How does Scrum Conform to the Regulatory Requirements Defined in MDevSPICE®?

Özden Özcan-Top[1], Fergal McCaffery[1,2]

[1]Regulated Software Research Centre & Lero
Dundalk Institute of Technology, Dundalk, Ireland
[2]STATSports Group, Dundalk, Ireland
ozden.ozcantop@dkit.ie, fergal.mccaffery@dkit.ie

Abstract. Medical device software development is subject to high regulations due to the potential risk of harming patients with unsafe medical devices. These regulations require software development to be performed with high discipline and evidence to be provided for auditory purposes. It's not easy to manage both conformance to regulations and efficiency in medical device development. Therefore, there is a transition towards agility in safety critical systems development, to build high quality systems, shorten time to market, improve customer and employee satisfaction and ensure both safety and reliability. In this study, we evaluated one of the most highly adopted agile software development methods, Scrum from a regulatory perspective. We investigated to what extend the regulatory requirements defined in MDevSPICE® are met with implementation of the Scrum method and what additional processes and practices have to be performed to ensure safety and regulatory compliance in the healthcare domain.

Keywords. MDevSPICE®, Scrum, Regulatory Compliance, Safety Critical Domain, Agile Software Development.

1 Introduction

The safety critical nature of medical device software requires Medical Device (MD) regulations are in place to ensure the safety of these devices. Manufacturers have to comply with the requirements to market an MD within a particular region. International standardizing bodies and regional regulatory authorities issue these requirements as standards or guidance. In the US, the Food and Drug Administration (FDA) issues the regulation through a series of official channels, including the Code of Federal Regulation (CFR) Title 21, Chapter I, Subchapter H, Part 820 [1]. In the EU, the corresponding regulation is outlined in the general Medical Device Directive (MDD) 93/42/EEC [2], the Active Implantable Medical Device Directive (AIMDD) 90/385/EEC [3], and the In-vitro Diagnostic (IVD) Medical Device Directive 98/79/EC [4] - all three of which have been amended by 2007/47/EC [5].

Software development in medical domain is typically performed with traditional, plan-driven approaches like Waterfall and V-Model. The V-Model is perceived to be the *best fit* with regulatory requirements [6]. Some of the reasons why these methods are still valid today, despite their rigidness and limitations, can be listed as follows: (a) It is pretty straightforward to produce the necessary deliverables required to achieve regulatory audits with these models. (b) Verification, validation and risk assessments are particularly important in medical device software development and these processes are planned and executed in parallel with a corresponding development phase of the V-Model. (c) In these models, each phase must be completed before the next phase begins. This approach works well when there is high confidence in the requirements defined.

Ensuring regulatory requirements continuously is only one of the challenges that medical companies face. Some of others are managing the change during development, being timely to market, ensuring high quality, safety and high productivity. Agile software development methods have positive results for overcoming these challenges [7]. Therefore, there is a transition going on in medical device development companies to achieve agility as well as safety and reliability.

In this study, we evaluated Scrum [8], to understand the level of regulatory compliance when they are implemented. A mapping between these methods and the medical device software process assessment framework, MDevSPICE® has been performed for this purpose. The second purpose of this research is to reveal additional practices that have to be performed to ensure compliance when there is no specific adaptation of Scrum for the medical domain.

The rest of the paper is structured as follows: In Section 2, we provide the background for this research which includes brief descriptions of MDevSPICE® and Scrum.

We also provide a literature review of Scrum in the medical device development domain. In Section 3, we described the research methodology. In Section 4, we present the mapping and discuss the additional practices that have to be considered. In Section 5, we provide conclusions for this research.

2 Background

2.1 MDevSPICE®

MDevSPICE® is a medical device software process assessment framework developed with the purpose of integrating the regulatory requirements from the relevant medical device software standards and guiding medical device software developers to produce medical software that will be safe and reliable. It has been built upon 19 medical software development and software engineering standards, some of which can be seen on Fig 1.

The MDevSPICE® process assessment model is a two-dimensional model of the process quality characteristic of process capability. In one dimension, the process dimension, the processes are defined. In the other dimension, the capability dimension, a set of process attributes are grouped into capability levels. Processes in this process assessment model are described in terms of their Purpose, Process Outcomes, Base Practices and Work Products. Although the set of Process Outcomes is necessary and sufficient to achieve the Purpose of the process, the Base Practices together with Work Products provide a possible way to achieve the Process Outcomes. The list of processes in MDevSPICE® process assessment model is given in

Fig. **2**.

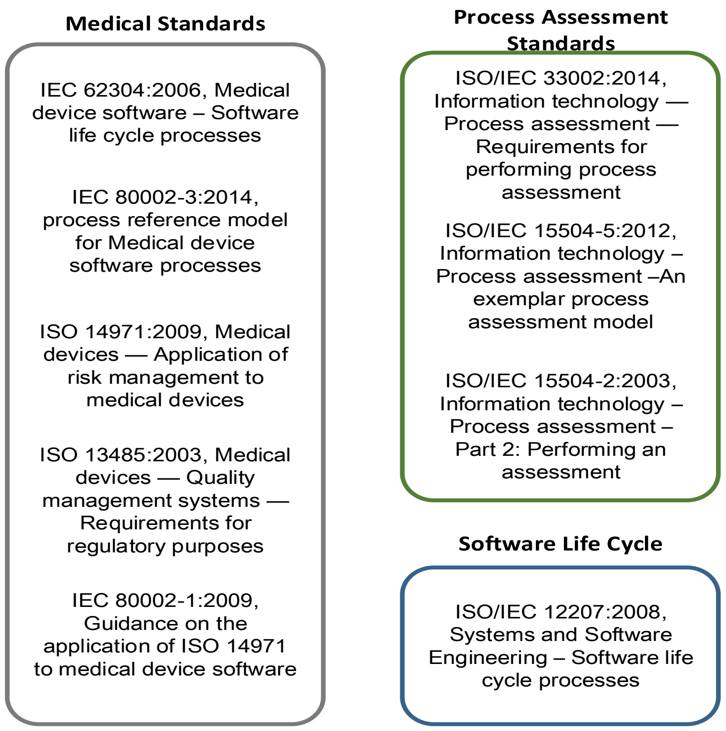
Medical Standards
IEC 62304:2006, Medical device software – Software life cycle processes
IEC 80002-3:2014, process reference model for Medical device software processes
ISO 14971:2009, Medical devices — Application of risk management to medical devices
ISO 13485:2003, Medical devices — Quality management systems — Requirements for regulatory purposes
IEC 80002-1:2009, Guidance on the application of ISO 14971 to medical device software
Process Assessment Standards
ISO/IEC 33002:2014, Information technology — Process assessment — Requirements for performing process assessment
ISO/IEC 15504-5:2012, Information technology – Process assessment –An exemplar process assessment model
ISO/IEC 15504-2:2003, Information technology – Process assessment – Part 2: Performing an assessment
Software Life Cycle
ISO/IEC 12207:2008, Systems and Software Engineering – Software life cycle processes

Fig. 1. Some of the Standards and Guidelines within MDevSPICE®

Safety classifications reflect the degree of harm that can result from medical device usage. Every medical device has to be assigned a safety class. Different international safety classification systems are in use throughout the world. There are three medical device safety classifications under US and EU regulations. Based on IEC 62304:2006, Class A devices are not intended to support or sustain human life, and may not lead to unreasonable risk of illness or injury. Class B medical devices may cause damage or harm to humans. Class C devices are usually those that support or sustain human life, and present a potential risk on illness or injury. Hand-held surgical instruments are Class A devices. An example of a Class B medical device is a powered wheelchair. An example of a Class C device is an implantable pacemaker.

The software safety classification of a medical device will determine the amount of IEC 62304 requirements that have to be fulfilled, with class A requiring much less practices to be put in place than for Class C. Additionally, the higher the safety classification the greater the amount of overhead associated with defining, implementing and providing objective evidence that the defined processes have actually been implemented. In this context, for each process, MDevSPICE® defines what outcomes have to be achieved and which base practices need to be performed for these safety classes.

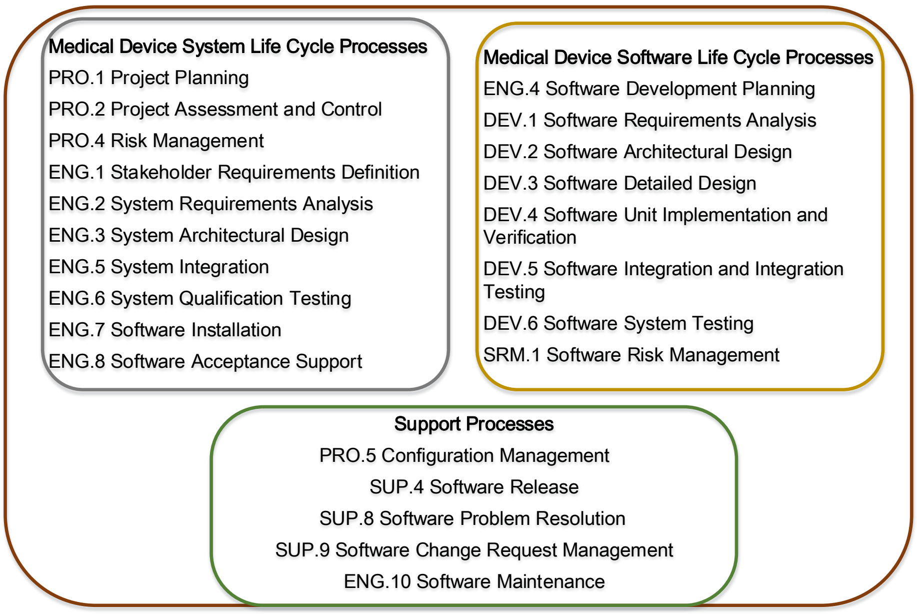
Medical Device System Life Cycle Processes
PRO.1 Project Planning
PRO.2 Project Assessment and Control
PRO.4 Risk Management
ENG.1 Stakeholder Requirements Definition
ENG.2 System Requirements Analysis
ENG.3 System Architectural Design
ENG.5 System Integration
ENG.6 System Qualification Testing
ENG.7 Software Installation
ENG.8 Software Acceptance Support
Medical Device Software Life Cycle Processes
ENG.4 Software Development Planning
DEV.1 Software Requirements Analysis
DEV.2 Software Architectural Design
DEV.3 Software Detailed Design
DEV.4 Software Unit Implementation and Verification
DEV.5 Software Integration and Integration Testing
DEV.6 Software System Testing
SRM.1 Software Risk Management
Support Processes
PRO.5 Configuration Management
SUP.4 Software Release
SUP.8 Software Problem Resolution
SUP.9 Software Change Request Management
ENG.10 Software Maintenance

Fig. 2. MDevSPICE® Processes

2.2 Scrum

Scrum was developed by Schwaber and Sutherland with the purpose of providing a management framework for software development [8, 9]. Scrum does not provide any specific technical practices for implementation.

The fundamental idea behind Scrum is to apply process control theory to software development to achieve flexibility, adaptability and productivity [7]. It relies on a set of values, principles and practices which can be adopted based on specific conditions. Scrum gives value on providing frequent feedback, embracing and leveraging variability, being adaptive, balancing upfront and just-in-time work, continuous learning, value-centric delivery and employing sufficient ceremony [10]. It offers effective solutions by providing specific roles, artifacts, activities and rules.

A Scrum Team consists of a Product Owner, a Scrum Master and the Development Team roles. Scrum Teams are self-organizing and cross-functional so that they could accomplish their work by themselves, rather than being directed by others outside the team and without depending on others not part of the team [9]. There are special events in Scrum which have been developed to create regularity and to minimize the need for meetings and are time-boxed.

2.3 Scrum Implementation in Safety Critical Domain

In the literature, we see many examples of Scrum implementation in the safety critical domain [11-16] . We briefly discuss some of these studies below:

Wolff [11] presents implementation of a formal specification language and Scrum with combination in an aircraft project. Executable specifications were used in order to validate system functionality, to understand the requirements and design of the system more precisely. In addition to conventional software implementation tasks within a sprint, formal specification investigation tasks were also defined.

Regulated Scrum [12] is an example of an adapted approach which has been implemented and validated in a highly regulated organization. Scrum was enhanced to ensure regulatory compliance in the medical domain. Some of the enhancements of the approach are having quality assurance people who ensure regulatory compliance at the end of each sprint (called continuous compliance), using templates to guide the development process, implementing coding standards and performing peer code review, establishing end-to-end traceability from the requirements elicitation stage to the code base with the help of tool support (called living traceability), risk management and continuous integration.

Another implementation of Scrum in a European space industry company with Test Driven Development, Continuous Integration and Pair Programming was discussed in [13]. Siemens Healthcare integrates Scrum into their software development process and additionally implements “feature orientation” practice to resolve the challenge of managing the flow of requirements coming from several product lines [14].

This literature review shows that Scrum was not used in the safety critical domain with their original versions, but, tailored for this domain and also combined with supplementary practices to ensure safety and regulatory compliance.

3 Research Approach

The purpose of this research is to reveal to what extend the regulatory requirements defined in MDevSPICE® are met when implementing Scrum. We defined the following research questions in relation to this purpose:

RQ1**:** How well the regulatory requirements of a safety Class B type medical device are met by through implementation of Scrum? ***RQ2:*** Which processes of MDevSPICE® are covered by implementing Scrum? ***RQ3:*** Which base practices of MDevSPICE® are covered by an implementation of Scrum? ***RQ4:*** What additional practices regarding those processes specified need to be performed in order to fully achieve a process at *Level 1: Performed Process*?

Research steps.

1. Listing Scrum practices at a fine granularity level.
2. Mapping MDevSPICE® base practices with Scrum Practices.
3. Identifying which processes were affected from the mapping.
4. Identifying the coverage ratio and deciding which MDevSPICE® base practices need to be included for those processes to satisfy a fully-achieved level.

Abrahamsson *et al.* [7], compared different agile software development methods to show which phases of software development were supported by these methods. Based on the comparison, Scrum covers project management, requirements specification, integration test and system test phases

However, instead of selecting these processes mentioned above first, and then checking the coverage within MDevSPICE®, we preferred to do the mapping in the other way around. We first listed the Scrum practices and then mapped them to MDevSPICE® base practices. With this approach we were able to identify which processes of MDevSPICE® were covered with a basic Scrum implementation.

Limitation of the Research.
Scrum could be taken as a prescriptive method with the descriptions of how the Scrum events will be performed and artifacts will be developed. However, Scrum is not defined at the practice description level provided by MDevSPICE®. Mapping of the method was limited to the given information in the following resource: The Scrum Guide™ by Ken Schwaber and Jeff Sutherland [17].

4 The Mappings and Discussions

Scrum and practices were mapped against MDevSPICE® (IEC 62304) Class B requirements. As the level of detail for the Scrum practices was limited, we needed to make some assumptions during the mapping. We assumed that process artifacts such as project plans or project monitoring reports would be developed during a Scrum implementation, as evidence required for the audits needed to be collected. Although it is very likely that some base practices would be performed during software development using Scrum, we couldn't rate a 100% coverage for them, as they might not be performed at the level of the detail required in MDevSPICE®. The coverage ratio is calculated based on the formula of: "the number of achieved base practices in a process / all base practices in a process".

4.1 Scrum Mapping

Scrum Method was described in terms of its roles, events and artifacts. Below, we provide the mapping for the roles and events. The artifacts which are basically product backlog and sprint backlog were not included in the mapping separately, as they were part of the events. Even though MDevSPICE® does not emphasize any specific roles, we mapped the activities that needs to be performed by the Scrum roles to the base practices of MDevSPICE®, shown in Table 1. In Table 2, the mapping between the Scrum events and the MDevSPICE® Processes and Base Practices are provided (**RQ2-RQ3**). The bold written text in the 3rd column of Table 1 and Table 2 show the mapped processes. The other text in the same column refer to the mapped base practices (BPs).

Table 1. Mapping of Scrum Roles‘ Activities and MDevSPICE® Processes & Base Practices

Scrum Roles	**Specific Activities of the Roles**	**MDevSPICE® Processes & Base Practices**
Product Owner	"— deciding which features and functionality to build and the order in which to build them — communicating to all other participants a clear vision of what the Scrum team is trying to achieve — being responsible for the overall success of the solution being developed or maintained"	**PRO.1 Project Planning** PRO.1.BP1: Define the scope of work PRO.1.BP3: Evaluate feasibility of the project. PRO.1.BP6: Define needs for experience, knowledge and skills. PRO.1.BP7: Identify and monitor project interfaces PRO.1.BP9: Allocate resources and responsibilities. PRO.1.BP11: Implement the project plan. **ENG.1 Stakeholder Requirements Definition** ENG.1.BP1: Identify stakeholders.
Scrum Master	"— helping everyone involved understand and embrace the Scrum values, principles, and practices. — helping the organization through the challenging change management process that can occur during a Scrum adoption — protecting the team from outside interference and takes a leadership role in removing impediments that inhibit team productivity"	**PRO.1 Project Planning** PRO.1.BP2: Define life cycle model for the project.
Dev-Team	"— a diverse, cross-functional collection of these types of people who are responsible for designing, building, and testing the desired product"	**PRO.1 Project Planning** PRO.1.BP4: Define and maintain estimates for project attributes PRO.1.BP5: Define project activities and tasks. PRO.1.BP8: Define project schedule.

		PRO.1.BP10: Establish project plan. PRO.1.BP11: Implement the project plan.

Table 2. Mapping of Scrum Events and MDevSPICE® Processes & Base Practices

Scrum Events	**Descriptions of the Events**	**MDevSPICE® Processes and Base Practices**
Sprint Planning	"The work to be performed in the Sprint is planned at the Sprint Planning. This plan is created by the collaborative work of the entire Scrum Team."	**PRO.1 Project Planning** PRO.1.BP4: Define and maintain estimates for project attributes PRO.1.BP5: Define project activities and tasks. PRO.1.BP7: Identify and monitor project interfaces.
Daily Scrum	"A 15-minute time-boxed event for the Development Team to synchronize activities and create a plan for the next 24 hours."	**PRO2. Project Assessment and Control** PRO.2.BP3: Report progress of the project. PRO.2.BP4: Perform project review.
Sprint Review	"A meeting held at the end of the Sprint to inspect the Increment and adapt the Product Backlog. The timeline, budget, potential capabilities, and marketplace for the next anticipated release of the product are reviewed"	**PRO2. Project Assessment and Control** PRO.2.BP1: Monitor project attributes PRO.2.BP2: Monitor project interfaces PRO.2.BP3: Report progress of the project. PRO.2.BP4: Perform project review. PRO.2.BP5: Act to correct deviations.
Sprint Retrospective	"A meeting to inspect how the last Sprint went with regards to people, relationships, process, and tool"	**PRO2. Project Assessment and Control** PRO.2.BP6: Collect project experiences
Product Backlog Grooming	"Product Backlog (PB) is an ordered list of everything that might be needed in the product and is the single source of requirements for any changes to be made to the product." "PB lists	**ENG.1 Stakeholder Requirements Definition** ENG.1.BP2: Obtain requirements. ENG.1.BP3: Define constraints. ENG.1.BP4: Define user interaction**.** ENG.1.BP5: Identify critical requirements.

	all features, functions, requirements, enhancements, and fixes that constitute the changes to be made to the product in future releases. Product Backlog items have the attributes of a description, order, estimate and value." "PB Grooming is the act of adding detail, estimates, and order to items in the Product Backlog. This is an ongoing process in which the Product Owner and the Dev-Team perform"	ENG.1.BP6: Evaluate requirements ENG.1.BP7: Agree on requirements. **ENG.2 System Requirements Analysis** ENG.2.BP1: Establish system requirements. ENG.2.BP3: Optimize project solution. ENG.2.BP4: Analyze system requirements. ENG.2.BP5: Evaluate and update system requirements. ENG.2.BP7: Communicate system requirements. **DEV.1 Software Requirements Analysis** DEV.1.BP1: Define and document all software requirements. DEV.1.BP2: Prioritize requirements. DEV.1.BP6: Evaluate and update requirements DEV.1.BP7: Baseline and communicate software requirements.

According to the mapping shown in Table 1 and Table 2, Scrum is related to 5 processes of MDevSPICE® when it is implemented fully (RQ2). Within the mapping process, we also evaluated and calculated the coverage ratio of the MDevSPICE® base practices for Scrum. Table 3, shows the coverage ratio for each mapped process. The coverage evaluation performed by one of the authors for base practices from a Scrum perspective, was subjective, but peer reviewed by the other author. Therefore, depending on the implementation details and perception of the methods, different coverage ratios than we provided could be obtained. However, the purpose of giving this ratio is to provide readers and practitioners with an indication of how much value is achieved with basic Scrum implementation and how much needs to be done more from a regulatory perspective.

Table 3. Coverage of Mapped MDevSPICE® Processes from Scrum Perspective

	Mapped MDevSPICE® Processes	**Coverage Ratios**
1.	PRO.1 Project Planning	100%
2.	PRO.2 Project Assessment and Control	90%
3.	ENG.1 Stakeholder Requirements Definition	55%
4.	ENG.2 System Requirements Analysis	71%
5.	DEV.1 Software Requirements Analysis	33%

Below, we discuss why processes #3, #4, and #5 in Table 3 did not have a full coverage ratio and what additional practices need to be performed for compliance to medical requirements (**RQ4**).

#3 ENG.1 Stakeholder Requirements Definition Process: (Coverage Ratio: 5 BPs / 9 BPs). The following base practices of ENG.1 are assumed to be achieved by the product owner and the development team in product backlog grooming sessions: *ENG.1.BP1: Identify stakeholders, ENG.1.BP2: Obtain requirements, ENG.1.BP3: Define constraints, ENG.1.BP6: Evaluate requirements, ENG.1.BP7: Agree on requirements*. However, the other base practices of this process need special attention which are not addressed in Scrum.

For an IEC 62304 Class B type medical software, user interaction has to be defined and evidence has to be provided. Based on the *ENG.1.BP4: Define user interaction* base practice the following information has to be defined for a medical device:

– Intended medical indication, e.g. conditions(s) or disease(s) to be screened, monitored, treated, diagnosed, or prevented; – Intended patient population, e.g. age, weight, health, condition; – Intended part of the body or type of tissue applied to or interacted with; – Intended user profile; – Intended conditions of use, e.g. environment including hygienic requirements, frequency of use, location and mobility; and –Operating principle.

In a product backlog grooming session, we may assume that all stakeholder requirements are specified. However, as part of the *ENG.1.BP5: Identify critical requirements* practice of MDevSPICE®; it has to be ensured that *health, safety, security, environment and other stakeholder requirements and functions that relate to critical qualities and* *shall address possible adverse effects of use of the system on human health and safety* are identified as well.

In medical device software development, every change on the product, whether it is on the artifacts or the code has to be made in a controlled way. This is one of the major contradictions between agile and the regulated worlds. For a change to be controlled, a version control system should be in place and baselines established. This is referred to in *ENG.1.BP8: Establish stakeholder requirements baseline* base practice. However, a product backlog is a dynamic list which is continuously changing and no baselines are taken over it.

The other major requirement in medical device software development is to build traceability links between artifacts as this plays a significant role in defect management and change management. This is referred to in *ENG.1.BP9: Manage stakeholder requirements changes.* The purpose is to "Maintain stakeholder requirements traceability to the sources of stakeholder need". However, there is no specific emphasis on the development of a traceability schema in Scrum method.

#4 ENG.2 System Requirements Analysis Process: (Coverage Ratio: 5 BPs / 7 BPs). We may assume that base practices: *ENG.2.BP1: Establish system requirements, ENG.2.BP3: Optimize project solution, ENG.2.BP4: Analyze system requirements, ENG.2.BP5: Evaluate and update system requirements, ENG.2.BP7: Communicate system requirements* are performed in product backlog grooming sessions, as there are mechanisms to achieve them. However, the following two base practices need to be handled separately.

As part of *ENG.2.BP2: Assign a safety class to the medical device based on the regional regulations* process, at the system requirements analysis phase, a safety class has to be assigned to the product as the specific regulations apply based on the safety class in order to prevent potential harm to human life. As mentioned also in base practice ENG.1.BP9, bilateral traceability between the stakeholder requirements and the system requirements needs to be established as part of *ENG.2.BP6: Ensure consistency* base practice.

#5 DEV.1 Software Requirements Analysis Process: (Coverage Ratio: 3 BPs / 9 BPs) We assumed that base practice, *DEV.1.BP1: Define and document all software requirements* is partially achieved, as there are specific issues that needs to be addressed for this BP. Based on FDA rules, software requirements have to be documented in a software requirements specification document and this document should contain details of the software functions.

It is important to determine the interfaces between the software requirements and other elements of the operating environment such as third party software. This is achieved as part of base practice, *DEV.1.BP3: Determine the impact the requirements have on the operating environment.* At this stage, it is expected that the acceptance criteria for the software tests are defined from software requirements *(DEV.1.BP4: Develop acceptance criteria for software testing based on the software requirements.)* Scrum does not have such a rule.

As mentioned above, consistency of system requirements to software requirements has to be ensured. This is achieved through establishing and maintaining bilateral traceability between system requirements and the software requirements *(DEV.1.BP5: Verify all software requirements.)*

The 7th base practice of DEV.1 requires establishing a baseline of software requirements and also providing communication of the software requirements. Due to use of communication channels in Scrum, we feel that the second part of this base practice can be achieved. However, the baseline of software requirements should also be added.

In medical device software development, special attention is given to risk analysis and mitigation. With base practices, *DEV.1.BP.9: Re-evaluate and maintain medical device risk analysis* and *DEV.1.BP8: Establish and maintain risk control measures in software requirements*, it is ensured that risks regarding the software requirements are identified and risk control measures are defined. Risk management should be a part of daily or weekly Scrum review meetings.

Although we have mapped the Stakeholder, System and Software Requirements Analysis processes with the product backlog grooming practice in Scrum, it is necessary to ensure that distinction between these requirement types are clear, the traceability links are established, and the changes made to them is managed.

In MDevSPICE®, there another process, **ENG.4 Software Development Planning** includes very specific practices for regulatory requirements compliance. Some of these base practices include *assigning the software safety class of the software system, having a software integration test plan, a verification plan, a software risk management plan* *and configuration management plan.* Although Scrum proposes effective ways to manage projects, these plans are not part of a basic Scrum method. Therefore, we assumed that *ENG.4 Software Development Planning* is not covered with Scrum, even though it is a "planning" process.

5 Conclusions

In this study, we evaluated if a Scrum implementation could meet the regulatory requirements defined in MDevSPICE®, the software process assessment framework for medical device software development. Scrum was selected due to its high recognition and adoption in software development world. The research approach included the mapping of Scrum practices to MDevSPICE® processes and base practices. With this approach, we were able to define MDevSPICE® processes and base practices that could be achieved in a basic Scrum implementation, more importantly the additional base practices that have to be performed for ensuring safety and regulatory compliance.

We also identified the coverage ratio of MDevSPICE® processes from a Scrum perspective. Even though the coverage ratios are calculated from a subjective point of view, they provide important information to readers and practitioners about which MDevSPICE® processes are covered to what extent.

The significance of this study is that it presents a coverage analysis at the MDevSPICE® base practice level which is very detailed and has never been performed before. The coverage ratios showed the level of the gap between methods. The study has also revealed conflicting practices such as "controlled change management over continuous and dynamic change". In addition, the discussions made around the additional practices that need to be performed, complete the missing pieces to ensure safety and be successful over a regulatory audit in the medical device domain. The results of this study also provide guidance us for the development of an agile integrated medical device software development framework.

As future work, we will extend the mapping by adding XP, other agile methods which propose a whole software development life cycle coverage such as Dynamic Systems Development Method and scaling agile frameworks such as Disciplined Agile Delivery and SAFE.

Acknowledgement. This research is supported by Science Foundation Ireland under a co-funding initiative by the Irish Government and European Regional Development Fund through Lero - the Irish Software Research Centre (http://www.lero.ie) grant 13/RC/2094. This research is also partially supported by the EU Ambient Assisted Living project – Maestro.

References

[1] FDA. (15.05). *Chapter I - Food and drug administration, department of health and human services subchapter H - Medical devices, Part 820 - Quality system regulation*. Available: http://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfcfr/CFRSearch.cfm?CFRPart=820

[2] *Directive 93/42/EEC of the European Parliament and of the Council concerning medical devices*, 1993.

[3] *Council directive 90/385/EEC on active implantable medical devices (AIMDD)*, 1990.

[4] *Directive 98/79/EC of the european parliament and of the council of 27 october 1998 on in vitro diagnostic medical devices*, 1998.

[5] *Directive 2007/47/EC of the European Parliament and of the Council concerning medical devices*, 2007.

[6] M. Mc Hugh, O. Cawley, F. McCaffcry, I. Richardson, and X. Wang, "An agile v-model for medical device software development to overcome the challenges with plan-driven software development lifecycles," in *Software Engineering in Health Care (SEHC), 2013 5th International Workshop on*, 2013, pp. 12-19: IEEE.

[7] P. Abrahamsson, O. Salo, J. Ronkainen, and J. Warsta, "Agile software development methods: Review and analysis," ed: VTT Finland, 2002.

[8] K. Schwaber, "Scrum development process," in *Business Object Design and Implementation*: Springer, 1997, pp. 117-134.

[9] K. Schwaber and J. Sutherland, *Software in 30 days: how agile managers beat the odds, delight their customers, and leave competitors in the dust*. John Wiley & Sons, 2012.

[10] K. S. Rubin, *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Addison-Wesley Professional, 2012.

[11] S. Wolff, "Scrum goes formal: Agile methods for safety-critical systems," in *Proceedings of the First International Workshop on Formal Methods in Software Engineering: Rigorous and Agile Approaches*, 2012, pp. 23-29: IEEE Press.

[12] B. Fitzgerald, K.-J. Stol, R. O'Sullivan, and D. O'Brien, "Scaling agile methods to regulated environments: An industry case study," in *Software Engineering (ICSE), 2013 35th International Conference on*, 2013, pp. 863-872: IEEE.

[13] E. Ahmad, B. Raza, R. Feldt, and T. Nordebäck, "ECSS standard compliant agile software development: an industrial case study," in *Proceedings of the 2010 National Software Engineering Conference*, 2010, p. 6: ACM.

[14] M. Kircher and P. Hofman, "Combining systematic reuse with Agile development: experience report," in *Proceedings of the 16th International Software Product Line Conference-Volume 1*, 2012, pp. 215-219: ACM.

[15] R. Faber, "Architects as service providers," *IEEE software,* vol. 27, no. 2, 2010.

[16] J. W. Spence, "There has to be a better way![software development]," in *Agile Conference, 2005. Proceedings*, 2005, pp. 272-278: IEEE.

[17] J. Sutherland and K. Schwaber, "The scrum guide," *The Definitive Guide to Scrum: The Rules of the Game. Scrum. org,* 2013.