these answers the following metrics have been collected:

(B.M1) Project Documentation

(B.M2) Opinions of the team members

(B.M3) AgileSafe assurance arguments for ISO 14971

Taking into consideration both, relatively short time for the project and the student's profile we decided not to involve them into the planning phase. It means that the students were given the *AS.A.6 Project Practices Set*, without participating in the preparation of the assurance cases. The *AS.P.6 Assert Conformance* process also was not the students' responsibility.

ISO 14971 has been selected as the reference for Regulatory Requirements and all the prepared assurance cases were based on this standard.

Table 67 DocBeacon Project Analysis

Id	1
Name	DocBeacon
Description	The product is a full-featured system supporting work of medical clinics. There are two main problems addressed and solved by this project. The first one is really earthbound: queues. Many medical clinics allow for online signup, yet clients are compelled to confirm their presence at the registration desk right before the visit. This usually means waiting in a long queue. This is particularly a problem during rush hours such as before 9 am or after 5 pm. DocBeacon solves this issue through automating the whole process by connecting beacons in the medical clinic and iOS

	devices in clients' pockets. This simple idea and seamless implementation allow for „no hassle" experience for the users. The second issue with medical clinics is the reliability of patients' medical history interview. People are often confused about what and when is exactly happening to them. They exaggerate or, even worse, ignore some symptoms, which makes doctors' work to identify health problems often a very painful experience. This product tries to fix that. DocBeacon utilizes HealthKit API which makes medical reports and interviews much more reliable and partially independent from patients' sensations and emotions. Potential users of the product: All people and all medical clinics around the world! For starters, we target only iOS users, but the iBeacon technology used allows for communication with all kinds of different devices.	
Regulatory Requirements	ISO 14971	
Characteristics	Factor	Values
	Team Size	A – Under 10 developers;
	Geographical Distribution	D – Within driving distance;
	Domain Complexity	C – Quickly changing; D – Complicated
	Organisational Distribution	A – Collaborative;
	Technical Complexity	B - Multiple technology; C – New technology; D - System/embedded solutions;
	Organisational Complexity	B – Flexible, structured;
	Enterprise Discipline	A – Project focus;

In the next step, without the students' participation, a selection of 20 risk management and software engineering practices was prepared, including the agile and agile based practices. The source for these practices were both the agile and disciplined methods guides (Schwaber, Beedle, 2001; eXtreme Programming, 1999; Boehm, Turner, 2003; BABOK, 2015) as well as two additional practices developed by the author of this thesis: Hazard Stories and Risk Backlog (Łukasiewicz, 2017). Then, based on the

AS.A.4 Practices Compliance Pattern, an *AS.A.5 Practices Compliance Argument* for ISO 14971 was prepared, using the 20 practices. Below is given an excerpt of this *AS.A.5* related to the ISO 14971 requirements: 4.4 Risk Analysis: Estimation of the risk(s) for each hazardous situation – 4.4a A sequence of situations along with resulting hazard should be recorded (Figure 38 DocBeacon Practices Compliance Argument for an excerpt of ISO 14971).

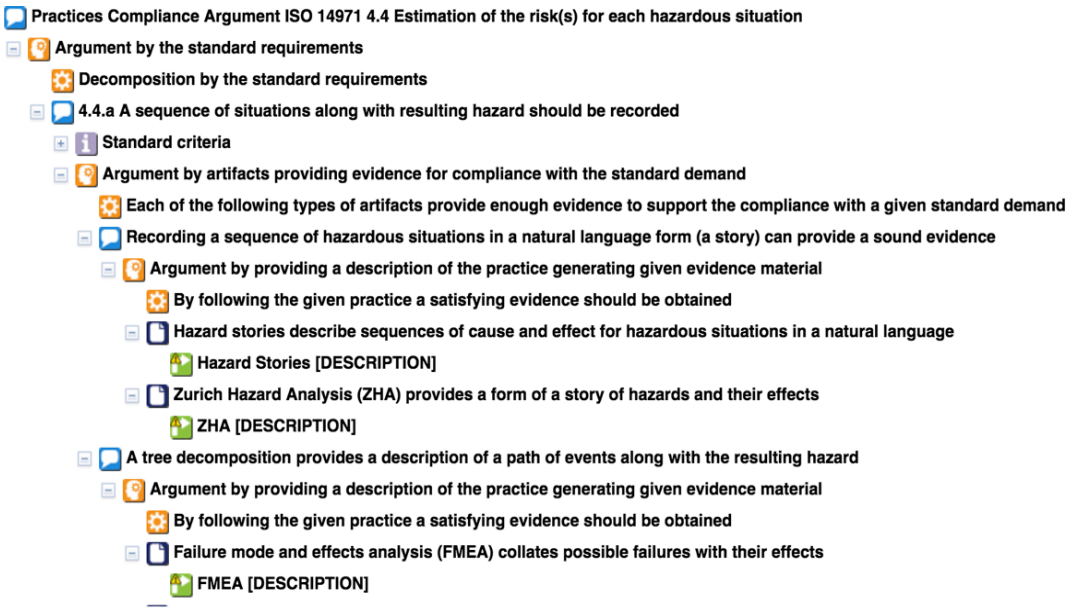


Figure 38 DocBeacon Practices Compliance Argument for an excerpt of ISO 14971

Based on the *AS.A.3 Project characteristics* and *Practices Compliance Argument* for ISO 14971 with the 20 chosen Practices, an *AS.A.6 Project Practices Set* was compiled. The Practices were selected with the four risk management phases in mind (ISO 14971, 2007):

1. Risk Analysis

In this project risk identification will be carried out through Hazard Stories/Scenarios. This term is proposed here to describe extended Risk Statements, similar to User Stories in their lifecycle. Core of such stories can be determined by using techniques like Condition-Consequence , If-Then, What/Why, 5 Whys etc. Example: “As a result of <definite cause>, <uncertain event> may occur, which would lead to <effect on objective(s)>” (Hillson, 2009)

Hazard Stories should be identified through brainstorming engaging the whole team.

2. Risk Evaluation

Hazards identified in the previous step will be analysed in the form of a Risk Backlog – a lightweight form of Risk Register. Each Hazard Story should be evaluated using a Risk Matrix.

Whole team should be engaged in the Risk Evaluation activity. Owner of the Risk Backlog is the Team along with Product Owner.

For each hazard a strategy should be chosen. A template for Risk Backlog is presented in the Figure 39:

Id.	Hazard Story	Risk Probability	Priority	Strategy	Owner	Product Backlog Task	State
1.							
2.							

Figure 39 Risk Backlog template

3. Risk Control & Residual Risk Evaluation

Risk Backlog will be maintained throughout the project development and updated incrementally. Risk Backlog should be a part of the Sprint Planning as well as Sprint Reviews discussions and should be kept up to date.

Any residual risk should be evaluated after each meeting, assessing its level.

4. Post production information

During the final review Risk Backlog should be evaluated and analysed.

One of the practices suggested for use in the project was creating Hazard Stories – the stories that would describe potential risks in an agile way, following the User Stories style, but focused on hazards.

The students were then advised on the practices they should implement in their project, based on the *AS.A.6 Project Practices Set*.

Parallel to the above activities an *AS.A.8 Project Practices Compliance Argument* for ISO 14971 was prepared. In Figure 40 an excerpt from this argument, containing Hazard Stories, is presented:

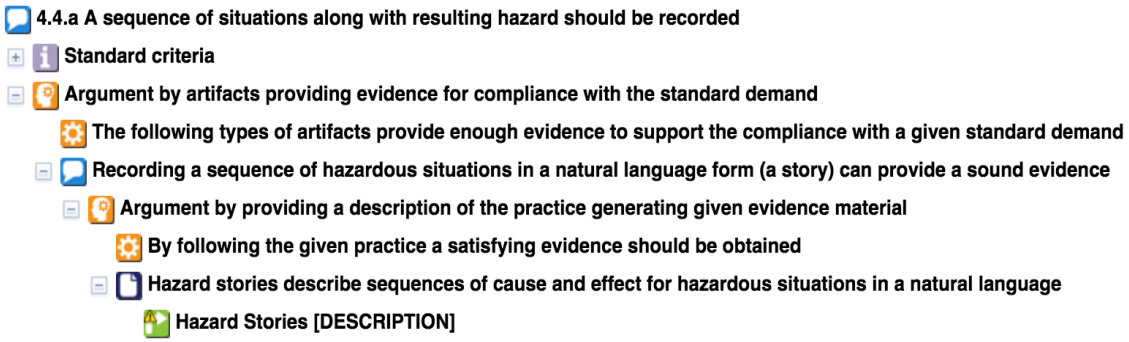


Figure 40 DocBeacon Project Practices Compliance Argument for an excerpt of ISO 14971

Subsequently, an *AS.A.10 Project Compliance Argument* for ISO 14971 was built (Figure 41).

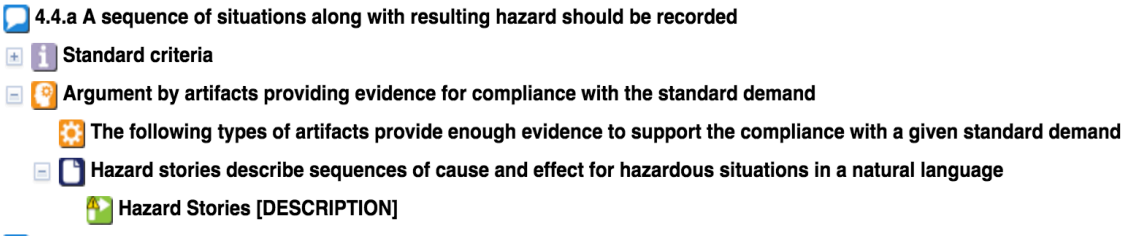


Figure 41 DocBeacon Project Compliance Argument for an excerpt of ISO 14971 standard

As the majority of practices chosen for the project were Scrum-based the students began with planning their work according to this method. In addition to the well-known practices, they were asked to perform the risk management activities, among them the aforementioned hazard stories. Below is an example of a hazard story from the project prepared by the students (Miszczyszyn and Naliwajek, 2016):

"*As a result of user light-heartedness, phone may be lost, which would lead to possible unauthorized access to the app, if the app isn't secured (for example with a pin code)*."

The artefacts collected during the project were used as evidence in the *AS.A.10 Project Compliance Argument.*

8.3.2. ***Results***

The case study eventually finished at the end of February 2016. We have followed the steps of the AgileSafe, except from the *AS.AL.1*, *AS.AL.2* and *AS.AL.3* algorithms, which were not fully developed at this stage of research. The artefacts of AgileSafe were prepared accordingly as specified in the method. All the metrics were collected as planned. A complete set of *AgileSafe assurance cases* was prepared for

ISO 14971 standard. The product was presented to Bright Inventions and accepted as a proof of concept application.

Regarding B.Q1, taking into consideration the collected metrics, the conclusion is that the DocBeacon project managed to keep the agility of software development while applying the selected risk management practices.

When it comes to B.Q2, the answer is less certain. While the selected practices in the *AS.A.6 Project Practices Set* brought a significant level of risk management, they would need to be supported with a few more disciplined practices as well in order to fully comply with ISO 14971, as could be observed in the prepared assurance cases. That being said, the academic character of the project as well as time available and the team size of three people, were the vital constraints for introducing a more complex approach in this case.

All in all, the team perceived the suggested practices as suitable and satisfactorily agile for this kind of small project.

The resulting software development process was assessed by the participants as intuitive and convenient.

Based on the B.M3 metric, the evaluation metric M4 was determined. The practices suggested to the participant were able to cover 75% of the ISO 14971 requirements. This has shown the need to develop an algorithm for AgileSafe method, which would focus on standard compliance.

8.3.3. ***Limitations and validity***

When conducting the study, we were aware of the limitations and threats to validity such setting could introduce.

First of all, the developers involved in the study were all students with a limited experience. Nevertheless, this threat was tackled by engaging the students with some experience in the industry projects, who had been working part time in the companies during their studies.

The time appointed for this project was restricted by the course time-boxes. For this reason, the author of this thesis herself performed some of the activities suggested by the method. It decreased the students' interaction with the method but allowed more feature to be evaluated. It was the trade-off we decided to accept.

8.4. ***CASE STUDY C: GLUCOMET***

The goal (C.G) of this case study was to evaluate the processes of the method from *AS.P.1 Analyse project* to *AS.P.5 Prepare Project Compliance Argument* process. The chosen method for evaluation was GQM. The following questions were formed:

(C.Q1) Does the *AS.AL.3 Algorithm* for *AS.A.13 Suggested Project Practices Set* in the *AS.A.2 Practices Knowledge Base* suggest a set of Practices (*AS.A.13 Suggested Practices Set)* that seem reasonable and potentially applicable to the Project?

(C.Q2) Are the guidelines provided to the User in the Section 5.5 coherent and sufficient for transitioning from *AS.A.13 Suggested Practices Set* to the *AS.A.6 Project Practices Set*?

(C.Q3) Can the *AS.A.5 Practices Compliance Argument* be constructed for the variety of Practices from the *AS.A.2 Practices Knowledge Base*, organizing them in Claims and Facts about their potential compliance?

(C.Q4) Do the processes operate smoothly without any gaps and errors?

The metrics chosen for this case study were as follows:

(C.M1) The completed *AS.A.3 Project Characteristics* for the analysed project

(C.M2) The set of AgileSafe Assurance Arguments

(C.M3) The list of Practices in *AS.A.13 Suggested Practices Set*

(C.M4) The list of Practices in *AS.A.6 Project Practices Set*

8.4.1. ***Collection of Practices in the AgileSafe Knowledge Base***

In order to calibrate the method, the *AS.A.2 Practices Knowledge Base* has been populated with a set of 50 chosen software development practices, ranging from the flagship agile practices to the disciplined, high ceremony ones.

The motivations behind the choice of these practices were as follows:

1. The Practices in the calibrating set should be diverse, coming from both Agile and disciplined approaches. Aim for the ratio between agile and disciplined practices was between 2:3 and 3:2. The exact ratio is difficult to determine due to the lack of a defined line between agility and discipline in some cases.
2. The sources of these Practices should be diverse, for both Agile and disciplined approaches. It meant different Agile methodologies (i.e. Scrum, eXtereme Programming), some hybrid propositions (i.e. Hazard Stories, SafeScrum practices) and various disciplined approaches (i.e. risk management methods, PRINCE 2, BABOK (BABOK: A Guide to the Business Analysis Body of Knowledge, Volume 3, 2015))
3. The calibrating set should contain a number of risk related practices, both agile and traditional, due to the evaluation activities, which concern risk management standards.

The practices were added using an Excel spread sheet import to Protégé tool.

The 50 Practices have been also analysed with respect to their ability to provide necessary evidence for regulatory requirements. The standard chosen for a calibration of the method and mapped into the *AS.A.2 Practices Knowledge Base* was ISO 14971:2007. In order to obtain valuable results with such limited collection, the 50 Practices were carefully chosen with this particular standard in mind. Naturally, other standards can also be added and mapped to the *Knowledge Base*, either in the future research or by the User using Protégé tool.

The list of these 50 Practices can be found in the (AgileSafe, 2018). This basic collection has been used in the evaluation process.

8.4.2. ***The case study***

This case study has been carried out for a fictional Project called GlucoMet. It was based on the project presented to students during the Student's Assessment Case Study (section 8.2) as well as the remarks presented in (Chen et al., 2014) and (Zhang, Jones and Klonoff, 2010).

AS.P.1 Analyse the Project:

The GlucoMet project concerns developing continuous glucose monitoring-enabled insulin pump system. It has been analysed and its *AS.A.3 Project Characteristics* have been described in the Table 68, with respect to the C.M1 metric.

Table 68.Project Characteristics Analysis – GlucoMet

Id	1	
Name	GlucoMet	
Description	Continuous glucose monitoring-enabled insulin pump system	
Regulatory Requirements	ISO 14971	
Characteristics	Factor	Values
	Team Size	**B – From 10 to 50 developers;**
	Geographical Distribution	**B – Same building;**
	Domain Complexity	**D – Complicated;**
	Organisational Distribution	**B – Different teams;**
	Technical Complexity	**B - Multiple technology; D - System/embedded solutions;**

	Organisational Complexity	**B – Flexible, structured;**
	Enterprise Discipline	**A – Project focus;**

The *AS.A.3 Project characteristics* values were chosen for their potential to respond well both to agile and disciplined practices. It is undoubtedly a safety-critical project, operating in a complicated domain but at the same time the size of the team, distribution and complexity of the organization open the possibility to introduce some agility.

AS.P.2 Develop/Update AgileSafe Practices Compliance Argument

The *AS.A.5 Practices Compliance Argument* for ISO 14971 has been updated and extended to correspond with the state of the *AS.A.2 Practices Knowledge Base* and its 50 Practices. Most of these collected Practices were connected in some way with risk management, they were added to the *AS.A.2 Practices Knowledge Base* with this particular standard in mind and therefore most were suitable for ISO 14971 conformance. An excerpt from the *AS.A.5 Practices Compliance Argument* for ISO 14971 is presented in the Figure 42.

Figure 42 An excerpt of Practices Compliance Argument for ISO 14971 in NOR-STA tool

Three of the regulatory requirements at the most detailed level were not covered at all by the Practices from the *AS.A.2 Practices Knowledge Base*: 3.3.a The personnel

should have appropriate qualification and records of this shall be maintained; 4.1.c Additionally the documentation of the risk analysis shall include identification of the person(s) and organization who carried out the risk analysis; 9.b The system should collect and review available information about similar devices. None of the Practices from the *AS.A.2 Practices* Knowledge Base could provide sufficient evidence for these requirements. Nevertheless, these requirements can be easily covered by adding simple activities in the *AS.A.8 Project Practices Compliance Argument*. These results were collected as the evaluation metric M10.

AS.P.3 Select Practices

In the first step, the GlucoMet Project has been added to the *AS.A.2 Practices Knowledge Base* (an instance of a Project concept, *GlucoMet_Proj1*) and its *AS.A.3* Project Characteristics have been defined by adding corresponding properties, using Protégé tool.

At the moment of the Evaluation the *AS.A.2 Practices Knowledge Base* consisted of 50 software development and risk management Practices, which were used as a calibration for the method. The state of the *AS.A.2 Practices Knowledge Base* was satisfying for this kind of project. It has been important to operate on a collection of Practices, which would be relevant to the analysed context.

Once the Project instance has been established, with all the Factor values, the suggestion algorithm has been used to determine the *AS.A.13 Suggested Practices Set* for GlucoMet Project. A following DL expression was used:

inverse (hasPracticeSuggestion) **value** *GlucoMet_Proj1*

As a result, a list of 33 Practices has been suggested (C.M3 metric).

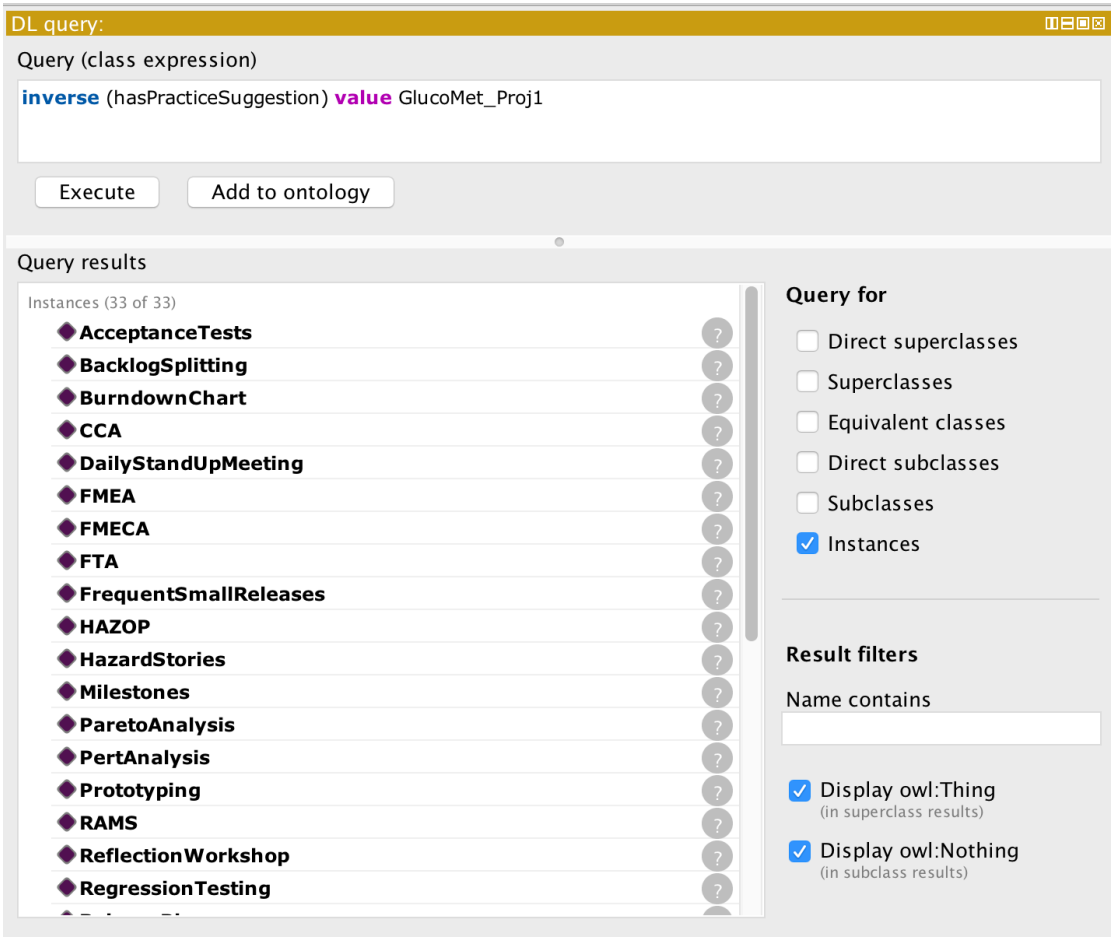


Figure 43 A screen from Protege tool with the Suggested Practices Set

The number of *Suggested Practices* reflects the risk orientation of the set of Practices collected in the *AS.A.2 Practices Knowledge Base*.

In addition to the traditional, disciplined risk management Practices, some agile and hybrid Practices were suggested as well.

The *AS.A.13 Suggested Practices Set* was further analysed in order to develop the *AS.A.6 Project Practices Set*. The objectives for the final choice of Practices were as follows:

1. Favouring agile and hybrid Practices.
2. Following the advice given to Users in the Section 6.4

In the decision process a crucial aspect was also the character of the project, meaning a close relation with hardware.

As a result, the *AS.A.6 Project Practices Set* consisted of 20 Practices (metric C.M4), covering all of the Disciplines and keeping the Requirements coverage from the *AS.A.5*

Practices Compliance Argument. Thus, the metric M9 could be established as well, as 33 Practices in *AS.A.13 Suggested Practices Set* to 20 Practices in *AS.A.6 Project Practices Set.*

The information about *AS.A.6 Project Practices Set* was subsequently added to the *AS.A.2 Practices Knowledge Base*. The list of the Practices is presented in the Figure 44.

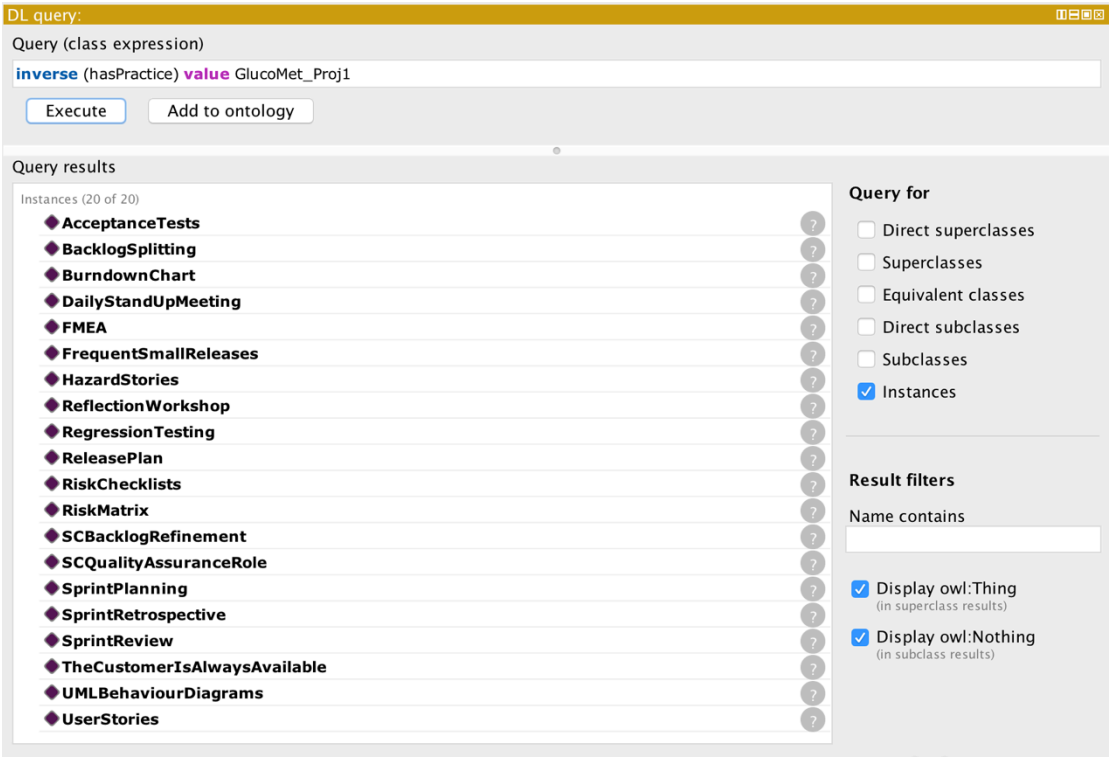


Figure 44 A screen from Protege tool with the Project Practices Set

AS.P.4 Generate AgileSafe Project Practices Compliance Argument

Based on the *AS.A.6 Project Practices Set* and *AS.A.5 Practices Compliance Argument*, a *AS.A.8 Project Practices Compliance Argument* was built.

For the regulatory requirements, which were not covered in the *AS.A.5 Practices Compliance Argument*, some additional activities were presented.

Moreover, the argument was arranged to correspond with the intended use of the Practice and its artefacts in this specific Project.

An excerpt of this argument is presented in the Figure 45.

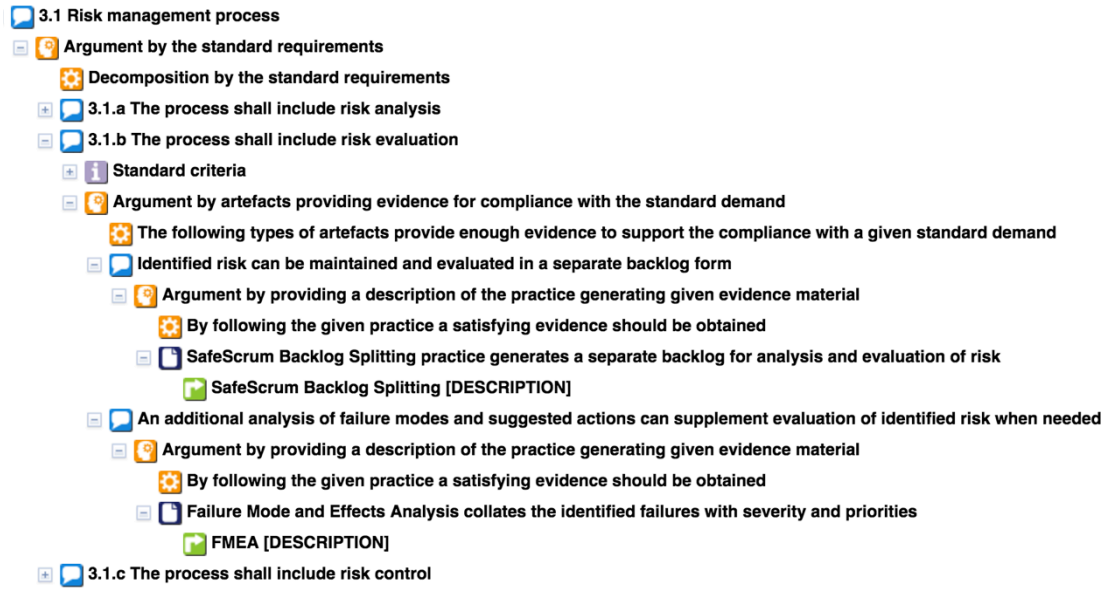


Figure 45 An excerpt of Project Practices Compliance Argument for ISO 14971 in NOR-STA tool

AS.P.5 Prepare Project Compliance Argument

In the next step, a *AS.A.10 Project Compliance Argument* was prepared, based on the *AS.A.8 Project Practices Compliance Argument* and with accordance to *AS.A.9 Project Compliance Pattern*.

An excerpt of this argument is presented in the Figure 46.

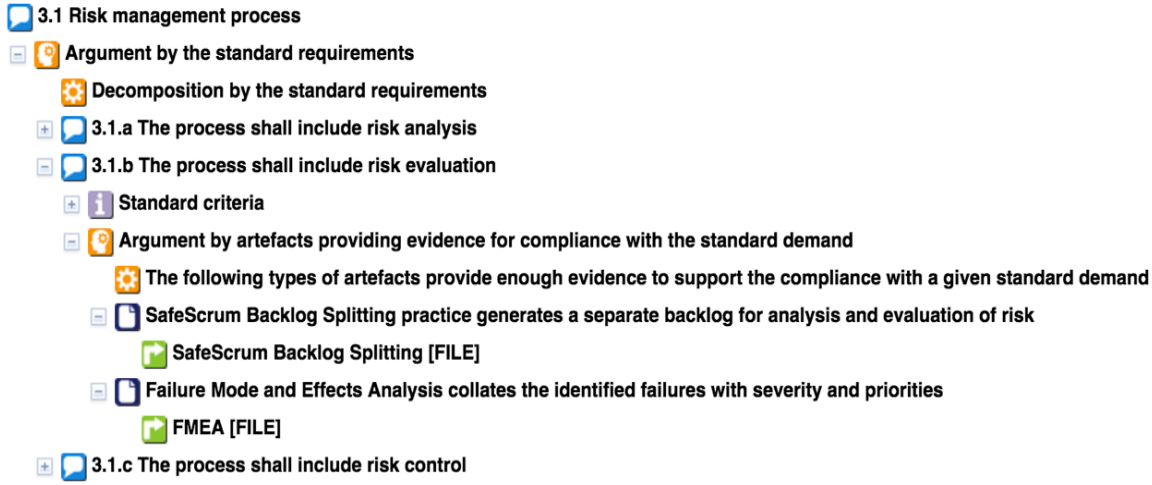


Figure 46 An excerpt of Project Compliance Argument for ISO 14971 in NOR-STA tool

In further processes of AgileSafe method the evidence should be collected and placed in the appropriate nodes of the *AS.A.10 Project Compliance Argument*. The argument should be audited.

Both *AS.A.8 Project Practices Compliance Argument* and *AS.A.10 Project Compliance Argument* could be then used to present the assessor Project's compliance with ISO 14971 standard. The assessor might use the Assessment feature in NOR-STA Argevide tool to assess the provided evidence (Figure 47).

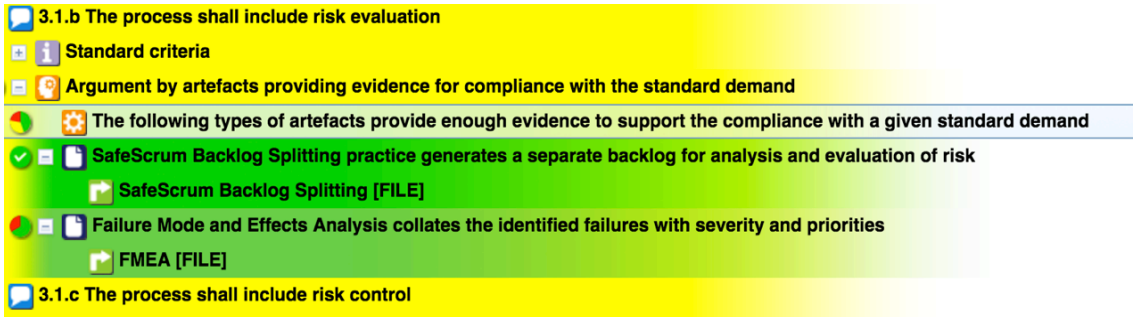


Figure 47: The Asssessment feature in NOR-STA tool for Project Compliance Argument

All of the arguments were collected with respect to the M2 metric.

8.4.3. *Conclusions*

The *AS.A.13 Suggested Practices Set* for the Project was relevant to the *AS.A.3 Project Characteristics* and the Practices seemed applicable to the Project, which forms an answer to the question C.Q1. What is more, in relation to the C.Q2 question, the guidelines for the transition between *AS.A.13 Suggested Practices Set* to the *AS.A.6 Project Practices Set* were useful, albeit it was helpful to know how much of the agility a User want to introduce.

Due to the targeted character of the calibrating set, most of the Practices from the *AS.A.2 Practices Knowledge Base* were arranged into the *AS.A.5 Practices Compliance Argument* and they were arranged into the structure without problems (question C.Q3). That being said, a thorough reflection and analysis of the Practices is crucial for the process.

To sum up and answer question C.Q4, the transitions between subsequent processes were smooth and continuous.

8.4.4. *Limitations and validity*

The main limitation and possible threat to validity is the fact, that the author herself performed the study. It means that the results might not be generalized so easily to potential users of the method. This choice was dictated by the limitations coming from the tool solutions used at this stage of the method, especially the knowledge base part.

In order to avoid potential distortions of the results coming from the tool skills, such decision was taken.

Obviously, for better results, the method should be implemented in a real-life project, which is in scope of the future research plans.

8.5. EVALUATION: INTERVIEWS WITH DOMAIN EXPERTS

Next step of the evaluation process was aimed at gathering opinion of the experts and practitioners in the field of project management, software development methodologies and safety-critical systems, about the AgileSafe.

This step was divided into the following categories:

- Interview with the authors of a hybrid agile method for safety-critical projects, SafeScrum (Mycklebust, Stålhane and Hanssen, 2016)
- Interview with practitioners from a safety-critical software company Autronica (Autronica Fire and Security AS, 2017)
- Interviews with experts - the questionnaire

They represented different points of view, which are relevant for the method: the process engineering, the industry and the safety assessment.

8.5.1. SafeScrum

The goal of this step was to obtain opinions on the AgileSafe method from the perspective of IT process-engineering specialists.

The specialists involved in this step of evaluation were two of the authors of a hybrid agile method for safety-critical projects called SafeScrum (Mycklebust, Stålhane and Hanssen, 2016).

SafeScrum is a method based on Scrum with added roles and Practices for safety management and compliance with relevant standards. The overview of the method's process is presented in the Figure 48.

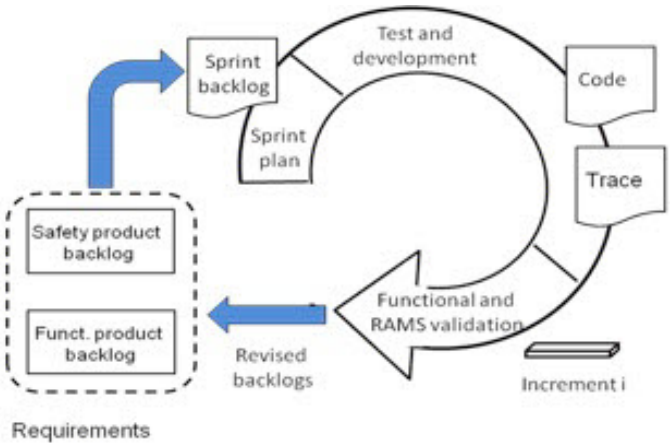


Figure 48 Scrum's role in safety critical software development (Mycklebust, Stålhane and Hanssen, 2016).

SafeScrum has been developed in SINTEF (SINTEF, 2017) with cooperation of Norwegian University of Science and Technology.

There were several meetings held between SafeScrum team and AgileSafe author, in 21-24 March 2017 and 4-8 September 2017.

The main participants of the meetings were:

- Thor Myklebust (SINTEF)
- Geir Kjetil Hanssen (SINTEF)
- Katarzyna Łukasiewicz (Gdańsk University of Technology)

During these meetings the AgileSafe method was presented in detail to the SafeScrum authors, along with the tools used in the process. The presentation was followed with discussions on the possible adoption of elements of AgileSafe method as a framework for SafeScrum and the potential uses of AgileSafe.

The conclusions most relevant to the evaluation of AgileSafe were as follows:

1. The SafeScrum authors assessed the *AS.A.14 Practice description* template and the way it is represented in the *AS.A.2 Practices* Knowledge Base. They found the chosen Characteristics and their Values relevant and the opinion has been noted as metric M5.
2. They agreed on introducing SafeScrum Practices to the *AS.A.2 Practices Knowledge Base* and to use them in further evaluation of the method (Figure 49).

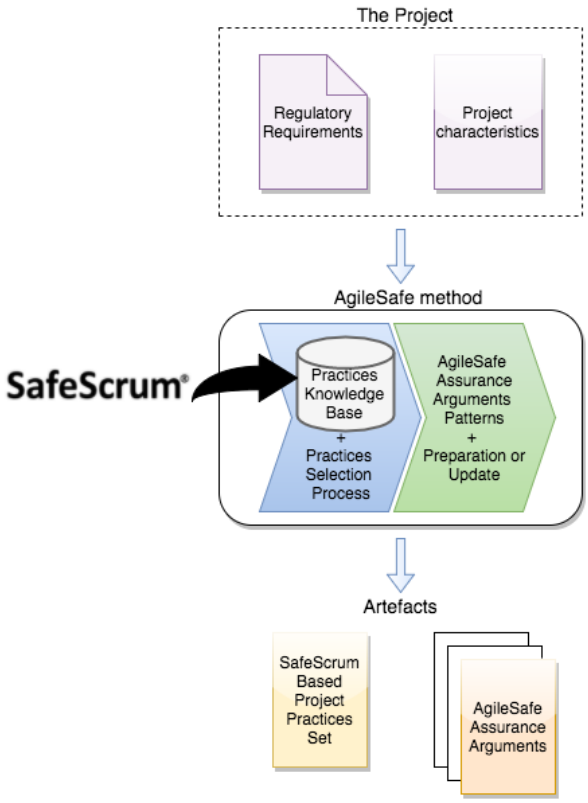


Figure 49 Introduction of SafeScrum practices to AgileSafe Knowledge Base

3. The assessment of SafeScrum Practices using AgileSafe *AS.A.14 Practice description* were consulted with the authors and as such, they can be regarded as verified *Practice descriptions* in *Knowledge Base*. An example of a SafeScrum *Practice description* is given in Table 69: SafeScrum Backlog Splitting .

Table 69: SafeScrum Backlog Splitting Practice description

Id	28
Name	SafeScrum Backlog splitting
Description	"In SafeScrum, all requirements are split into safety critical requirements and other requirements and inserted into separate product backlogs. Alternatively, the safety requirements are tagged. Adding a second backlog is an extension of the original Scrum process and is needed to separate the frequently changed functional requirements from the more stable safety requirements. With two backlogs we can keep track of how each item in the functional product backlog relates to the items in the safety product backlog, i.e. which safety requirements that are affected by which functional requirements. This can be done by using simple cross-references in

	the two backlogs and can also be supported with an explanation of how the requirements are related if this is needed to fully understand a requirement. The staffing of the Sprint team and the duration of the sprint (30 days is common), together with the estimates of each item decides which items that can be selected for development. Sometimes also e.g. the Safety responsible or the RAMS responsible takes part in the selection of which items have to be prioritized. “	
Discipline	Architecture	No
	Deployment	No
	Development	Yes
	Environment	No
	Project Management	Yes
	Requirements	Yes
	Test	No
Capability	Factor	Values
	Team Size	A – Under 10 developers; B – From 10 to 50 developers; C – From 50 to 100 developers; D – 100’s of developers;
	Geographical Distribution	A – Co-located; B – Same building; C – Some working from home; D – Within driving distance; E – Globally distributed
	Domain Complexity	A – Straightforward; B - Predictable; C – Quickly changing; D – Complicated; E – Intricate/Emerging
	Organisational Distribution	A – Collaborative; B – Different teams; C – Different departments; D – Different partner companies; E – Contractual
	Technical Complexity	A – Homogenous; B - Multiple technology; C – New technology; D - System/embedded solutions;
	Organisational Complexity	A – Flexible, intuitive; B – Flexible, structured; C – Stable, evolutionary; D – Stable, planned;
	Enterprise Discipline	A – Project focus; B – Mostly project focused; C – Balanced; D – Mostly enterprise focused;

Used in:	Name of the Regulation and regulatory requirement	General Practice	Fact
	ISO 14971 / 3.1.b The process shall include risk evaluation	Identified risk can be maintained and evaluated in a separate backlog form	SafeScrum Backlog Splitting practice generates a separate backlog for analysis and evaluation of risk

Overall, the method was very well received, with much interest. The most valuable element of the method, from SafeScrum team point of view, was the set of *AgileSafe assurance arguments* as a mean to monitor safety as well as to present better the Evidence and conformance with a standard to the assessors.

8.5.2. ***Autronica***

Autronica Fire and Security is an international company based in Trondheim in Norway, producing fire and gas safety solutions (Autronica Fire and Security AS, 2017). The company has become interested in revising their software development processes and inspired by their project managers they have begun their cooperation with SINTEF and SafeScrum group in order to introduce some agility in their teams.

The goal of the interview was to establish what might be the industry's attitude towards AgileSafe. As a safety-critical company with an interest in agile approach, Autronica might be treated as a target of the AgileSafe method.

The interview took place at 22 March 2017 at the Autronica premises in Trondheim, Norway and lasted for about two hours.

The case study concerns two of Autronica projects: Autrosafe and AutroMaster. Each of the developers represented one of these projects. Both projects are safety-critical and need to be compliant with specific standards and norms.

They have implemented some agile SafeScrum practices in their projects and have been working on their hybrid agile process for 4 years at the time of writing.

People attending the meeting:

- Katarzyna Łukasiewicz (Gdańsk University of Technology)
- Thor Myklebust (SINTEF)
- Two Autronica Project managers/developers

In the course of the meeting the attendants were at first presented the outline of AgileSafe, including a live presentation of assurance arguments in NOR-STA Argevide tool and were allowed to ask questions about the method in order to fully comprehend it.

Secondly, the attendants from Autronica were asked three questions, one by one and answered them collaboratively in a brainstorm manner. Katarzyna Łukasiewicz was responsible for recording the answers.

The questions along with their answers are presented below:

1. **What were the incentives for introducing agile practices into the company? What problems did you wish to solve?**

   Both developers have been in favour of using more agile approach in safety-critical software development. Their experience both from previous projects and the current ones led them to some reflections and the need for change in the existing approach. The main problems they both observed in the more disciplined, plan-driven approach were:

   a. Need for shorter, day-to-day goals rather than big tasks that might take months to accomplish
   b. Documentation load in the disciplined approach for safety-critical software. In their opinion developers should be able to focus on writing the code rather than writing the documents. They observed that a lot of the documents could be obtained from the tools they have been using.
   c. Need for more frequent releases and shorter time for introducing changes in the system. They noticed that the actual changes in products are cheaper and faster than they used to be because systems rely increasingly on software thus the changes are made mainly in the software part rather than the hardware. The whole process could be faster, and the guidance did not take this into consideration. What is more, in their opinion delivering updates faster, in the agile way, is actually safer because sometimes the changes may solve some potentially fatal errors and as such they should be released as quickly as possible.
   d. Difficulties in managing co-located teams with disciplined approach. They noticed that agile practices tackled the problem of globally distributed teams in much better way, introducing modern and practical solutions along with the tools enabling remote cooperation.
   e. Need for more manageable approach to traceability that would allow keeping the information with less documentation around it.

2. **Which agile practices do you currently use in your projects**?
    a. Stand-ups
    b. Sprint Planning
    c. Sprint Reviews
    d. Sprint Retrospectives
    e. FMEA
    f. Safety Manuals
    g. Code reviews
    h. Issue reviews
    i. Quality Assurance Role (SafeScrum)
    j. Peer reviews
    k. RAMS Engineer (SafeScrum)
    l. Backlog Refinement (SafeScrum)
    m. Automatic tests – one of the projects, the second one is an older project with manual tests load

3. **Do you think that AgileSafe might be a useful approach to introducing agile practices in the safety-critical industry? Which parts of the method do you find most interesting and potentially useful?**

    The developers found the AgileSafe method interesting and some of the elements they found applicable and potentially helpful in their company. In their opinion the most beneficial aspects of AgileSafe are:
    a. As a whole it may be good initial help for developing the process, a new one or updated.
    b. It may serve as tool for visualising the need for specific practices to the team – developers can see in what way their work is needed for the project.
    c. Convincing the management or other bodies involved in the project that the more agile approach might work and satisfy the necessary safety requirements.
    d. Good way to organise the evidence, to make sure you're are collecting the right things.
    e. Developers might get the broader perspective and distance themselves from the code to see the bigger picture.
    f. The *AS.A.2 Practices* Knowledge Base is a good idea, even more if it can be shared in the agile community.

g. The method could be well used by the SafeScrum RAMS Engineer- the person responsible for reliability, availability, maintainability and safety.

The answers for the question number 2 has been collected with respect to the metric M6 and the answers for the question 3 were collected as part of the metric M7.

Finally, the two Project Managers were asked to perform the analysis of their projects, based on the AgileSafe template:

Table 70.Autronica Autrosafe Project Analysis

Id	Autro.1	
Name	Autrosafe	
Description	Fire detection system	
Regulatory Requirements	IEC 61508, EN54-2, UL, SOLAS, National Standards	
Project Characteristics	Factor	Values
	Team Size	**A – Under 10 developers**;
	Geographical Distribution	**E – Globally distributed**
	Domain Complexity	**D – Complicated;**
	Organisational Distribution	**C – Different departments**;
	Technical Complexity	**D - System/embedded solutions;**
	Organisational Complexity	**E – Rigid**
	Enterprise Discipline	**A – Project focus;**

Table 71.Autronica AutroMaster Project Analysis

Id	Autro.2
Name	AutroMaster
Description	Referred to as a “top system”, AutroMaster is a graphical user interface mainly used for maintaining, configuring and controlling our fire detection systems.
Regulatory Requirements	IEC 61508, EN54-2, UL, SOLAS, National Standards

Project Characteristics	Factor	Values
	Team Size	**A – Under 10 developers**;
	Geographical Distribution	**E – Globally distributed**
	Domain Complexity	**D – Complicated;**
	Organisational Distribution	**C – Different departments**;
	Technical Complexity	**B - Multiple technology;**
	Organisational Complexity	**E – Rigid**
	Enterprise Discipline	**B – Mostly project focused;**

The AgileSafe template for *AS.A.3 Project Characteristics* was assessed as clear and relevant to these two Autronica projects.

All in all, the method has been received positively, with much interest. What is important, the company expressed interest in future cooperation as well.

8.5.3. ***Questionnaire for the experts***

In order to evaluate the potential consequences of AgileSafe method application in the industry, an indirect method of an interview has been selected. The goal was to investigate the potential applicability from a perspective of experts in the software development methods and safety aspects. The process consisted of the following steps:

1. Determine a group of the experts.
2. Contact the experts.
3. Present the AgileSafe method in person to the experts, for better understanding.
4. Collect the feedback from the experts.

In the process, a list of five European experts was prepared. It included both academics and practitioners, with experience in applying software development methods in the safety-critical industry as well as a member of the IEC standard committee. These experts were contacted and presented the method in person.

In order to perform Step 4, a questionnaire was prepared. It was distributed in electronic form of an editable PDF file. In the course of the interviewing process, five questionnaires were sent back with the answers.

Below are presented the questions from the questionnaire along with the summary of provided answers:

1. The aim of AgileSafe method is to provide safety-critical software companies with a solution allowing them to incorporate agile practices into their software development process while still maintaining the compliance with the software assurance requirements imposed by the application domain. How do you rate relevancy of this objective to the safety-critical industry?

All respondents recognized the relevancy of the objective of AgileSafe method as high and very high, pointing out that with rise of agile methods this is a current problem in the safety-critical industry.

2. In your opinion, are the characteristics used in AgileSafe to describe Project and Practices in the knowledge base suitable for suggesting a set of practices that would be helpful during the definition of a customised, more agile, software development process?

The Project Characteristics were assessed as suitable, albeit with suggestions for improvement. The suggestions included: not allowing the size of the team to exceed 20 people, clarifying the terms and metrics used and including the starting level/experience in agility.

3. Do you think that the form of representation of regulatory requirements proposed in the AgileSafe method can provide the necessary means for monitoring and supporting the conformance with safety requirements?

The use of assurance arguments in AgileSafe was regarded as well suited, enabling a more structured representation of the regulatory requirements, although one respondent suggested that it might depend on the assessor as well.

4. Do you think that AgileSafe method might be successfully adopted within the safety-critical industry?

Most of the respondents concluded that yes, it can and one stated that it is difficult to say as it depends on the relationship with the companies and their willingness to cooperate. The respondents noted that the scale of success would depend on the personnel involved from the User side as well as the quality of the Knowledge Base and its Practices.

5. What do you perceive as the main potential benefits of introducing AgileSafe method to a project?

The main potential benefits recognized by the respondents were: encouraging compliance driven evidence, increasing motivation and satisfaction of the project team,

better understanding of the impact of agile practices on safety/security, improving safety case, increasing the probability of assessor acceptance and ensuring the time and cost efficiency.

6. *What issues do you see as potentially problematic while implementing the method in a project?*

The concerns raised by the respondents included: redundancy in *AS.A.13 Suggested Practices Set*, limited number of Practices and *AS.A.5 Practices Compliance Arguments* ready to use at this stage, the need to engage management, combining the tools to work together, the fact that human factor might fail as well as the perception of the initial workload.

7. *Do you have any additional comments?*

One respondent suggested that it might be beneficial to the method to share with larger public both the *Knowledge Base* and the *AS.A.5 Practices Compliance Argument* in order to gain the feedback and assessments of claims and facts. This way the method can be "taught" and improved. Another respondent mentioned the need to develop the method further with closer cooperation with key stakeholders for better fit with the industry. The need for a demonstrative example was also mentioned, by another respondent.

To sum up, the experts' opinion was positive. They recognized the value of AgileSafe approach to the industry and acknowledged potential benefits. What is more, they made some remarks on features that they would like to see improved and suggested the directions for future research.

8.5.4. ***Limitations and validity***

Some threats to validity may be raised. The number of experts questioned in this study is as low as 5, which was dictated by the quality imperative, nevertheless gives a limited sample of the opinions experts in the domain may present. What is more, the choice of the experts for this study might have been biased, as well as the responses themselves because the method was presented first by the author personally in all cases. This kind of personal approach might have an impact in some way. However, all of the experts taking part in the study are well-respected researchers and practitioners and as such their answers could be treated as reliable.

The answers in this questionnaire formed the basis for the evaluation metrics M7 and M8.

8.6. EVALUATION CONCLUSIONS

The results of the evaluation were as follows:

(Q1) *What is the potential impact of introducing agile practice to a safety-critical software development process?*

The metric M1 revealed that there are substantial benefits that might be expected from introducing certain agile practices to a safety-critical software development, with reduction of cost, reduction of time-to-market and enhancements in quality being frequently mentioned in the reports. The potentially negative aspects of applying agile practices included incomplete risk management and unsatisfying documentation as evidence in the conformance processes. The metric M2 confirmed these worries by showing that 55% of the Scrum and eXtreme Programming practices combined were assessed as carrying high risk when implemented in safety-critical software projects. That being said, this risk could be reduced by introducing additional practices, as indicated in the metric M3.

Based on the metric M1, M2 and M3 we can conclude that the prospect of potential benefits stemming from the introduction of agile practice to a safety-critical software development is a valid incentive for doing so, although the risk that such introduction might carry should be tacled by introducing additional safety-oriented practices.

(Q2) *Is AgileSafe method capable of reducing the negative impact agile practices might have on the safety-critical software development process?*

In the metric M3 it was indicated that a tailored composition of some software development practices might reduce the negative impact that the introduction of agile practices might have on the process. It can be assumed that AgileSafe method, by suggesting appropriate practices in addition to the agile practices, might reduce such negative impact. In the metric M7 the potential of AgileSafe to support risk management activities was indicated as one of its benefits, as well as the support in the process of selecting practices for the software development process. At the same time, in the metric M8 the criticism did not concern the ability of AgileSafe to reduce the potential negative impact of agile practices.

(Q3) *Does the method suggest practices which are relevant to a project and its environment?*

The metric M5 was positive about the form of presenting practices in the Knowledge Base. Taking into consideration metric M6, the list of practices currently used in these industry projects might have been as well suggested by the AgileSafe algorithms. While the metric M7 shows that there is a potential in AgileSafe of suggesting relevant and appropriate practices, the metric M8 also indicates some issues with the suggestions being only as sound as the Practices stored in the Knowledge Base. This makes AgileSafe dependent on the human factor. Nevertheless, the metric M9 collected in a controlled environment, presents the ability of AgileSafe to produce a satisfying outcome.

(Q4) *Does the method support the safety assurance aspect of a project when introducing new practices?*

At first, the metric M4 showed that the AgileSafe in its older version needed vital improvements when it comes to suggesting the practices that might respond to the needs of the standards, although the assurance argument representation felt correct. Metric M10, collected with the AgileSafe in its current form, shows that the safety assurance aspect is well catered for by the method's algorithms and the introduction of new practices, with some input from the user, can be well managed using AgileSafe method.

(Q5) *Do an introduction of agile practices into a software development process carry a potential benefit of reducing effort needed for a project?*

Based on metric M1 the introduction of agile methods can reduce the overall effort needed for a project, reducing the cost and time-to-market. It was also confirmed in the M6 metric. The practices being used in two industry projects were chosen, among other reasons, for their potential to reduce the effort.

(Q6) *Does AgileSafe method allow for an introduction of agile practices that can potentially reduce the effort needed for a project?*

The metric M7 shows that AgileSafe might support the effectiveness and quality of work within the team. It also shows it supports the introduction of agile practices to a safety-critical project. As indicated in the answer to Q5, assuming that agile practices enable effort reduction, the method allows the introduction of such practices. While in the metric M8 the initial workload devoted to the method itself was mentioned, it does not affect in a negative way the practices themselves and their ability to reduce effort.

The sources of the data for specific metrics are presented in the Figure **50**.

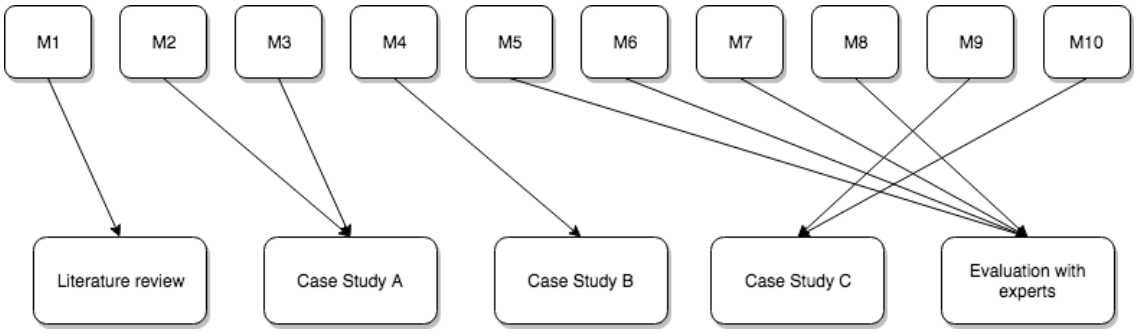


Figure 50 Metrics with their data sources

9. CONCLUSIONS

The decision on which software development practices to implement in a project can decide about the ultimate success of the project. Introduction of the AgileSafe method for practices selection has been an attempt to increase the chances of selecting the most beneficial practices for the safety-critical projects.

9.1. CONTRIBUTION OF THE RESEARCH

The main achievement of the research is the development of AgileSafe method for selecting software development practices and monitoring conformance. The main innovations of the AgileSafe method are:

1. A comprehensible framework for practices selection in the critical software domain, from the definition of the Project to its finishing phase and assessment.
2. Application of the concept of evidence-based argumentation to represent the safety requirements imposed by the relevant standards and regulations and introduction of a three-level structure of assurance arguments to support conformance achieving and monitoring (*AS.A.5 Practices Compliance Argument, AS.A.8 Project Practices Compliance Argument, AS.A.10 Project Compliance Argument* and their patterns).
3. Specification of the *AS.A.2 Practices Knowledge Base* and the related reasoning algorithms supporting selection of the Practices for a given Project (AS.AL.3)

To validate the proposed AgileSafe method, in the course of the research, three case studies (Case Study A, Case Study B and Case Study C, presented in Section 8) have been conducted. The results of these case studies have been partially published in (Górski and Łukasiewicz, 2012a, 2012b, 2013; Łukasiewicz, 2017; Łukasiewicz and Górski, 2018).

In the course of the case studies some new agile practices related to risk management have been discovered, one of the already has been introduced to the set of recommended practices of the SafeScrum methodology (Hanssen, G. K., Stålhane, T., Myklebust, T., 2018).

The research and resulting AgileSafe method have already generated a considerable interest in the industry. This resulted in a grant from Polish-Norwegian Research Programme for bilateral relations and an ongoing cooperation with a research team in SINTEF Trondheim. The author of this thesis has been an active member of the agile in safety community and has been asked to join the programme committee of the

Agile Development of Safety-Critical Software (ASCS) 2018 workshop, held as a part of the Agile Alliance XP 2018 conference. The author is also a programme committee member of the International Conference on Lean and Agile Software Development in 2017 and 2018.

9.2. THESIS EVALUATION

The following strategy to demonstrate the thesis has been adopted:

1. Based on literary studies, justify that introduction of agile practices results in cost reduction and acceleration of manufacturing processes.
2. Propose a method (AgileSafe) of introducing agile practices to safety-critical software development while maintaining the ability to demonstrate safety.
3. Evaluate the method by case studies and expert assessments.

The evaluation of AgileSafe was performed following the GQM (Goal, Question, Metrics) approach. The goals of evaluation were following;

(G1) *Analyse whether the proposed method supports introduction of agile practices into a software development process while still maintaining the compliance with the software assurance requirements imposed by the application domain.*

This goal has been reached by answering questions Q1, Q2, Q3 and Q4. Based on them we can conclude that AgileSafe supports introduction of agile practices into a software development process while still maintaining the compliance with the software assurance requirements.

(G2) *Analyse whether the proposed method makes it possible to reduce the effort devoted to software development processes in safety related projects.*

This goal has been demonstrated by answering questions Q5 and Q6. From these we can conclude that agile practices can reduce the effort needed for a project and AgileSafe, by supporting an introduction of such practices, makes it possible to reduce the effort devoted to software development processes in safety related projects.

It forms the basis of a conviction that the thesis of this research:

The proposed method makes it possible to reduce the effort devoted to software development processes in safety related projects, without compromising the requirements of applicable norms and standards.

Has been proved.

9.3. VALIDITY

The thesis of this research has been evaluated based on the series of studies described in detail in the Section 8. The validity of the separate studies has been discussed in the corresponding chapters presenting each of the study. In this chapter, the overall validity of proving the thesis will be addressed.

As mentioned in the previous chapter, the reasoning behind the applied evaluation approach was that, if agile practices can reduce the effort needed for a project, then AgileSafe method by supporting an introduction of such practices supports the reduction of effort as well. There may be some limitations and possible threats to validity when implementing this reasoning.

This way of indirect evaluation does not bring the same solid evidence as a direct evaluation performed in a real-life project, with measurable effects in effort reduction (i.e. a comparative study). Nevertheless, the indirect evaluation was chosen for the following reasons:

- Time constraints related with doctoral research
  The implementation of the method in a real-life project and collecting the needed metrics might take a few years, which would exceed the time that could be formally devoted to the doctoral research. Nevertheless, this is one of the goals for the future work.
- The safety-critical nature of application domain
  The organizations that develop safety-critical software are cautious when introducing new methods and rightly so. However, this causes difficulties with finding a partner for evaluation. In this research, we found a partner in Norway, willing to evaluate the method, but firstly on a theoretical level (Section 8.5.2) with a possibility to broaden the cooperation. Nevertheless, this kind of negotiations is time consuming and, as mentioned in the previous paragraph, the time constraints allowed only for that first step to be included in the doctoral thesis.

Nevertheless, the interest in AgileSafe already generated in the research and industrial community is a good prognostic for a possibility of its application in a full-scale real-life software development project in a critical domain.

9.4. FUTURE WORK

The research on the AgileSafe method and ideas presented in it is to be continued. Areas of future work might include:

- An online tool for handling *AS.A.2 Practices Knowledge Base* and *AS.P.3 Select practices* more easily, without the use of specific languages or tools
- Sharing with community the *AS.A.2 Practices Knowledge Base* and collect the Practices from other sources to increase the number of available Practices
- An evaluation of the method in a real-life safety-critical project
- Develop better and faster ways to connect the *AS.A.6 Project Practices Set* with NOR-STA tool (i.e. XML exports and imports), in order to automate the process of connecting the Practices from the *AS.A.2 Practices Knowledge Base* with the arguments in NOR-STA.
- Working on a broader base of pre-defined *AS.A.5 Practices Compliance Arguments* for a larger number of standards
- Getting experts' opinions on *AS.A.5 Practices Compliance Arguments* with NOR-STA appraisal mechanisms and using it to "teach" the AgileSafe method
- Adapting AgileSafe for security domain – the directions for this have been outlined in (Górski and Łukasiewicz, 2017) and are currently continued in a master's thesis project
- Applying AgileSafe in a real-life project and direct measurement of the benefits resulting from the method.

REFERENCES

AAIB (Air Accidents Investigation Branch) (2005), Aircraft Accident Report 4/2007 - Airbus A340-642, G-VATL, 9 February 2005, [online] Available at: https://www.gov.uk/aaib-reports/aar-4-2007-airbus-a340-642-g-vatl-9-february-2005 (Accessed: May 2018)

Abbott (2017), [online] Available at: http://www.abbott.com/ (Accessed: January 2017)

Abrahamsson, P., Salo, O., Ronkainen, J. & Warsta, J. (2002) Agile software development methods: Review and analysis, VTT publication 478, Espoo, Finland, 107p.

Agile Manifesto, (2001). *Manifesto for Agile Software Development*. [online] Available at: http://agilemanifesto.org. (Accessed: June 2018)

AgileSafe (2018), List of 50 Practices, [online] Available at: http://agilesafe.eu/ (Accessed: August 2018)

AgileTek, (2011). [online] Available at: http://www.agiletek.com. (Accessed: April 2016)

Alleman, G., Henderson, M. and Seggelke, R. (2003). Making Agile Development Work in a Government Contracting Environment - Measuring velocity with Earned Value. In: *Agile Development Conference*. Washington, DC: IEEE, pp.114 - 119.

Ambler, S. (2010). *IBM agility@scale: Become as Agile as You Can Be*. IBM Global Services.

Ambler, S. (2012), *Summer 2012 DDJ State of the IT Union Survey*. [online] Available at: http://www.ambysoft.com/surveys/stateOfITUnion201209.html (Accessed: May 2017)

Animas Insulin Pumps, (2012). *OneTouch Ping®*. [online] Available at: http://www.animas.com/animas-insulin-pumps/onetouch-ping. (Accessed: November 2012)

Apple.com, *Your heart rate. What it means, and where on Apple Watch you'll find it*. [online] Available at: https://support.apple.com/en-us/HT204666 (Accessed: June 2016)

Argevide (2017), [online] Available at: http://www.argevide.com/ (Accessed: December 2017)

Association for the Advancement of Medical Instrumentation, (2011). *AAMI TIR45/Ed.1, Guidance on the use of agile practices in the development of medical device software*.

Astah.net. (2017). *Astah GSN Editor Overview | Astah.net*. [online] Available at: http://astah.net/editions/gsn . (Accessed: March 2017)

ATSB (Australian Transport Safety Bureau) (2005), In-flight upset event 240 km north-west of Perth, WA, Boeing Company 777-200, 9M-MRG, 1 August 2005, [online] Available at: http://www.atsb.gov.au/media/24550/aair200503722_001.pdf (Accessed: May 2018)

Autronica Fire and Security AS, (2017) [online] http://www.autronicafire.com/ (Accessed: February 2017)

BABOK: A Guide to the Business Analysis Body of Knowledge, Volume 3. (2015). IIBA.

Babuscio, J. (2009). How the FBI Learned to Catch Bad Guys One Iteration at a Time. In: *Agile Conference*. IEEE, pp.96-100.

Banner, M.G., Fenn, J.L., Hawkins, R.D., Kelly, T.P., Oakshott, Y., & Williams, P.J. (2007). The Who, Where, How, Why and When of Modular and Incremental Certification Representing the Industrial Avionics Working Group.

Bishop P.G., Bloomfield R.E. (1995) The SHIP Safety Case Approach. In: Rabe G. (eds) Safe Comp 95. Springer, London

Bishop, P. G., Bloomfield, R. E. (1998). A Methodology for Safety Case Development. In: F. Redmill & T. Anderson (Eds.), Industrial Perspectives of Safety-critical Systems: Proceedings of the Sixth Safety-critical Systems Symposium, Birmingham 1998.

Bloomfield, R. and Bishop, P. (2010). Safety and Assurance Cases: Past, Present and Possible Future – an Adelard Perspective. In: *Making Systems Safer. Proceedings of the Eighteenth Safety-Critical Systems Symposium*. London: Springer, pp.51-67.

Bloomfield, R., Chozos, N. and Cleland, G. (2012). *Supplement G: Safety case use within the medical devices industry*. Supplements to: Using safety cases in industry and healthcare. [online] London: The Health Foundation, pp.G2-G17. Available at: http://www.health.org.uk/sites/default/files/UsingSafetyCasesInIndustryAndHealthcare_supplements.pdf.

Boehm, B. & J. Hansen, W. (2000). Spiral Development: Experience, Principles, and Refinements

Boehm, B. (2002). Get ready for agile methods, with care. *Computer*, 35(1), pp.64-69.

Boehm, B. and Turner, R. (2003). *Balancing agility and discipline*. Boston: Addison-Wesley.

Bright Inventions (2017), [online] Available at: http://brightinventions.pl/ (Accessed: July 2017)

Brooks, Fred P. (1987). No Silver Bullet Essence and Accidents of Software Engineering. *IEE Computer*, 20(4), pp.10-19.

Bulska, K. and Miler, J. (2010). Łączenie zwinności metodyki Scrum z dojrzałością modelu CMMI (Integration of the agile Scrum practices with the maturity of CMMI).

In: *XII National Conference of Software Engineering (KKIO)*. Pomorskie Wydawnictwo Naukowo-Techniczne, pp.89-96.

Software Engineering Institute (2006) CMMI–DEV: CMMI for development, V1.2 model, CMU/SEI-2006-TR-008

Software Engineering Institute (1993) Capability Maturity ModelSM for Software, Version 1.1, CMU/SEI-93-TR-024

Chen, Y., Lawford, M., Wang, H. and Wassyng, A. (2014). Insulin Pump Software Certification. *Foundations of Health Information Engineering and Systems*, pp.87-106.

Cyra, L. and Górski, J. (2011). Support for argument structures review and assessment. *Reliability Engineering & System Safety*, 96(1), pp.26-37.

Cyra, L. and Gorski, J. (2011). SCF — A framework supporting achieving and assessing conformity with standards. *Computer Standards & Interfaces*, 33(1), pp.80-95.

Denney E., Pai G., Pohl J. (2012) AdvoCATE: An Assurance Case Automation Toolset. In: Ortmeier F., Daniel P. (eds) Computer Safety, Reliability, and Security. SAFECOMP 2012. Lecture Notes in Computer Science, vol 7613. Springer, Berlin, Heidelberg

Designsafe.(2012) *Designsafe*. [online] Available at: http://www.designsafe.com/. (Accessed: May 2012)

Diaz, J., Garbajosa, J. and Calvo-Manzano, J. (2009). Mapping CMMI Level 2 to Scrum Practices: An Experience Report. In: *Proceedings of 16th European Systems and Software Process Improvement and Innovation Conference (EuroSPI)*. Berlin: Springer Berlin Heidelberg, pp.93-104.

Drobka, J., Noftz, D. and Raghu R., (2004). Piloting XP on four mission-critical projects. *IEEE Softw.*, 21(6), pp.70-75.

Earned Value Management (1967). *Earned Value Management*. [online] Available at: http://www.earnedvaluemanagement.com/. (Accessed: November 2016)

Elmqvist, J., Nadjm-Tehrani, S., Forsberg, K. and Nordenbro, S. (2008). Demonstration of a formal method for incremental qualification of IMA systems. *2008 IEEE/AIAA 27th Digital Avionics Systems Conference*.

Emmet, L. and Cleland, G. (2002). Graphical notations, narratives and persuasion: a Pliant Systems approach to Hypertext Tool Design. *Proceedings of the thirteenth ACM conference on Hypertext and hypermedia - HYPERTEXT '02*, pp.55-64.
ERM - Workshop on Selected Problems in Environmental Risk Management and Emerging Threats, 2009. Proc. of the Workshop on Selected Problems in Environmental Risk Management and Emerging Threats, June 2009, Gdansk, Poland [online] Available at: http://kio.pg.gda.pl/ERM2009/

Extreme Programming: A gentle introduction. (1999) [online] Available at: http://www.extremeprogramming.org/ (Accessed: July 2018)

Faller R., Goble W. M. (2007). Open IEC 61508 Certification of Products, exida GmbH.

Fda.gov. (2017). *U S Food and Drug Administration Home Page*. [online] Available at: http://www.fda.gov/ (Accessed June 2017).

Food and Drug Administration, (2014). *Infusion Pumps Total Product Life Cycle. Guidance for Industry and FDA Staff*.

Food and Drug Administation (2015) *A Mobile Medical Applications. Guidance for Industry and Food and Drug Administration Staff*

Forsberg, K. and Mooz, H. (1991), The Relationship of System Engineering to the Project Cycle. INCOSE International Symposium, 1: 57–65

Fritzsche, M. and Keil, P. (2007). Agile Methods and CMMI: Compatibility or Conflict?. *e-Informatica*, 1(1), pp.9-26.

Gary, K., Enquobahrie, A., Ibanez, L., Cheng, P., Yaniv, Z., Cleary, K., Kokoori, S., Muffih, B. and Heidenreich, J. (2011). Agile methods for open source safety-critical software. *Softw: Pract. Exper.*, 41(9), pp.945-962.

Ge, X., Paige, R. and McDermid, J. (2010). An Iterative Approach for Development of Safety-Critical Software and Safety Arguments. In: *Proceedings of the 2010 Agile Conference*, IEEE Computer Society, pp.35-43.

Glazer, H., Dalton, J., Anderson, D., Konrad, M. and Shrum, S. (2008). *CMMI or Agile: Why Not Embrace Both!*. Software Engineering Institute.

Górski, J., Jarzębowicz, A., Leszczyna, R., Miler, J. and Olszewski, M. (2005). Trust case: justifying trust in an IT solution. *Reliability Engineering & System Safety*, 89(1), pp.33-47.

Górski, J. (2005). Trust Case—A Case for Trustworthiness of IT Infrastructures. *Cyberspace Security and Defense: Research Issues*, 196, pp.125-141.

Górski, J. (2007). Trust-IT - a framework for trust cas. In: *Proceedings of DSN 2007 : 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. pp.204-209.

Górski, J., Jarzębowicz, A. and Miler, J. (2008). Arguing trustworthiness of e-health services with the Trust-IT framework. In: *Proceedings of 25th Anniversary Healthcare Computing : Invitation to the Future : Conference & Exhibition (HC 2008)*.

Górski, J., Jarzębowicz, A. and Miler, J. (2012). Validation of Services Supporting Healthcare Standards Conformance. *Metrology and Measurement Systems*, 19(2), pp.269-282.

Górski, J., Jarzębowicz, A., Miler, J., Wardziński A. (2014) Challenges in providing support for evidence based argument management. *4th International Symposium on Model Based Safety Assessment IMBSA 2014*, October 27-29 2014, Munich, Germany

Górski J., Łukasiewicz K. (2012) Agile development of critical software, can it be justified?, 7th International Conference on Evaluation of Novel Approaches to Software Engineering, Wrocław, Springer

Górski J., Łukasiewicz K. (2012) Assessment of risks introduced to safety critical software by agile practices - a software engineer's perspective, Computer Science 13(3), AGH University of Science and Technology Press

Górski, J., Łukasiewicz K., (2013) Towards Agile Development of Critical Software", A. Gorbenko, A. Romanovsky, V. Kharchenko (Eds). Software Engineering for Resilient Systems - 5th International Workshop, SERENE 2013, 3-4 October, Kiev, Ukraine, 2013. Proceedings. LNCS 8166. Springer

Górski, J. and Łukasiewicz, K. (2017). Meeting Requirements Imposed by Secure Software Development Standards and Still Remaining Agile. *Lecture Notes in Computer Science*, pp.3-15.

Graydon, PJ & Kelly, TP (2013), Using argumentation to evaluate software assurance standards, vol 55, no. 9, pp. 1551-1562

Greenwell WS, Knight JC, Holloway CM, Pease J (2006). A taxonomy of fallacies in system safety argument. 24th International System Safety Conference, Albuequerque

Hanssen, G., Myklebust, T., & Stålhane, T. (2012). The application of Safe Scrum to IEC 61508 certifiable software.

Hanssen G. K., Stålhane T., Myklebust T. (2018) SafeScrum® – Agile Development of Safety-Critical Software. Springer, to be published in November 2018

Hawkins, R.; Kelly, T. P.; Knight, J. C. & Graydon, P. J. (2011), A New Approach to creating Clear Safety Arguments., *in* Chris Dale & Tom Anderson, ed., 'SSS' , Springer, , pp. 3-23

Hillson D. (2009). Managing risk in projects. Gower

iBeacon (2017), [online] Available at: https://developer.apple.com/ibeacon/ (Accessed: June 2017)

International Electrotechnical Commission (2010) *IEC 61508:2010 CMV*, Available at [online] http://www.iec.ch/functionalsafety/standards/ (Accessed: February 2018)

International Organization for Standardization (2006) *IEC 62304:2006 Medical device software -- Software life cycle processes*, [online] Available at: https://www.iso.org/standard/38421.html (Accessed: February 2018)

International Organization for Standardization (2007) *ISO 14971:2007 Medical devices -- Application of risk management to medical devices*, [online] Available at: https://www.iso.org/standard/38193.html (Accessed: March 2018)

International Organization for Standardization (2009) *ISO/TS 16949*, Available at [online] https://www.iso.org/standard/52844.html (Accessed: February 2018)

International Organization for Standardization (2015) *IEC 60601-1-11:2015*, Available at [online] https://www.iso.org/standard/65529.html (Accessed: March 2018)

International Organization for Standardization (2015) *IEC 62304:2006+AMD1:2015 CSV,* [online] Available at: https://www.iso.org/standard/64686.html (Accessed: March 2018)

International Organization for Standardization (2015) *ISO 9000 - Quality management, Available at [online] https://www.iso.org/iso-9001-quality-management.html* (Accessed: February 2018)

International Organization for Standardization (2016) *ISO 13485:2016 Medical devices -- Quality management systems,* Available at [online] https://www.iso.org/standard/59752.html (Accessed: February 2018)

International Organization for Standardization (2017). *About ISO - ISO*. [online] Available at: http://www.iso.org/iso/home/about.htm (Accessed: February 2018)

International Society for Pharmaceutical Engineering (2008) GAMP 5 Guide: Compliant GxP Computerized Systems, [online] Available at: https://ispe.org/publications/guidance-documents/gamp-5 (Accessed: March 2018)

Jones P, Górski J. Abstract of the Research Plan: Evidence-based arguments to support assurance and qualification of medical devices, [online] Available at: http://iag.pg.gda.pl/download/RCA_between_FDA_and_GUT-summary_page.pdf

Kelly, T.P.(1998). Arguing Safety – A Systematic Approach to Managing Safety Cases. University of York, Department of Computer Science

Knight, J. (2002). Safety critical systems: challenges and directions. In Proceedings of the 24th International Conference on Software Engineering (ICSE '02). ACM, New York, NY, USA, 547-550

Kromholz, Alfred H. and Ankrum, T. Scott (2005). Structured Assurance Cases: Three Common Standards. *Ninth IEEE International Symposium on High-Assurance Systems Engineering (HASE'05)*, pp. 99-108

Kruchten, P. (2011). Contextualizing agile software development. *Journal of Software: Evolution and Process*, 25(4), pp.351-361.

Leveson, N. (1995). Medical Devices: The Therac-25 Accidents. *Safeware: System Safety, and Computers* (Update of the 1993 IEEE Computer article ed.). Addison-Wesley

Lindvall M., Muthig D., Dagnino A., Wallin C., Stupperich M., Kiefer D., May J. & Kähkönen T. (2004). *Agile Software Development in Large Organizations* in Computer, 37(12), pp. 26-34

Łukasiewicz, K. (2017) Method of selecting programming practices for the safety critical software development projects – a case study. Technical report no. 02/2017, Gdańsk University of Technology

Łukasiewicz K., Górski J., (2018) Introducing agile practices into development processes of safety-critical software. In Proceedings of ASCS'18, Porto, Portugal, May 21-25, 2018

Marçal, A. C., de Freitas B. C., Furtado Soares F. S., Furtado M. S., Maciel T. M., Belchior A. D. (2008) *Blending Scrum practices and CMMI project management process areas*. Innovations in Systems and Software Engineering, 4(1), pp. 17-29

MAUDE (Manufacturer and User facility Device Experience) Database (2017), [online] Available at: https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfmaude/search.cfm

McHugh, M., McCaffery, F., Casey, V. and Pikkarainen, M. (2012). Integrating Agile Practices with a Medical Device Software Development Lifecycle. In: *Proceedings of European Systems and Software Process Improvement and Innovation Conference (EuroSPI)*

McHugh, M., McCaffery, F. and Coady, G. (2014). An Agile Implementation within a Medical Device Software Organisation. *Communications in Computer and Information Science*, pp.190-201.

Miszczyszyn M., Naliwajek J. (2016) *Documentation of the Group Project DocBeacon - iBeacon based clinic appointment & queue management system for iOS*

Musen, M.A. (2015) The Protégé project: A look back and a look forward. AI Matters. Association of Computing Machinery Specific Interest Group in Artificial Intelligence, 1(4), June 2015. DOI: 10.1145/2557001.25757003.

Mycklebust, T., Stålhane, T. and Hanssen, G. (2016). Use of Agile Practices when developing Safety-Critical software. In: *Proceedings of The 34th International System Safety Conference (ISSC)*.

NOR-STA project Portal (2017). [online] Available at: www.nor-sta.eu (Accessed: Deember 2017)

Norwegian University of Science and Technology (2017), [online] Available at: https://www.ntnu.edu/ (Accessed: March 2017)

Nyfjord, J. (2008). Towards integrating agile development and risk management (PhD dissertation). Kista.

Nyfjord, J., & Kajko-Mattsson, M. (2008). Integrating risk management with software development : state of practice. In: *Proceedings of The International MultiConference of Engineers and Computer Scientists* 2008 Vol. I

OpenUP (2012) [online] Available at: http://epf.eclipse.org/wikis/openup/ (Accessed: May 2016)

Paige R., Charalambous R., Ge X., Brooke P. (2008) *Towards Agile Engineering of High-Integrity Systems*. Proceedings of 27th International Conference on Computer Safety, Reliability and Security *(SAFECOMP)*, Newcastle upon Tyne, UK

Päivärinta, T. and Smolander, K. (2015). Theorizing about software development practices. *Science of Computer Programming*, 101, pp.124-135.

Palanque, P., Paternò, F. & Wright, P. (1998). Designing user interfaces for safety critical systems. ACM Sigchi Bulletin. 30. 200.

Pellet, [online] https://www.w3.org/2001/sw/wiki/Pellet, 2011 (Accessed: December 2017)

Petersen, K., & Wohlin, C. (2010). *The effect of moving from a plan-driven to an incremental software development approach with agile practices*. Empirical Software Engineering, 15(6), pp. 654-693

Pikkarainen M., Mantyniemi, A. (2006) *An Approach For Using CMMI in Agile Software Development Assessments: Experiences From Three Case Studies*. Proceedings of SPICE Conference, Luxembourg

Poppendieck M. and T. (2003) *Lean software development: an agile toolkit*, Addison-Wesley

Potter, N., Sakry M. (2009). Implementing Scrum (Agile) and CMMI together. Process Group Post Newsletter, 16(2), Available at: http://www.itmpi.org/assets/base/images/itmpi/Potter-ScrumCMMI.pdf

Pressman R. (2009) *Software Engineering: A Practitioner's Approach* (7 ed.). McGraw-Hill, Inc., New York, NY, USA.

Rasmussen, R., Hughes, T., Jenks, J. R., & Skach, J. (2009) *Adopting Agile in an FDA Regulated Environment.* Agile Conference Proceedings, Chicago, USA, 24-28 August 2009. IEEE Computer Society, Los Alamitos, pp. 151-155

Rottier A., Rodrigues V. (2008) *Agile Development in a Medical Device Company*, IEEE, Agile Conference

Royce, W. W. (1970) *Managing the development of large software systems*, Proceedings of IEEE Wescon, pp. 382-338.

Schwaber, K., Beedle (2001) Agile Software Development with Scrum. Prentice Hall

Sentez K., Ferson S., (2002). *Combination of evidence in Dempster-Shafer theory*. SANDIA National Laboratories.

Siddique L., Hussein B. A. (2016) *Managing risks in Norwegian Agile Software Projects: Project Managers´ perspective*, International Journal of Engineering Trends and Technology (IJETT), V41(2),56-65

SINTEF (2017), [online] Available at: http://www.sintef.no/ (Accessed: December 2017)

Spivey, J. M. (1992). The Z Notation: A reference manual. International Series in Computer Science (2nd ed.). Prentice Hall.

Standish Group (2015) Chaos Report

Stålhane, T., Myklebust, T. and Hanssen, G. (2013). *Safety standards and Scrum – A synopsis of three standards*. [article] Available at: http://www.sintef.no/globalassets/safety-standards-and-scrum_may2013.pdf.

Stephenson Z., McDermid J., (2006). Ward A. *Health Modelling for Agility in Safety-Critical Systems Development*. Proceedings of the First IET International Conference on System Safety Engineering, London, UK

Stålhane, T., & Malm, T. (2016). Risk assessment: Experts vs. lay people. In L. Walls, M. Revie, & T. Bedford (Eds.), *Risk, Reliability and Safety: Innovating Theory and Practice* (pp. 1345-1352). CRC Press

SWRL: A Semantic Web Rule Language Combining OWL and RuleML,(2004) [online] https://www.w3.org/Submission/SWRL/, (Accessed: July 2017)

Trapp, M., Schneider, D. and Liggesmeyer, P. (2013). A Safety Roadmap to Cyber-Physical Systems. *Perspectives on the Future of Software Engineering*, pp.81-94.
Toulmin S. E., The Uses of Argument (Updated Edition) (2003), Cambridge University Press

Van Solingen, R.; Egon Berghout (1999).The Goal/Question/Metric Method. McGraw-Hill Education

VersionOne (2016), 10th Annual State of Agile Report, [online] http://www.agile247.pl/wp-content/uploads/2016/04/VersionOne-10th-Annual-State-of-Agile-Report.pdf

Weiguo L., Xiaomin F. (2009). *Software Development Practice for FDA-Compliant Medical Devices.* Proceedings of the 2009 International Joint Conference on Computational Sciences and Optimization, Sanya, China

Weinstock C. and John B. Goodenough. (2009) Towards an Assurance Case Practice for Medical Devices. *TECHNICAL NOTE Software Engineering Institute* October. Available at: *http://www.sei.cmu.edu/reports/09tn018.pdf*

Weinstock, C., Goodenough, J. and Klein, A. (2013). *2013 1st International Workshop on Assurance Cases for Software-Intensive Systems (ASSURE 2013)*. 1st ed. Piscataway, NJ: IEEE, pp.7-11.

Zhang, Y., Jones, P. L., & Klonoff, D. C..(2010). Second Insulin Pump Safety Meeting: Summary Report. Journal of Diabetes Science and Technology, 4(2), pp 488–493

APPENDIX A: METODA DOBORU PRAKTYK PROGRAMISTYCZNYCH W WYTWARZANIU OPROGRAMOWANIA ZWIĄZANEGO Z BEZPIECZEŃSTWEM – ROZSZERZONE STRESZCZENIE

A.1 Wprowadzenie

Nadrzędnym celem procesów wytwórczych w projekcie informatycznym jest dostarczenie oprogramowania wysokiej jakości, które zadowoliłoby klienta i przyniosłoby satysfakcjonujące dochody. Na sukces projektu wpływa wiele czynników, a jednym z kluczowych wydaje się być właściwy wybór metody wytwarzania oprogramowania.

Metody wytwarzania oprogramowania oferują wytyczne dotyczące stosowania określonych działań i aktywności oraz sposobu organizowania ich w jednolity proces wytwórczy. Obecnie dostępnych jest wiele metod, od zdyscyplinowanych, sterowanych planem i zorientowanych na proces po bardziej elastyczne, zorientowane na ludzi, zwinne podejścia.

Metody sterowane planem, jak sama nazwa wskazuje, koncentrują się na aspektach planowania procesów wytwórczych. Przykłady takich metodyk to m.in. model kaskadowy i SW-CMM (Software Engineering Institute, 1993). Na drugim końcu spektrum znajdują się zwinne metodyki. W Manifeście Agile (Agile Manifesto, 2001), który opisuje zasady stojące za tą grupą metodyk, podkreślono, że to działające oprogramowanie, współpraca i elastyczność w reagowaniu na zmiany są cenione bardziej niż dokumentacja, procesy i plany. Scrum (Schwaber, Beedle, 2001) i eXtreme Programming (eXtreme Programming, 1999) są przykładami takich metodyk.

Pomiędzy tymi dwiema skrajnościami można zidentyfikować metody hybrydowe. Traktują one wszystkie metodyki jako zbiór praktyk, działań i wytycznych, wybierając elementy potencjalnie korzystne, zarówno wśród praktyk podejścia zdyscyplinowanego, jak i zwinnego, łącząc je następnie w innowacyjny sposób.

Powszechnie przyjmuje się, że nie istnieje jedna, idealna metodyka pasująca do wszystkich rodzajów projektów informatycznych (Brooks, 1987). Zarówno elastyczne, jak i oparte na planach podejścia mają swoje mocne i słabe strony, w zależności od dziedziny i cech danego projektu.

Dziedzina, dla której dedykowany będzie dany system, jest ważnym czynnikiem wpływającym na decyzję o wyborze metody wytwarzania oprogramowania dla danego projektu. Niektóre dziedziny, takie jak dziedziny systemów o wymaganiach krytycznych względem bezpieczeństwa, są obarczone tak specyficznymi ograniczeniami i wymaganiami, że mają kluczowy wpływ na to, z jakich metod wytwarzania

oprogramowania można korzystać. Systemy o wymaganiach krytycznych względem bezpieczeństwa to systemy, "których awaria może zagrozić życiu ludzkiemu, doprowadzić do znaczącej straty finansowej lub spowodować znaczne szkody w środowisku" (Knight, 2002). Takie systemy można znaleźć w transporcie, ochronie zdrowia, przemyśle energetycznym i wielu innych dziedzinach, w których pewne kluczowe zadania są przenoszone na rozwiązania technologiczne. Potencjalne szkody w przypadku nieprawidłowego działania systemu mogą mieć znaczący wpływ na jego środowisko i otoczenie. Z tego powodu systemy o wymaganiach krytycznych względem bezpieczeństwa podlegają wielu przepisom i normom, które określają w jaki sposób należy zadbać o to, aby ostateczne oprogramowanie było bezpieczne. Niektóre standardy regulują również proces rozwoju i definiują jego niezbędne działania i artefakty.

W ostatnich latach rosnąca konkurencja na rynku IT miała również wpływ na systemy związane z bezpieczeństwem. Dzięki powszechnemu stosowaniu elementów oprogramowania, na przykład w samochodach lub samolotach, każdy jest dziś potencjalnym użytkownikiem oprogramowania o znaczeniu krytycznym dla bezpieczeństwa, co naturalnie zmieniło perspektywę firm oferujących takie oprogramowanie. Na przykład, jeśli chodzi o firmy produkujące oprogramowanie medyczne, przeszły one od wspierania jedynie szpitali i lekarzy do dostarczania usług w zakresie spersonalizowanych rozwiązań w dziedzinie e-zdrowia bezpośrednio dla pacjentów. Kluczowe stało się oferowanie lepszego oprogramowania, bardziej atrakcyjnego dla użytkownika, przy utrzymaniu kosztów tak niskich, jak to tylko możliwe, w celu konkurowania na tym szybko zmieniającym się i dynamicznie rosnącym rynku. W związku z tym istnieje duże zapotrzebowanie na zwiększenie wydajności procesów wytwarzania oprogramowania (pod względem nakładu pracy i czasu), przy jednoczesnym przestrzeganiu wymogów bezpieczeństwa narzuconych przez odpowiednie normy i przepisy.

A.2 Cel i teza rozprawy

Celem badań opisanych w niniejszej pracy było opracowanie podejścia mającego na celu ułatwienie wprowadzenia bardziej elastycznego podejścia do procesu wytwarzania oprogramowania, zależnego od cech projektu, przy zachowaniu zgodności z wymaganymi standardami i przepisami bezpieczeństwa. Przedstawiona w tej pracy metoda AgileSafe jest głównym rezultatem tych badań.

Teza pracy została sformułowana następująco:

Zaproponowana metoda umożliwia istotne obniżenie nakładów na wytwarzanie oprogramowania w projektach związanych z bezpieczeństwem, bez naruszania wymagań wynikających z norm i standardów dotyczących zapewniania bezpieczeństwa.

A.3 Zastosowane podejście badawcze

Badania przedstawione w tej rozprawie obejmowały następujące kroki:

Krok I. Analiza zalecanych i obecnie stosowanych (rygorystycznych) praktyk tworzenia oprogramowania dla systemów o wymaganiach krytycznych względem bezpieczeństwa.

Krok II. Analiza zalecanych i obecnie stosowanych zwinnych praktyk w zakresie wytwarzania oprogramowania i ich wpływu na wydajność procesów wytwórczych

Krok III. Analiza ograniczeń wynikających z obecnych regulacji i standardów związanych z domeną oprogramowania o wymaganiach krytycznych względem bezpieczeństwa (ze szczególnym uwzględnieniem urządzeń medycznych)

Krok IV. Opracowanie nowego, hybrydowego podejścia - metody AgileSafe - która jest głównym wynikiem tych badań.

Krok V. Identyfikacja drogą eksperymentu, tych zwinnych praktyk, które mogłyby być włączone w zakres proponowanej metody.

Krok VI. Opracowanie i integracja narzędzi wspierających AgileSafe w celu zademonstrowania metody.

Krok VII. Wybór kryteriów oceny proponowanej metody.

Krok VIII. Ocena AgileSafe - według studiów przypadków i ankiet z udziałem ekspertów.

A.4 Dziedzina problemowa

W celu zapewnienia bezpieczeństwa działania systemu w jego docelowym środowisku, troska o jego bezpieczeństwo powinna zacząć się już w procesie wytwarzania. System powinien być analizowany pod kątem potencjalnego ryzyka, które rozumiane jest jako "możliwość utraty, uszkodzenia lub niekorzystnej sytuacji" w zakresie wytwarzania oprogramowania, jak i eksploatacji w jego docelowym otoczeniu (Miler i Górski, 2001).

Niemniej jednak, obecnie szacuje się, że błędy oprogramowania są źródłem około 25% zdarzeń niepożądanych (Jones, Górski, 2017) zarejestrowanych w amerykańskiej bazie danych Manufacturer and User facility Device Experience (MAUDE,

2017). Warto wspomnieć także o znanych wypadkach, w wyniku których wielu ludzi straciło życie lub odniosło ciężki uszczerbek na zdrowiu właśnie w wyniku wadliwego działania oprogramowania, takich jak awaria systemu ostrzegania w Airbus A340-642 (AAIB, 2005), błędne działanie akcelerometru w Boeing 777-200 (ATSB, 2005) lub tragiczne w skutkach błędy obliczeniowe przy radioterapii maszyną Therac-25 (Leveson, 1995).

Rynek oprogramowania krytycznego dla bezpieczeństwa najprawdopodobniej będzie rozwijał się w przyszłości, wraz ze spadkiem kosztów sprzętu i rozwojem nowych możliwości oferowanych przez oprogramowanie (Knight, 2002). W związku z tym, znaczenie rozwiązań zapewniających bezpieczeństwo, będzie rosło, zwłaszcza że czas wydania produktu oraz koszty odgrywają coraz ważniejszą rolę w tej dziedzinie.

Podczas gdy metody sterowane planem dowiodły swojej wartości i przydatności w projektach o kluczowym znaczeniu dla bezpieczeństwa, przy rozwijającym się rynku oprogramowania ostatnich kilku lat, podejście to jest wystawiane na próbę (Ge, Paige, McDermid, 2010). Rosnąca konkurencja, stale zmieniające się technologie i bardziej zróżnicowane grupy klientów zmieniły oczekiwania wobec metod wytwarzania oprogramowania. Potrzeba dostarczenia systemów o odpowiedniej jakości, szybciej i przy niższych kosztach w porównaniu do konkurentów sprawiły, że zaczęto poszukiwać alternatywnych rozwiązań (Petersen, Wohlin, 2010).

W odpowiedzi na te potrzeby, metodyki zwinne oferują praktyki ceniące bliską relację z klientami, pozwalające na bardziej swobodne podejście do dokumentacji i zapewniające elastyczny cykl życia w oparciu o krótkie iteracje (Abrahamsson et al., 2002). Pomyślnie wdrożone, zwinne praktyki mogą potencjalnie obniżyć koszty produkcji, jak również czas wprowadzenia produktu na rynek (Drobka, Noftz, Raghu, 2004; Lindvall et al., 2004)

Według raportu CHAOS (Standish Group, 2015) z lat 2011-2015, projekty, w których zastosowano metody zwinne, w 39% zakończyły się powodzeniem. Dla porównania, tylko 11% projektów realizowanych zgodnie z podejściem kaskadowym można było uznać za ukończone z sukcesem. Co więcej, tylko 9% zwinnych projektów zakończyło się niepowodzeniem, podczas gdy projekty kaskadowe zawiodły w 29% przypadków.

VersionOne w swoim 10 raporcie State of Agile przedstawia zalety zwinności, na podstawie 3 880 zebranych odpowiedzi od swoich respondentów (VersionOne, 2016). W odniesieniu do naszych badań, najciekawsze były wyniki dotyczące redukcji nakładu pracy: 85% respondentów zauważyło wzrost produktywności zespołu, 80% krótszy czas wejścia na rynek i 79% poprawę jakości oprogramowania, gdy wprowadzili zwinność w swoich projektach.

Wobec takiego potencjału, jakie prezentują metodyki zwinne, zaczęły pojawiać się metody proponujące integrację praktyk metodyk zwinnych do procesów wytwórczych oprogramowania o wymaganiach krytycznych względem bezpieczeństwa. W 2009 roku Weiguo i Xiaomin (Weiguo, Xiaomin, 2009) zaprezentowali podejście oparte na zwinności, odpowiednie dla projektów urządzeń medycznych zgodnych z FDA. Podejście zostało ograniczone do konkretnej dziedziny, dlatego trudno je uznać za uniwersalne rozwiązanie. Innym interesującym podejściem jest model AV (McHugh, McCaffery and Coady, 2014), łączący tradycyjny model V ze Scrum i koncentrujący się na oprogramowaniu medycznym oraz na standardzie IEC 62304. Podczas, gdy model AV przedstawia obiecujące rozwiązanie, jego potencjalne zastosowania są ograniczone i jako takie nie mogą być powszechnie zalecane.

Bardziej ogólne i praktyczne rozwiązanie zaproponowała grupa badawcza z SINTEF (SINTEF, 2017) oraz Norweskiego Uniwersytetu Nauki i Technologii (Norwegian University of Science and Technology, 2017). Zaproponowali oni metodę SafeScrum (Mycklebust, Stålhane and Hanssen, 2016), która koncentruje się na dostosowaniu Scruma do rozwoju oprogramowania o znaczeniu krytycznym dla bezpieczeństwa. Metoda ta zapewnia dobrze zbadany zestaw praktyk, chociaż aspekt dowodzenia zgodności z wymaganiami dotyczącymi bezpieczeństwa jest wciąż w trakcie rozwoju.

Więcej na temat powiązanych prac i badań można przeczytać w rozdziale 3 niniejszej pracy.

Chociaż omówione modele adaptowania zwinnych praktyk do projektów krytycznych z punktu widzenia bezpieczeństwa są cennymi źródłami wiedzy, nadal istnieje potrzeba opracowania łatwiejszego w użyciu i dokładnego zestawu wytycznych dla projektów o krytycznych wymaganiach względem bezpieczeństwa, które chciałyby dostosować zwinne metodyki do swoich potrzeb. Metoda AgileSafe, będąca wynikiem tych badań doktoranckich, jest próbą dostarczenia takiego rozwiązania, bardziej uniwersalnego niż analizowane podejścia i jednocześnie umożliwiającego zapewnienie bezpieczeństwa i efektywności procesu.

A.5 Metoda AgileSafe

Większość działań związanych z metodą AgileSafe odbywa się na etapie planowania projektu. Dodatkowe obciążenie pracą na etapie wytwarzania jest ograniczone do minimum i koncentruje się na wymaganiach standardów, a także praktykach wprowadzonych do projektu. Większość elementów metody, po jej

przygotowaniu, może być ponownie wykorzystana lub dostosowana później do innych projektów.

Istnieją dwa główne przypadki użycia AgileSafe. Pierwszym i podstawowym jest *Zastosowanie AgileSafe,* czyli uzyskanie porady na temat procesu tworzenia oprogramowania, z sugestiami, które praktyki zastosować i jak zapewnić zgodność z wybranymi standardami. Drugim sposobem jest *Udoskonalenie metody* poprzez aktualizację wiedzy przechowywanej w metodzie, dostarczając informacji zwrotnych i danych o nowych praktykach.

Schemat użycia AgileSafe na wysokim poziomie przedstawiono na Rysunku 1 w rozdziale 4.1 niniejszej pracy.

A.5.1 Zastosowanie AgileSafe

Informacje o projekcie oraz o kontekście regulacyjnym ograniczającym projekt i jego produkt są danymi wejściowymi do metody. Użytkownik musi określić charakterystykę projektu i regulacje, którym projekt podlega. Na ich podstawie użytkownik prowadzony jest przez dwa główne procesy AgileSafe: proces, który sugeruje praktyki programistyczne, które mogą być zastosowane w projekcie, oraz proces, w którego efekcie powstaje zestaw argumentów wiarygodności odpowiadający regulacjom zawartym w ograniczeniach wejściowych.

Pierwszy krok przypadku *Zastosowanie AgileSafe* to analiza cech charakteryzujących projekt. W tym procesie użytkownik powinien zebrać informacje o projekcie, które są następnie wykorzystywane jako dane wejściowe do kolejnych kroków AgileSafe. W celu określenia charakterystyki projektu, AgileSafe stosuje podejście do skalowania podejścia zwinnego, przedstawione przez Scotta W. Amblera (Ambler, 2010). Jak zauważył Kruchten w (Kruchten, 2011), kontekst jest kluczowy w decydowaniu o tym, jak zwinny może być proces wytwarzania oprogramowania i zarządzanie nim w danym projekcie. Skalowalne czynniki Amblera reprezentują szerokie spektrum okoliczności, zarówno związanych z firmą, jak i projektem.

Czynniki opisane przez Amblera zostały rozbudowane o skale ocen na potrzeby metody AgileSafe. Każdy czynnik może być oceniony w 5-punktowej skali.

1. Rozmiar zespołu (na podstawie badania Amblera (Ambler, 2012))

(Liczba programistów pracujących w projekcie)

A - Mniej niż 10 programistów; B - Od 10 do 50 programistów; C - Od 50 do 100 programistów; D - 100 programistów; E-1000 programistów

2. Dystrybucja geograficzna zespołu (na podstawie ankiety Amblera (Ambler, 2012))
(Gdzie są fizycznie zlokalizowani członkowie zespołu?)
A – To samo pomieszczenie; B - Ten sam budynek; C - W odległości dojazdu samochodem; D - Niektórzy pracują w domu; E - Globalnie rozdystrybuowani
3. Złożoność dziedziny produktu
(Jak skomplikowana jest docelowa domena produktu?)
A - Prosta; B - Przewidywalna; C - Szybko się zmienia; D - Skomplikowana; E - Skomplikowana / rozwijana
4. Dystrybucja organizacyjna
(Jaka jest przynależność osób pracujących w projekcie, jak zorganizowana jest praca?)
A – Wspólna praca; B - Różne zespoły; C - Różne działy; D - Różne firmy partnerskie; E - Kontraktowi
5. Złożoność techniczna
(Jak skomplikowana jest strona technologiczna projektu?)
A - Homogeniczny; B - Wiele technologii; C - Nowa technologia; D - Rozwiązania systemowe / wbudowane; E - Heterogeniczny / starszy
6. Złożoność organizacyjna
(Jakie są struktury firmy, w jaki sposób są zarządzane?)
A - Elastyczna, intuicyjna; B - Elastyczna, uporządkowana; C - Stabilna, ewolucyjna; D - Stabilna, zaplanowana; E - Sztywna
7. Dyscyplina przedsiębiorstwa
(Co leży w centrum uwagi kierownictwa firmy?)
A – Skoncentrowana na projekcie; B - Głównie skoncentrowana na projekcie; C - Zrównoważona; D - Głównie skoncentrowana na przedsiębiorstwie; E - Skoncentrowana na przedsiębiorstwie

W AgileSafe wbudowana została baza wiedzy, która zapewnia dopasowanie odpowiednich zwinnych praktyk do charakterystyki projektu. Algorytmy, które sugerują praktyki dla użytkownika są zaimplementowane w bazie wiedzy w postaci reguł SWRL (SWRL, 2004). Dlatego też, charakterystyka projektu musi zostać dodana do bazy wiedzy, w celu uzyskania sugestii dotyczących praktyk odpowiednich dla tego projektu.

Aby zapewnić, że wymogi bezpieczeństwa określone w odpowiednich regulacjach zostały wbudowane w projekt, AgileSafe wykorzystuje argumenty wiarygodności. Główną ideą jest zapewnienie argumentów wiarygodności dla procesu wytwarzania oprogramowania, jak również dla samego produktu końcowego. Podczas gdy ten ostatni służy do wykazania zgodności produktu z daną normą lub standardem, pierwszy ma na celu wykazanie, że wybrane praktyki wytwarzania oprogramowania zapewniają wystarczający poziom bezpieczeństwa dla powstałego produktu. Dzięki

temu połączonemu podejściu użytkownik może zapewnić, że wybrane praktyki są odpowiednie dla konkretnego projektu, z jego wymaganiami bezpieczeństwa nałożonymi przez wymagane normy i standardy.

Wzorce argumentów wiarygodności bazują na odpowiednich normach, standardach i wytycznych. Oparte są o podejście Trust-IT (NOR-STA, 2017), (Cyra, Górski, 2011), (Górski, Jarzębowicz, Miler, 2012). Aby ułatwić korzystanie z AgileSafe, w metodzie zostało wykorzystane narzędzie NOR-STA Argevide (Argevide, 2017), służące do zarządzania zestawem argumentów AgileSafe.

Wszystkie argumenty w metodzie opracowywane są osobno dla każdego obowiązującego standardu w celu obsługi oddzielnych procesów certyfikacji i są oparte na strukturze danego standardu Istnieją trzy typy argumentów wiarygodności w AgileSafe: Argumenty zgodności praktyk, Argument zgodności praktyk projektu i Argument zgodności projektu. Pierwsze dwa koncentrują się na praktykach wytwarzania oprogramowania, które są w stanie wytworzyć niezbędny materiał zgodności, zaś ostatni przedstawia argumentację opartą na faktycznie zebranych artefaktach projektu. Rysunek przedstawiający relację między argumentami, wraz z rozszerzonym opisem można znaleźć w rozdziale 7 niniejszej pracy.

A.5.2 Udoskonalenie AgileSafe

W celu dalszego ulepszenia metody i dokładniejszego dopasowania jej do potrzeb użytkownika, wiedza przechowywana w metodzie powinna być regularnie weryfikowana i aktualizowana.

Użytkownik może wprowadzić nowe praktyki programistyczne do puli praktyk, z których wybierane są sugerowane praktyki. W tym celu użytkownik powinien dodać nową praktykę lub uaktualnić już istniejącą w bazie wiedzy metody. Można je również dodać do argumentów wiarygodności AgileSafe.

Użytkownik może także uaktualnić listę standardów wspieranych przez argumenty wiarygodności.

Więcej szczegółowych informacji na temat przypadków użycia oraz procesów i artefaktów metody AgileSafe można znaleźć w rozdziale 4 niniejszej pracy.

A.6 Walidacja metody

W celu oceny AgilSafe zastosowano metodę Cel-Pytanie-Metryka (Goal-Question-Metric) (Van Solingen, Berghout, 1999).

Na podstawie tezy tej rozprawy doktorskiej zadeklarowano dwa cele:

(G1) Analiza proponowanej metody pod kątem wspierania wprowadzenia zwinnych praktyk do procesu wytwarzania oprogramowania, zachowując jednocześnie zgodność z wymaganiami dotyczącymi bezpieczeństwa narzuconymi przez dziedzinę projektu.

(G2) Analiza proponowanej metody pod kątem wpływu na zmniejszenie nakładów poświęconych na proces wytwarzania oprogramowania w projektach związanych z bezpieczeństwem.

W odniesieniu do tych celów określono następujące pytania badawcze:

(Q1) Jaki jest potencjalny wpływ wprowadzenia zwinnych praktyk na proces wytwarzania oprogramowania o krytycznym znaczeniu dla bezpieczeństwa?

(Q2) Czy metoda AgileSafe może zmniejszyć negatywny wpływ zwinnych praktyk na proces wytwarzania oprogramowania krytycznego dla bezpieczeństwa?

(Q3) Czy metoda sugeruje praktyki, które mają odniesienie do projektu i jego otoczenia?

(Q4) Czy metoda ta wspiera aspekt zapewniania bezpieczeństwa projektu podczas wprowadzania nowych praktyk?

(Q5) Czy wprowadzenie zwinnych praktyk do procesu wytwarzania oprogramowania przynosi potencjalnie korzyść w postaci zmniejszenia nakładów związanych z projektem?

(Q6) Czy metoda AgileSafe pozwala na wprowadzenie zwinnych praktyk, które mogą potencjalnie zmniejszyć nakłady potrzebne do projektu?

Aby znaleźć odpowiedzi na te pytania, zebrano następujące metryki:

(M1) Lista korzyści i problemów związanych z wprowadzeniem zwinnych praktyk - w oparciu o dane z literatury dotyczące stosowania zwinnych praktyk w rozwoju oprogramowania krytycznego dla bezpieczeństwa, zebranego w rozdziałach 2 i 3 niniejszej pracy.

(M2) Ocena ryzyka dotycząca wprowadzenia praktyk Scrum i eXtreme Programming do wytwarzania oprogramowania krytycznego dla bezpieczeństwa - w oparciu o dane dotyczące postrzeganego wpływu praktyk zwinnych na procesy wytwórcze oprogramowania krytycznego dla bezpieczeństwa, zebrane w rozdziale 8.2 niniejszej pracy.

(M3) Lista dodatkowych praktyk potrzebnych do ograniczenia ryzyka przy wprowadzaniu praktyk Scrum i eXtreme Programming - w oparciu o dane dotyczące postrzeganego wpływu praktyk zwinnych na procesy wytwórcze oprogramowania krytycznego dla bezpieczeństwa, zebrane w rozdziale 8.2 niniejszej pracy.

(M4) Pokrycie argumentów wiarygodności AgileSafe dla standardu ISO 14971 przez praktyki sugerowane w studium przypadku - w oparciu o dane dotyczące stosowalności głównych elementów metody AgileSafe w projektach związanych z bezpieczeństwem, w kontrolowanym i ograniczonym środowisku, zebrane w rozdziale 8.3 niniejszej pracy.

(M5) Ocena szablonu dodawania nowej praktyki przez ekspertów SafeScrum - w oparciu o teoretyczne oceny AgileSafe przez ekspertów, zebrane w rozdziale 8.5 niniejszej pracy

(M6) Lista zwinnych praktyk stosowanych obecnie w dwóch branżowych projektach o krytycznym znaczeniu dla bezpieczeństwa - w oparciu o teoretyczną ocenę AgileSafe przeprowadzoną przez ekspertów, opisaną w rozdziale 8.5 niniejszej pracy

(M7) Lista najbardziej wartościowych aspektów metody AgileSafe - na podstawie ocen AgileSafe przez ekspertów, zgromadzonych w rozdziale 8.5 niniejszej pracy

(M8) Lista najbardziej kłopotliwych aspektów metody AgileSafe - w oparciu o oceny AgileSafe przez ekspertów, zebranych w rozdziale 8.5 niniejszej pracy

(M9) Związek między liczbą sugerowanych praktyk (AS.A.13) a liczbą praktyk projektowych (AS.A.6) - na podstawie danych zebranych w rozdziale 8.4 niniejszej pracy

(M10) Pokrycie argumentów wiarygodności AgileSafe dla standardu ISO 14971 przez zestaw praktyk dla projektu GlucoMet - na podstawie danych zebranych w rozdziale 8.4 niniejszej pracy

Cel G.1 został osiągnięty poprzez udzielenie odpowiedzi na pytania Q1, Q2, Q3 i Q4. Na ich podstawie możemy wywnioskować, że AgileSafe wspiera wprowadzanie zwinnych praktyk w proces tworzenia oprogramowania, zachowując jednocześnie zgodność z wymaganiami dotyczącymi zapewnienia oprogramowania.

Cel G.2 został osiągnięty poprzez odpowiedź na pytania Q5 i Q6. Na tej podstawie możemy stwierdzić, że zwinne praktyki mogą zredukować nakłady potrzebne do projektu, a AgileSafe, wspierając wprowadzenie takich praktyk, umożliwia zmniejszenie nakładów poświęconych procesom wytwarzania oprogramowania w projektach związanych z bezpieczeństwem.

Stanowi to podstawę do przekonania, że teza tej pracy:

Zaproponowana metoda umożliwia istotne obniżenie nakładów na wytwarzanie oprogramowania w projektach związanych z bezpieczeństwem, bez naruszania wymagań wynikających z norm i standardów dotyczących zapewniania bezpieczeństwa.

została udowodniona.

A.7 Wkład rozprawy w rozwój dziedziny

Głównym osiągnięciem badań związanych z niniejszą pracą doktorską jest opracowanie metody do wyboru praktyk programistycznych i monitorowania zgodności, nazwanej AgileSafe. Główne innowacyjne elementy AgileSafe to:

1. Framework wspomagający dobór praktyk programistycznych w projektach o wymaganiach krytycznych względem bezpieczeństwa, od definicji projektu po fazę końcową i ocenę wiarygodności.

2. Zastosowanie koncepcji argumentacji opartej na dowodach do reprezentowania wymagań bezpieczeństwa narzuconych przez odpowiednie normy i standardy oraz wprowadzenie trzypoziomowej struktury argumentów wiarygodności w celu wsparcia osiągania i monitorowania zgodności (Argument zgodności praktyk, Argument zgodności praktyk projektu, Argument zgodności projektu i ich wzorce).

3. Specyfikacja bazy wiedzy i związanych z nią algorytmów wnioskowania, wspierających dobór praktyk dla danego projektu

Aby zweryfikować proponowaną metodę AgileSafe, w trakcie badań przeprowadzono trzy studia przypadku:

Case Study A

Głównym celem tego studium przypadku było sporządzenie listy kontrolnej zagrożeń, a także oceny ryzyka dla włączenia zwinnych praktyk w procesy wytwórcze projektów o krytycznym znaczeniu dla bezpieczeństwa, w oparciu o analizę ryzyka dostarczoną przez uczestników. Ponadto, sugestie uczestników dotyczące dodatkowych praktyk ograniczających potencjalne ryzyko zostały zbadane i uznane za cenne podczas przygotowywania metody AgileSafe.

Case Study B

Celem tego studium było wykorzystanie AgileSafe do włączenia wybranych praktyk zarządzania ryzykiem do zwinnego projekt z wymaganiami dotyczącymi bezpieczeństwa. Projekt był prowadzony przez grupę studentów informatyki i trwał 2 semestry, od lutego 2015 do stycznia 2016.

Projekt skupiał się na rozwoju aplikacji mobilnej związanej ze zdrowiem. Szczególnie interesujące było to, w jaki sposób mogłoby wyglądać wdrożenie wytycznych FDA, które zostały niedawno wydane (Food and Drug Administration, 2015).

Zespół projektu współpracował ze studiem programistycznym Bright Inventions (Bright Inventions, 2017), które zaoferowało wsparcie techniczne.

Zadanie studentów polegało na zaimplementowaniu systemu zarządzania wizytami i kolejkami w przychodni, przy użyciu iBeacon (iBeacon, 2017) na iOS.

Case Study C

Celem tego studium przypadku była ocena procesów metody AgileSafe, od Analizy projektu do Przygotowania Argumentu zgodności projektu.

To studium przypadku zostało przeprowadzone dla fikcyjnego projektu o nazwie GlucoMet. Oparto go na projekcie przedstawionym uczniom podczas studenckiego studium przypadku (rozdział 8.2) oraz uwag przedstawionych w (Chen i in., 2014) oraz (Zhang, Jones i Klonoff, 2010). Projekt GlucoMet dotyczył oprogramowania dla pomp insulinowych z funkcją monitorowania glukozy.

Szczegóły dotyczące wspomnianych studium przypadku zostały przedstawione w rozdziale 8 niniejszej pracy. Ich wyniki zostały częściowo opublikowane w (Górski i Łukasiewicz, 2012a, 2012b, 2013, Łukasiewicz, 2017, Łukasiewicz i Górski, 2018).

W trakcie studium przypadku Case Study B odkryto nowe praktyki Agile związane z zarządzaniem ryzykiem, jedna z nich został już wprowadzona do zestawu zalecanych praktyk metodyki SafeScrum (Hanssen, Stålhane, Myklebust, 2018).

Badania i wynikająca z nich metoda AgileSafe już wzbudziły duże zainteresowanie w branży. W rezultacie uzyskano dotację z Polsko-Norweskiego Programu Badawczego na stosunki dwustronne i stałą współpracę z zespołem badawczym w SINTEF Trondheim. Autorka tej pracy została poproszona o dołączenie do komitetu programowego warsztatów Agile Development of Safety-Critical Software (ASCS) 2018, które odbyły się w ramach konferencji Agile Alliance XP 2018. Autorka była także członkiem komitetu programowego Międzynarodowej Konferencji na temat Lean and Agile Software Development w 2017 i 2018 roku.