

2015-08-07

Adopting Agile Practices when Developing Medical Device Software

Martin McHugh

Technological University Dublin, martin.mchugh@tudublin.ie

Fergal McCaffery

Dundalk IT, fergal.mccaffery@dkit.ie

Garret Coady

Bluebridge Technologies, garretcoady@bluebridgetech.com

Follow this and additional works at: <https://arrow.tudublin.ie/ittsciart>

Recommended Citation

McHugh, M., McCaffery, F. & Coady, G. (2015) Adopting Agile Practices when Developing Regulatory Compliant Software, *J Comput Eng Inf Technology*, 4(3)

This Article is brought to you for free and open access by the School of Science and Computing at ARROW@TU Dublin. It has been accepted for inclusion in Articles by an authorized administrator of ARROW@TU Dublin. For more information, please contact yvonne.desmond@tudublin.ie, arrow.admin@tudublin.ie, brian.widdis@tudublin.ie.



This work is licensed under a [Creative Commons Attribution-Noncommercial-Share Alike 3.0 License](https://creativecommons.org/licenses/by-nc-sa/3.0/)

Adopting Agile Practices when Developing Medical Device Software

Martin McHugh¹, Fergal McCaffery², and Garret Coady³

¹School of Computing, Dublin Institute of Technology, Ireland

²Regulated Software Research Centre, Dundalk Institute of Technology, Ireland

³BlueBridge Technologies, 3015 Lake Dr., Citywest, Dublin Ireland

martin.mchugh@dit.ie, fergal.mccaffery@dkit.ie, garretcoady@bluebridgetech.com

Abstract— Agile methods are gaining momentum amongst the developers of non-safety critical software. They offer the ability to improve development time, increase quality and reduce development costs. Despite this, the rate of adoption of agile methods within safety critical domains remains low. On face value agile methods appear to be contradictory to regulatory requirements. However while they may appear contradictory, they align on key values such as the development of the highest quality software. To demonstrate that agile methods could in fact be adopted when developing regulatory compliant software they were implemented on a medical device software development project. This implementation showed that not only can agile methods be successfully followed, but it also revealed that benefits were acquired. For example, the medical device software development project was completed 7% faster when following agile methods, when compared to if it had been completed in accordance with a plan-driven approach. While this implementation is confined to a single project, within a single organization it does strengthen the belief that adopting agile methods within regulated domains can reap the same benefits as those acquired in non-safety critical domains.

Index Terms—Agile, Scrum, Medical, Regulated, FDA, Software, MDD

I. INTRODUCTION

Traditionally, software was developed in accordance with plan-driven approaches such as the Waterfall model. The plan-driven approach which later became known as the Waterfall model was first proposed by Winston Royce in 1970 [3]. Plan-driven approaches dictate that each stage of a development project should be fully completed before moving on to the next stage. However, while Royce first formalized the Waterfall model, he discussed it as a flawed approach to software development as it is not wholly possible to ensure you will not need to revisit a stage during a development project.

Recognizing the inadequacies associated with plan-driven approaches, a shift has occurred toward a more flexible or agile approach to software development. Agile software development was first formalized in 2001 and since then has gained greater acceptance in the software development industry. This is evident in a large scale survey, conducted in 2013, which identified that 88% organizations stated that they were following an agile approach [4¹]. This is an increase from 84% in 2012 [5²] and 80% in 2011 [6³].

Agile approaches are ideally suited to small teams in co-located environments [7]. While they are suited to these conditions, they are not precluded from use by large and distributed teams. Research has been conducted which confirms this [8, 9]. However, the research presented here is primarily concerned with the development of software for use in the medical device domain. As a result, research was conducted into the use of agile approaches in this domain. A comprehensive mapping study [10] was performed to identify where agile practices have been used in regulated domains. This mapping study revealed that despite the widespread adoption of agile methods in non-regulated domains there is still a low rate of adoption among software development organizations in regulated domains. While the medical device domain revealed a low rate of agile adoption of agile practices, of the regulated domains, agile practices have been adopted more in the medical domain than the other regulated domains such as the avionics and automotive.

Software development organizations in regulated domains experience unique challenges. The greatest of which is the necessity to adhere to regulatory controls within region the software is to be marketed. Prior to 2007, software was deemed a component of a hardware medical device. The efficacy of the medical device could be established and measured by examining the physical device. However in 2007 the European Union (EU) released its latest amendment to the Medical Device Directive (MDD) 1993/42/EEC

¹ <http://www.versionone.com/pdf/2013-state-of-agile-survey.pdf>

² <http://www.versionone.com/pdf/2012-state-of-agile-survey.pdf>

³ <http://www.versionone.com/pdf/2011-state-of-agile-survey.pdf>

[11] known as 2007/47/EC [12]. The most significant impact of this amendment with regards to software, is that now software can be considered as a medical device in its own right [13] resulting in the previous methods of ensuring the efficacy of the medical device consisting only of software being no longer viable. As a result regulatory bodies require medical device software organizations to produce deliverables, which demonstrate the efficacy and safety of medical device software. The deliverables include comprehensive documentation detailing the processes, which have been followed when developing the software. Furthermore, medical device software organizations are required to inform regulatory bodies of the specification of the software, which they are producing prior to commencing development.

Initially the Agile Manifesto presents the four key values of agile software development:

1. Individuals and Interactions *over* Processes and tools;
2. Working Software *over* Comprehensive Documentation;
3. Customer Collaboration *over* Contract Negotiation;
4. Responding to Change *over* Following a Plan

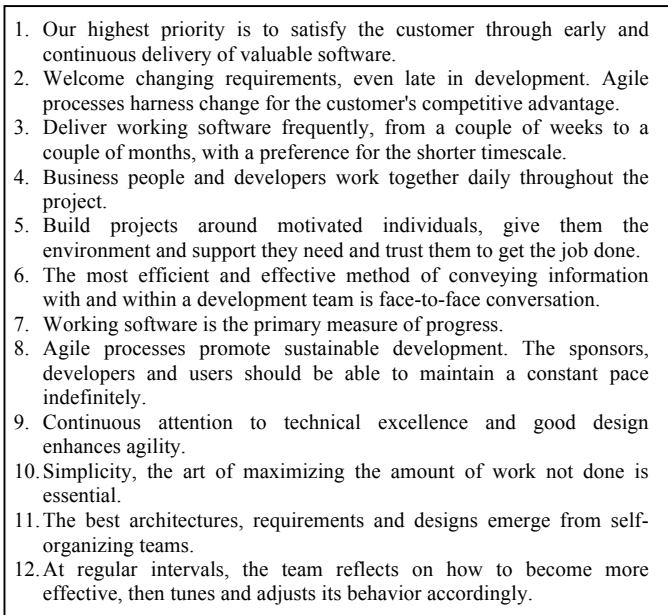
- 
1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
 2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
 3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale.
 4. Business people and developers work together daily throughout the project.
 5. Build projects around motivated individuals, give them the environment and support they need and trust them to get the job done.
 6. The most efficient and effective method of conveying information with and within a development team is face-to-face conversation.
 7. Working software is the primary measure of progress.
 8. Agile processes promote sustainable development. The sponsors, developers and users should be able to maintain a constant pace indefinitely.
 9. Continuous attention to technical excellence and good design enhances agility.
 10. Simplicity, the art of maximizing the amount of work not done is essential.
 11. The best architectures, requirements and designs emerge from self-organizing teams.
 12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Fig.1. 12 Agile Principles of the Agile Manifesto [2]

Cursory reading of these values would appear to suggest that agile methods are contradictory to regulatory requirements as none of the regulatory deliverables such as documentation are produced. Further examination of the agile values identified that the statements on the left are deemed of greater importance than those on the right in an agile project, however they do not replace the items on the right. For example, as identified by Robert Martin [2], one of the authors of the Agile Manifesto states “*Produce no document unless it's immediate and significant*” demonstrating that as long as there is value to be obtained by the items on the right, they should still be produced.

Another example as to how agile methods may be incompatible in the medical device domain is the difficulty associated with incremental development. In non-regulated software development, adopting agile approaches, the software is developed partially, examined and if necessary, reiterated. This examination can come in the form of alpha or beta testing or possibly release to end users with the subsequent elements being updated based on feedback obtained. In the medical device domain and other safety-critical domains, it is not possible to release software into a live environment without adequate testing and regulatory approval.

In this research, a medical device software development organization identified a number of challenges, which they were experiencing as a result of following a plan-driven approach. This research sought to identify ways in which these challenges could be overcome through the adoption of agile practices. To accompany the identification of agile practices, they were implemented on a medical device software development project to test whether they met their desired objective.

The structure of this paper is as follows; in the next section an overview of the mainstream agile methods is presented, following this an overview of key regulations and standards applicable to medical device software organizations is presented. Section IV presents an overview of previous research into the adoption of agile practices. Section V presents the approach taken by this research. Details of the implementation of agile practices within a medical device software development organization are discussed in section VI. Discussion surrounding the findings and limitations of this research are then presented and finally the conclusions and future work is presented.

II. AGILE SOFTWARE DEVELOPMENT

Prior to 2001 a common misconception often held was that agile software development was the same as iterative software development and as a result software development methods such as the Spiral Model and Iterative and Incremental Development were considered agile as a result. However in 2001 with the publication of the Agile Manifesto by the Agile Alliance, a definition as to what agile software development consists was provided. A number of software development methods have been identified as being agile, such as eXtreme Programming (XP), Scrum, Disciplined Agile Delivery (DAD), Agile Modelling (AM), Dynamic Systems Development (DSDM) and Feature Driven Development (FDD). A common element of each of these agile methods is that each uses the 12 principles of agile software development as defined by the agile manifesto as their foundation.

A. Scrum

The term Scrum originates from the game of Rugby Union. The software engineering approach known today as Scrum first originated in a paper written by Takeuchi and Nonaka [14] for the Harvard Business Review. In this paper, the authors detailed a Scrum approach adopting six characteristics: built in stability; self-organizing teams; overlapping development phases; multilearning; subtle control and organizational transfer of learning. These characteristics have been expanded upon and a number of practices now exist which combine to create the Scrum approach to software development.

A number of practices included as part of the Scrum approach are shown in figure 2. It can be seen that initially a Product Backlog, which contains a list of high-level requirements, is created which informs a Sprint Backlog. The Sprint Backlog is an ordered list of software items, which must be completed to satisfy the high level requirement contained in the Product Backlog. Sprints, which should be completed in between two and four weeks, however, this is not mandatory, with meetings every 24 hours. Once a Sprint is completed a potentially shippable product increment, also known as a prototype should be produced. The Scrum approach is seen as a project management approach as it provides high level guidance in each stage of development [15].

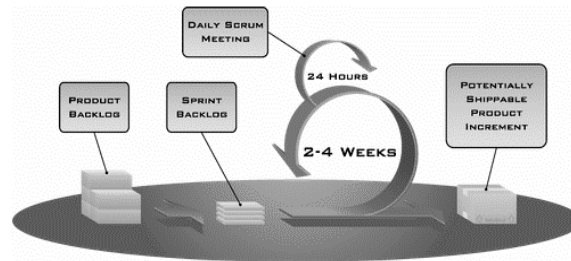


Fig. 2 Scrum Development Lifecycle [1]

B. eXtreme Programming

XP developed by Kent Beck whilst he was working on the Chrysler Comprehensive Compensation System. Unlike Scrum, which focuses on the entire software development project, XP primarily focuses on the software implementation element of a software development project [16].

With XP, user stories contribute to creating test scenarios and release planning, as do system metaphors. A release plan is produced as a result of release planning. The release plan defines what is to be completed during an iteration. During each iteration the software is implemented and tested in accordance with the test scenarios defined as part of the user stories. Once a single iteration is completed another one is undertaken. Each iteration is integrated and small releases are produced for customer approval.

C. Disciplined Agile Delivery

DAD [17] is a hybrid process framework which combines strategies from mainstream agile methods. DAD draws from Scrum, XP, Agile Modelling (AM), Unified Process (UP), Agile Data (AD) and Kanban. The creators of DAD recognized that not all of the agile methods are equal and have strengths and weaknesses in different areas. To accompany this, the creators of DAD recognized that most organizations adopt a Scrum approach as a foundation and then select appropriate practices from other agile methods. A pattern emerged that different organizations were selecting the same practices and as a result, the creators of DAD felt it prudent to create a framework, which uses Scrum as a foundation, which already includes the practices commonly added by organizations.

As with other agile methods, DAD has adopted simple and intuitive terms where possible. For example, in Scrum there is a Scrum Master, in XP there is a Coach and in DAD there is a Team Lead. This is simply a matter of terminology as each of these roles perform the same functions.

III. MEDICAL DEVICE SOFTWARE DEVELOPMENT

A. Standards

In November 1997, the FDA signed into law the Modernization act, known as the Food and Drug Administration Modernization Act (FDAMA). A key element of FDAMA is the advocating of the use of standards in the design review process. To support the FDAMA, the FDA published in the Federal Register a list of standards to which medical device manufacturers could declare conformity. A key objective of the FDAMA was to reduce the burden on both the FDA and medical device manufacturers by reducing the regulatory obstacle to entry to international and domestic medical device markets. When the FDAMA was signed into law, the Centre for Devices and Radiological Health (CDRH) established Standards Technology Groups (STG), one of which had a specific focus on software. A STG is responsible for software categorized as follows:

- General process standards, which are technology independent;
- General process standards, which are technology dependent;
- Specific process implementations.

A number of standards are included on the federal register list of standards, of most significance with regards to medical device software development is IEC 62304:2006 Medical Device – Software Life Cycle Processes. Also of significance to medical device software and all types of medical device is ISO 14971 Application of Risk Management to Medical Devices. In the EU, ISO 13485 Medical Devices – Quality Management Systems – Requirements for Regulatory Purposes is very central to the development of regulatory compliant software; however, prior to March 2012, medical device companies wishing to market a medical device within the US were required to adhere to the FDA QSR regulations. Therefore, if a medical device manufacturer was developing a medical device for use in the EU and the US, they needed to conform to both of the QMS guidelines. However, in March 2012, the FDA commenced a pilot program offering device manufacturers the option of submitting their quality system audits, which are compliant with ISO/IEC 13485:2003 to the FDA’s Centre for Devices and Radiological Health (CDRH) or Centre for Biologics Evaluation and Research (CBER). This is seen as a step forward in the FDA’s plan to create a partnership with Health Canada, which would result in a single audit program for both the US and Canada.

1) IEC 62304:2006 Medical Device – Software Life Cycle Processes

As medical devices are safety critical, manufacturers are recommended to follow current international standards during development. Adherence to these standards is not mandatory, but it is recommended in order to achieve regulatory approval in both the EU and the US. Adherence to the standards demonstrates the manufacturer’s ability to follow defined development procedures and their ability to perform the required risk management activities [18]. If a manufacturer chooses not to adhere to these standards, they must provide a sufficient explanation as to why not and they must demonstrate that the alternative method chosen is equally valid.

IEC 62304:2006 is the current software development standard followed by medical device software developers. The current version of IEC 62304 was released in 2006. IEC 62304 is derived from ISO/IEC 12207:1995 Software Lifecycle Processes [19], AMD 1:2002 [20] and 2:2004 [21]. ISO 12207:1995 is not domain specific although it is seen as being comprehensive in its approach and this is reflected in the number of standards that utilize the core principles of ISO 12207:1995, AMD 1 and AMD 2 as their foundation such as ISO/IEC 15504-5:2006. IEC 62304 is domain specific and is tailored to suit the specific requirements of the medical device software development industry. IEC 62304 is a software development standard and provides end to end guidance in the development of the software component of a medical device, however, it hands off system activities such as Requirements Elicitation and Validation, to its aligned standards which include ISO 13485:2003 Medical Devices – Quality Management Systems, ISO 14971:2007 Medical Devices – Application of Risk [22] and ISO/IEC 15288:2008 – Systems and Software engineering – System lifecycle processes [23]. IEC 62304 is a harmonized standard under the MDD [24] and is a FDA consensus standard [25].

IEC 62304 makes provision for the application of risk to software. IEC 62304 applies a classification system to software components similar to that of ISO 14971 Clause 4.4.5 and 6.1. The safety classification is as follows:

- Class A – No injury or damage to health possible;
- Class B – Non-serious injury is possible;
- Class C – Death or serious injury is possible.

The risk classification applied to an item of software is determined by the amount of potential risk the medical device places upon the patient, clinician or third party. With IEC 62304, the overall software component assumes the safety classification of the software element that poses the most risk. However, IEC 62304 does make allowance for software components to be segregated into individual software elements with each element receiving its own safety classification.

As the current version of IEC 62304 was released in 2006, prior to the release of the revised MDD and the FDA MDDS rule, it does not cater for the specific changes in the regulations. As mentioned, the revised MDD makes provision for standalone software to be an active medical device and the only component of a medical device subject to regulatory conformance. However, IEC 62304 states;

“This standard does not cover validation and final release of the medical device, even when the medical device consists entirely of software”

Consequently, validation and final release must be covered in aligned standards, but these aligned standards can be difficult to apply to standalone software as set out in the revised MDD.

2) ISO 13485:2012 Medical Devices – Quality Management Systems – Requirements for Regulatory Purposes

ISO 13485:2012 was published in 2012, which forms the basis for the development of a quality management system when developing medical device software. ISO 13485 is derived from ISO 9001; however, ISO 13485 is tailored to include elements specific to the development of medical devices. Additionally, ISO 9001 requires a device manufacturer to perform continuous improvement, while ISO 13485 only requires a manufacturer to implement and maintain a quality management system. As previously discussed, ISO 13485 is a harmonized standard as part of the MDD.

3) ISO 14971:2012 – Application of Risk Management to Medical Devices

ISO 14971 was first released in 2000 with the second edition released in 2007 and the latest version released in 2012. ISO 14971 is a FDA consensus standard and is harmonized as part of the MDD. It specifies the procedures and activities for identifying hazards in medical devices and accessories to medical devices, including software. ISO 14971 provides guidelines to medical device manufacturers on the preparation of a plan to prepare for risk management activities. The plan should contain:

- The scope of the plan;
- A verification plan;
- Allocation of responsibilities;
- Requirements for review of risk management activities;
- Requirements for collecting and reviewing production and post-production information;
- Criteria for risk acceptability.

ISO 14971 is not specific to the development of medical device software. As a result it can be difficult to apply it to the development of medical device software. Consequently, IEC produced a Technical Report (TR) providing guidance to medical device manufacturers on applying ISO 14971. This guidance document is known as IEC/TR 80002-1:2009 – Part 1: Guidance on the application of ISO 14971 to medical device software. IEC/TR 80002-1:2009 follows the same structure as ISO 14971, making it easier to follow for those familiar with ISO 14971’s structure.

B. Regulations

All medical devices intended for use within the EU must conform to the MDD and its latest amendment. The latest amendment MDD known as 2007/47/EC was released on October 11th 2007. However, it only became mandatory for CE compliance on March 21st 2010 [26]. The amendment to the MDD (2007/47/EC) is harmonized with a number of standards relating to the production of medical devices e.g. EN ISO 14971:2009 and IEC EN 62304:2006. MDD (2007/47/EC) Article I Section 2 defines a medical device as [27]:

*“any instrument, apparatus, appliance, **software**, material or other article, whether used alone or in combination, including the software intended by its manufacturer to be used specifically for diagnostic and/or therapeutic purposes and necessary for its proper application”*

As highlighted in the previous definition as to what constitutes a medical device, software can be considered a medical device. Whilst the original MDD (1993/42/EEC) allowed for software to be seen as a medical device, it did not extend to standalone software being recognized as an active medical device. The amendment to the MDD (2007/47/EC) Annex IX Section 1.4 defines an active medical device as:

“any medical device operation of which depends on a source of electrical energy or any source of power other than that directly generated by the human body or gravity and which acts by converting this energy. Medical devices intended to transmit energy, substances or other elements between an active medical device and the patient, without any significant change, are not considered to be active medical devices. Stand-alone software is considered to be an active medical device”

Methods used to ensure device conformity to the MDD (1993/42/EEC) have not been modified with the release of MDD (2007/47/EC) even though the definition of a medical device has changed with particular reference to standalone software being capable of being an active medical device. An example of software as an active medical device is software used to plan cancer treatment doses and to control the setting of oncology treatment devices.

In 1981, the FDA began to investigate the use of software in healthcare. Initially, the FDA classified medical device software based upon its Draft Software Policy published in 1987 and revised in 1989 [28]. However, the FDA recognized that as the rate of computer and software-based products was growing at an exponential rate, it was not practical to adopt a single “software” policy, which would cover all computer, and software based products. Consequently, the draft software policy was withdrawn in January 2005 [29]. As a result, the FDA does not specifically regulate software; rather they regulate devices used in healthcare, which meet the definition of being a medical device. The FDA definition of what constitutes a medical device is outlined in section 201(h) of the Federal Food Drug & Cosmetic (FD&C) Act [30]:

“an instrument, apparatus, implement, machine, contrivance, implant, in vitro reagent, or other similar or related article, including a component part, or accessory which is:

- *recognized in the official National Formulary, or the United States Pharmacopoeia, or any supplement to them,*

- *intended for use in the diagnosis of disease or other conditions, or in the cure, mitigation, treatment, or prevention of disease, in man or other animals, or*
- *intended to affect the structure or any function of the body of man or other animals, and which does not achieve its primary intended purposes through chemical action within or on the body of man or other animals and which is not dependent upon being metabolized for the achievement of any of its primary intended purposes."*

It can be seen from the definition provided that if software performs any of the functions outlined in the definition of a medical device, then it becomes subject to scrutiny by the FDA.

IV. PREVIOUS WORK

In 2007, Denger et al. [31] released the findings of a survey exploring the challenges experienced by medical device software development organizations. The organizations that participated in this survey typically develop software in accordance with a plan-driven SDLC. This survey revealed that 63% of respondents reported experiencing difficulties in the area of requirements engineering. Additionally, 16% of respondents reported experiencing difficulties with the software architecture stage of development.

In 2010, Embedded Market Forecasters [32] performed a survey of the entire software development industry but extracted results to specific domains within the software development industry such as the development of software for use in medical devices. This survey revealed that 54% of medical device software projects over run and that 9% of medical device software development projects failed. The reasons cited for these projects failing were:

- Requirements Management;
- Time Management;
- Design Complexity.

As the regulatory environment changed since these surveys, the Authors of this research felt it necessary to confirm if the challenges identified in the previous surveys were still challenges in the current regulatory environment and to establish if any new challenges emerged. In [33] it emerged that a number of organizations (17%) identified that they experience challenges with requirements management. This confirmed, that despite regulatory changes, the issue of handling requirements when developing medical device software has remained for a number of years without a widespread solution.

A comprehensive mapping study was performing which sought to identify where agile methods have been used in the development of medical device software [10]. This mapping examined publicly available research reports spanning the period of 2002 to 2012. Ten papers were identified which discussed agile methods when developing medical device software. Of those ten, only five related to an implementation of agile practices within a medical device project or organization.

Rasmussen, et al. [34] detail the successful implementation of agile practices within Abbott Diagnostics, the organization recognized the need to move away from a plan-driven approach. In this implementation, Abbott completed two projects side-by-side, one in accordance with agile methods and the other in accordance with a plan-driven approach. While both projects were not the same size, the organization identified that the project completed in accordance with agile methods made a cost saving of between 35% and 50% when compared to the plan-driven project.

Rottier and Rodrigues [35] detailed the implementation of an agile approach within Cochlear. As with Abbott Diagnostics, Cochlear wished to streamline their development process by moving away from a plan-driven approach to a more agile one. However, they quickly identified that it was not possible to wholly adopt a single agile method, such as Scrum or XP on its own, as no single agile method provides sufficient guidance of each of the stages, which are necessary when developing medical device software. This supports the findings of Vogel [36] and Turk, et al. [7], who also identified that no single agile method is sufficiently comprehensive for use when developing medical/safety critical software. However, within Cochlear, it was identified that combining an agile method such as Scrum with a plan-driven SDLC such as the V-Model, sufficient guidance for the development of regulatory compliant software is provided.

Spence [37] discussed the implementation of Scrum within Medtronic. Spence identified that it is not practical to follow a rigid plan-driven approach when developing medical device software, as it is not possible to fully complete one stage of development before moving on to the next, whilst ruling out the need to revisit a stage. The research conducted by Spence is related more to the organizational challenges associated with implementing agile in a medical device software development organization. Further to this, Weyrauch [38] published research on the adoption of agile practices within Medtronic. He builds further on the information presented by Spence, however, the detail he presented remained closer to the organizational impact and accommodation of agile methods, rather than the impact agile practices had on a software development project.

Weiguo and Xiaomin [39] presented the only SDLC which incorporates agile practices with a plan-driven approach which was implemented on a medical device software development project. Unfortunately, the information presented by the authors is very sparse and they do not provide enough guidance as to how their tailored SDLC was implemented should an organization wish to adopt their SDLC. However, they do outline that rather than wholly adopting a single agile method, they retained the V-Model/plan-driven approach to produce the necessary regulatory deliverables.

While the detail included as part of each of these implementations is sparse, commonalities can be identified. Each of the organizations initially examined the possibility of wholly adopting a single agile method such as Scrum or XP. They soon realized

this was not possible and as such they integrated selected agile practices with their traditional plan-driven approach. Furthermore, the selected practices typically originated from either the Scrum or XP approaches.

V. RESEARCH APPROACH

A. Organization Profile

Formed in 2007, BlueBridge Technologies⁴ (BBT) is a high tech product development and design engineering consultancy firm focused on the Medical Device and Life Science market. The company has a dynamic team of 14 individuals with a blend of skill sets in Product Design, Engineering, Life & Physical Sciences, focused on progressing technology and concepts towards viable products.

All members of the BBT team have spent most of their careers in product development. Senior members of the team are an amalgam of returned ex patriots from the U.K., US, Japan, Germany and France. For the most part, the context of their background is in working in close partnership with co-located manufacturing and operational teams on-site. Following this research, BBT have adopted a combined agile and plan-driven approach. Prior to this, it employed a traditional plan-driven approach such as the V-Model.

B. Research Method

The objective of this research was to establish if agile development could resolve the challenges experienced by a medical device organization following a traditional plan-driven approach to answer the following research question:

How can the V-Model for medical device software development be tailored to incorporate agile practices to overcome the challenge of changing requirements?

To that end “Action Research” was employed. In Action Research, the researcher works closely with a group to establish an improvement path for a given situation. In Action Research, the researcher does not perform traditional research, instead acts as a facilitator. This involved four distinct stages:

1. Diagnosing;
2. Planning;
3. Taking Action;
4. Evaluating.

1) Diagnosing

This research began by attempting to identify challenges experienced by BBT. One such challenge identified was the process of accommodating changing requirements. This was established through a survey and semi-structured interviews with key stakeholders within BBT. The problem of accommodating changing requirements was a problem experienced by BBT.

2) Planning

As it emerged at the diagnosing stage that BBT experienced difficulties accommodating changes in requirements, a method to overcome this challenge was required, the decision was made to develop a Hybrid Software Development Life Cycle (SDLC) which integrated agile practices with the traditional V-Model.

Once the hybrid SDLC was developed, the planning stage advanced to training within the organization. Prior to the implementation, training was provided to key members of the development team. This training involved a thorough explanation of the hybrid SDLC and performing a sample medical device software development project in accordance with the hybrid SDLC.

3) Taking Action

At this stage, the hybrid SDLC was implemented within BBT on a medical device software development project. This implementation involved the organization adopting the hybrid SDLC. The Authors performed the role of a consultant for the project. This involved regular updates on the progress of the project and also providing assistance to each member of the development team as and when it was needed. Fortunately, the initial training provided was very comprehensive and the transition from the traditional V-Model to the hybrid model was relatively straightforward.

4) Evaluating

Prior to implementing the hybrid model, BBT reported difficulties accommodating changes when following a traditional plan-driven SDLC. Upon successful implementation of the hybrid model in this research, they identified that this model can accommodate changes much easier than a traditional plan-driven SDLC. From a business perspective, BBT feel the ability to be able to integrate the customers’ requirements at any stage of development could provide them with a competitive advantage over organizations within the medical device software development industry.

C. Data Collection

Following the established guidelines for doing Action Research [40], a plan was established. Data were collected during a number of formal and informal meetings over a period of 18 months. Conducting the research over this period helped to improve validity [41]. Data were obtained from various sources in order to improve triangulation across data sources which helps to

⁴ <http://bluebridgetechnologies.com/>

establish reliability of findings [42, 43]. Sources of data were, surveys and semi-structured interviews with key members of the team (CEO, Marketing Director, Chief Engineer, Senior Developers). The semi-structured interviews were recorded with the participants consent and then transcribed. Notes taken during the interviews were combined with these transcriptions creating an audit trail through memoing. This audit trail was then be used for independent analyses and cross-comparing findings. As a result triangulation was achieved across researchers. This aided in strengthening the research’s validity [43]. The data collected were primarily qualitative.

The qualitative data were analyzed in accordance with Miles and Huberman [44]. With this method, three activities are performed; data reduction, data display, and conclusion and verification drawing. Once the data were collected it was displayed and reduced. Once it was reduced, conclusions were drawn. At this point the data was sent to those who participated in the initial research in BBT. This was a form of member checking, and is recommended for qualitative research [43].

VI. AGILE PRACTICES WITHIN BLUEBRIDGE TECHNOLOGIES

The interviews and surveys performed at the outset of this research established that the primary problem, which BBT experienced when developing medical device software, is accommodating changes in requirements once the requirements management stage was completed. Historically, BBT had an incubation period prior to beginning development, which would hopefully help all stakeholders to discover any potential changes in requirements. This incubation period did assist with identifying some requirements changes however it do not help in identifying emergent and consequential requirements changes, which by their very nature can only be identified once development has begun.

Table 1 Agile Practices selected for Inclusion in the AV-Model

Product Owner	Sprint Backlog
Daily Scrum Meeting	Demo’s after Sprint
Definition of Done	Sprint Retrospective
Product Backlog	Sprint Planning Meeting
Timeboxed Iterations	Taskboard
Burndown Chart	Test Driven Development
Continuous Integration	User Stories/Use Cases
Planning Poker	Face to Face Communication
Scrum Master	Refactoring

The hybrid SDLC adopted by BBT integrates agile practices from both Scrum and XP with the V-Model. The V-Model was chosen as the foundation of the hybrid SDLC for the following reasons:

- V-Model seen as the best fit with regulatory requirements;
- Organization familiar with structure of the V-Model;
- Previous regulatory approval to follow the V-Model.

Both Scrum and XP practices were chosen as they are deemed to be complimentary to each other [45]. Furthermore they have been most widely adopted in the medical device software domain. An analysis was performed of each of the practices contained within these methods. This analysis revealed that none of the practices contained within these methods are directly contradictory to regulatory controls, however as the hybrid SDLC is primarily concerned with facilitate changing requirements, only practices which either have a direct impact on changing requirements, or support practices which do, were included in the hybrid approach. Table 1 shows the practices, which were selected for inclusion in the hybrid approach.

A. Implementation of Hybrid Approach

Figure 3 shows the hybrid SDLC, known as the AV-Model, which integrates the agile practices shown in table 1 with the traditional V-Model. To prove the efficacy in overcoming the challenge of changing requirements, a decision was taken by BBT to implement the hybrid approach on a small medical device development project.

BBT was awarded the contract to develop a Class I, “field use” diagnostics device for the detection and quantitation of antibodies using an enzyme linked immunoassay approach. The technology consisted of an electrochemical biochip incorporated into a fluidics device, which is covered by a deformable membrane. Upon depression of the membrane at specific loci, a sample, together with on-chip reagents are transported to a screen-printed carbon electrode. A specific reaction then occurs, producing an electrochemical signal (current), which is proportional to the concentration of analyte in the sample.

The hand held “reader” component of this technology operates as a standalone unit capable of receiving and interfacing with the credit card size biochip. The product was designed for use by non-technical people and therefore, the ergonomic considerations are important and a very light Human Machine Interface is critical to the products acceptability and error free use in the field.

The activities completed when following the AV-Model are broadly similar to those completed when following the traditional V-Model. The key difference between the two models is the iterative approach taken by the AV-Model. When following the V-Model, the entire software package is seen as single entity, which passes as a whole from each stage of development to the next.

With the AV-Model the software package is divided, firstly into software items and then into software units. Each of these items and units pass singularly through a number of the stages and are integrated late on during development. This iterative approach allows for changes in requirements to be introduced as no single software requirement is completed until the last software item from leaves the implementation stage.

The AV-Model, as with the Scrum approach, requires a number of "Actors" to perform specific duties during the development project. Additional stakeholders and software project management tools support these "Actors".

Table 2 Project Details

Project Identifier	MiCRA Assay System 1328
Project Start Date	August 26 th 2013
Project End Date	December 2 nd 2013
Project Duration	14 Weeks
Team Members	Product Owner, Scrum Master, Software Developers 1,2,& 3
Additional Stakeholder	CEO and Marketing Director
Software Tools in Use	IBM Rational

1) Documentation and Traceability

A key concern for medical device software development organizations is to produce comprehensive documentation detailing the processes, which have been carried out during each stage of development. As part of the MiCRA Assay project this documentation was also required. Due to the iterative approach of the software development, maintaining reliable documentation was an issue. With the traditional V-Model, once a stage is completed, the relevant documentation produced during that stage is also completed. For example, once the Systems Architecture Design Stage has been completed then the Systems Architecture Design Document is also completed. To overcome this issue, a document is opened once the first software item or unit enters that stage. The document remains live until the final software item or unit leaves that stage. This further support the ability to be able to add additional requirements as no single stage is completed until very near the end of the project.

Traceability in medical device software is the ability to be unit. As the software requirements are somewhat fluid, particularly early on in the project this traceability can be difficult to achieve especially if traceability is being performed manually. To

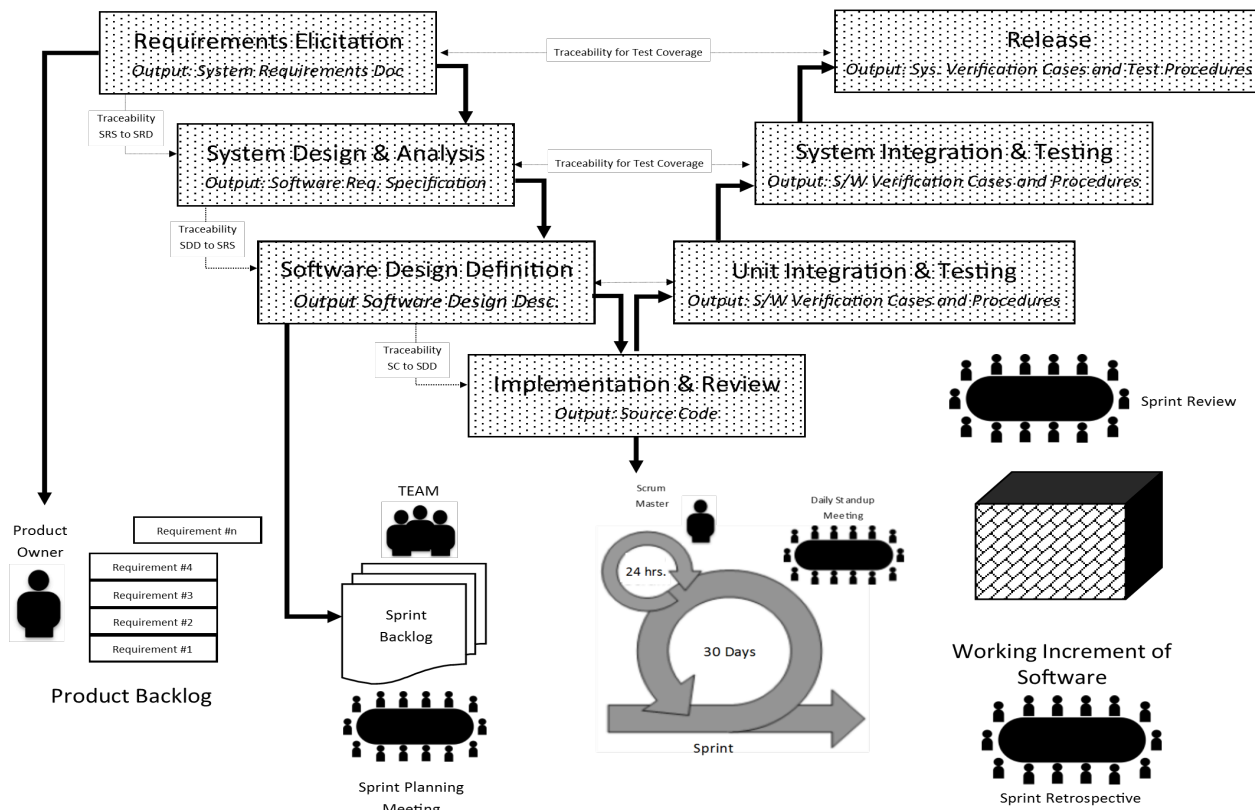


Fig. 3 AV Model approach taken by BBT

overcome this, BBT employed IBM Rational RTC, which successfully traced all of the software items to corresponding requirements, therefore achieving the desired traceability.

2) *Version Control*

In theory, when following the V-Model there is only one version of the software. However, with the AV-Model multiple versions of the software are completed as software items are continually integrated. Open source applications such as Subversion and Jenkins CI have been shown to be very useful to maintain this version control, however as with traceability BBT used IBM Rational RTC package to aid in version control.

VII. DISCUSSION

A. *Findings*

Following the implementation of the hybrid SDLC within BBT, semi-structured interviews were performed with key stakeholders within the organization. The objective of these interviews was to establish if implementing the hybrid SDLC which incorporates agile practices, help the organization to overcome the challenges identified previously. The data collected at these interviews were primarily qualitative. The primary challenge identified by the BBT was the inability to accommodate changes in requirements once development had begun, when following a plan-driven approach. Each of the stakeholders identified that they are now better equipped to accommodate changes in requirements at any point during a project. To accompany this BBT did not have to perform the same amount of up-front planning, as they would have had to if they had completed the same project in accordance with the V-Model. The reason cited for this is due to the stakeholders being confident in the hybrid approach, which allowed for changes at any point. The stakeholders were asked to compare the project implemented in accordance with AV-Model with similar projects completed in accordance with the V-Model. The stakeholders identified that should they have identified the same consequential changes in requirements the project duration would have overrun by approximately 14% and the budget would have overrun by approximately 7% if they had of been following the V-Model.

B. *Limitations*

A number of limitations of this research were identified. These were classified following the common classifications: reliability; construct validity; and external validity. As this research was performed solely within a single organization, no casual relationships were established, resulting in no discussion surrounding internal validity.

1) *Reliability*

Reliability is concerned with whether or not the study would yield the same results if performed again under the same research criteria. An example of this is a Systematic Literature Review. If the results or findings of the research are not repeatable under the same conditions, then an external influence such as research bias could have unduly affected the initial results [46]. Reliability was achieved as part of this research by employing several practices such as prolonged involvement i.e. 18 months, with the organization, data triangulation, member checking and using the recordings and transcripts to form an audit trail.

2) *Construct Validity*

Construct validity focuses on whether theoretical constructs are interpreted and measured correctly [46]. For example, if a researcher is using two methods and establishes one method is better than the other, will the researcher interpret “better” in the same way as other researchers? The mapping study outlined previously was conducted in accordance with rigorous guidelines. These guidelines ensured that another researcher could replicate the findings. Furthermore the validation portion of these research employed triangulation to ensure that each method of validation e.g. implementation, was supported by the other validation performed.

3) *External Validity*

External Validity also known as Transferability is focused upon whether or not claims of generality of the results of a research project are justified [46]. Transferability is achieved if a reader is confident about the research setting, the description and that the project is sufficiently detailed [47]. This information provides the reader context, and based upon this context the reader can decide whether or not the findings are transferable to their own research or interests. The focus of this research was to establish “if” agile practices could be adopted to overcome the challenges associated with developing regulatory compliant software. As a result this research was deemed exploratory. While a number of regulated domains exist i.e. automotive, avionics, nuclear etc. this research focused on the medical device software domain. The hybrid approach shown in figure 3 can be of use to other organizations in the same domain experiencing the same challenges as those in BBT.

VIII. CONTRIBUTION TO RESEARCH

The AV-Model integrates agile practices with the traditional V-Model. The V-Model was adopted as the foundation to the AV-Model for a number of reasons. Following the V-Model produces the necessary deliverables required when seeking regulatory approval and medical device software development organizations may be reluctant to adopt a SDLC wholly different to the V-Model, as they may have received regulatory approval to use the V-Model. The agile practices integrated within the V-

Model aim to facilitate handling changes in requirements throughout a software development project. The AV-Model was developed in collaboration with a number of medical device software development organizations.

Prior to this research being conducted, BBT wanted to be able to better meet their customers' needs. This involved being able to accommodate changes in requirements at any point in the software development project. Based upon the research presented here, BBT decided to implement the AV-Model and found it very beneficial. Prior to implementing the AV-Model, BBT followed a plan driven SDLC i.e. the V-Model. After successful implementation of the AV-Model, BBT confirmed that they are now better equipped to be able to accommodate changes in requirements, which not only helps them to better meet their customers' needs, but also improves the development process as often changes in requirements are identified by the development team as a project progresses. The AV-Model was implemented on a small medical device project. As part of the development of the software for this device, five different types of requirements changes were accommodated by the AV-Model during development i.e. Mutable, Consequential, Emergent, Functional and Non-Functional. As a result, it can be said that the AV-Model has the potential to be adopted on a small medical device software development project where requirements can change.

To accompany this, experts in the field of medical device software development validated the AV-Model from a theoretical perspective. These experts concluded that theoretically, the AV-Model could be followed by any medical device software development organization, that it produces the necessary deliverables as part of IEC 62304, and that it could assist in accommodating changes in requirements at any point in a medical device software development project.

Medical device software development organizations will benefit from this research as it gives them an insight into the practices performed within other medical device software development organizations. This research was conducted in collaboration with a number of medical device software organizations. Despite the identity of participants being kept confidential, medical device organizations can benefit from seeing industry trends and approaches.

IX. CONCLUSIONS AND FUTURE WORK

While this research sets out to solve challenges associated with following a plan-driven approach when developing medical device software through the adoption of agile practices, it by no means provides a panacea to all of the challenges experienced by medical device software organizations. This research identified that BBT experience challenges accommodating changes in requirements when following a plan-driven approach. This is a challenge often experienced when following a plan-driven approach.

To date there is little publicly available information detailing the successful implementation of agile approaches in the medical device software. While this paper focuses solely on the implementation of agile practices on a single project within a single organization, the findings do help to increase the confidence in the efficacy of agile practices when developing medical device software.

Prior to this research being conducted, BBT wanted to be able to better meet their customers' needs. This involved being able to accommodate changes in requirements at any point in the software development project. Based upon the research presented here, BBT decided to implement the AV-Model and found it very beneficial. Prior to implementing the AV-Model, BBT followed a plan driven SDLC i.e. the V-Model. After successful implementation of the AV-Model, BBT confirmed that they are now better equipped to be able to accommodate changes in requirements, which not only helps them to better meet their customers' needs, but also improves the development process as often changes in requirements are identified by the development team as a project progresses. The AV-Model was implemented on a small medical device project. As part of the development of the software for this device, five different types of requirements changes were accommodated by the AV-Model during development i.e. Mutable, Consequential, Emergent, Functional and Non-Functional. As a result, it can be said that the AV-Model has the potential to be adopted on a small medical device software development project where requirements can change.

This research does not seek to create generalizations which state that agile practices can be used in any medical device software organization, however, it can be stated that in organizations of similar structure to BBT and on projects similar to that of the MiCRA Assay project, employing the hybrid approach detailed as part of this research could be advantageous.

Through the successful implementation of the hybrid approach on the MiCRA Assay project BBT have decided to implement this approach again on their next larger project. BBT is the project leaders for this project and it is expected that it will take a number of years to complete. Upon successful completion of this project confidence surrounding the hybrid approach will increase and as a result it is expected than an increased adoption of this approach will be seen.

ACKNOWLEDGMENT

This research is supported by the Science Foundation Ireland (SFI) Stokes Lectureship Programme, grant number 07/SK/I1299, the SFI Principal Investigator Programme, grant number 08/IN.1/I2030 (the funding of this project was awarded by Science Foundation Ireland under a co-funding initiative by the Irish Government and European Regional Development Fund), and supported in part by Lero - the Irish Software Engineering Research Centre (<http://www.lero.ie>) grant 10/CE/I1855.

REFERENCES

- [1] M. Cohn, *Succeeding with Agile - Software Development Using Scrum*. Upper Saddle River NJ: Addison Wesley, 2011.
- [2] R. C. Martin, *Agile Software Development - Principles, Patterns and Practices*. Upper Saddle River, NJ: Prentice Hall, 2003.
- [3] W. Royce, "Managing the Development of Large Software Systems," presented at the Proceedings of IEEE WESCON, 1970.
- [4] VersionOne, "8th Annual State of Agile Survey: The State of Agile Development," 2013.
- [5] VersionOne, "7th Annual State of Agile Survey: The State of Agile Development," 2012.
- [6] VersionOne, "6th Annual State of Agile Survey: The State of Agile Development," 2011.
- [7] D. Turk, R. France, and B. Rumpe, "Limitations of Agile Software Processes," presented at the Third International Conference on Extreme Programming and flexible processes in Software Engineering (XP2002), Alghero, Sardinia, 2002.
- [8] A. Shatil, O. Hazzan, and Y. Dubinsky, "Agility in a Large-Scale System Engineering Project: A Case-Study of an Advanced Communication System Project," presented at the Proceedings of the 2010 IEEE International Conference on Software Science, Technology & Engineering, 2010.
- [9] M. Kircher, P. Jain, A. Corsaro, and D. Levine, "Distributed extreme programming," *Extreme Programming and Flexible Processes in Software Engineering, Italy*, pp. 66-71, 2001.
- [10] M. McHugh, O. Cawley, F. McCaffery, I. Richardson, and X. Wang, "An Agile V-Model for Medical Device Software Development to Overcome the Challenges with Plan-Driven Software Development Lifecycles," presented at the Software Engineering in Healthcare (SEHC) workshop at the 35th International Conference on Software Engineering (ICSE), San Francisco CA, 2013.
- [11] European Commission, "Council Directive 93/42/EEC of 14 June 1993 concerning medical devices," ed, 1993.
- [12] European Council, "Directive 2007/47/EC of the European Parliament and of the Council of 5 September 2007," European Council, EU Bookshop2007.
- [13] M. McHugh, F. McCaffery, and V. Casey, "Standalone Software as an Active Medical Device " presented at the The 11th International SPIE Conference Process Improvement and Capability dTermination, Dublin, 2011.
- [14] H. Takeuchi and I. Nonaka, "The new new product development game," *Harvard business review*, vol. 64, pp. 137-146, 1986.
- [15] K. H. Pries and J. M. Quigley, *Scrum project management*. London: CRC Press, 2010.
- [16] F. Paetsch, A. Eberlein, and F. Maurer, "Requirements engineering and agile software development," presented at the Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on, 2003.
- [17] S. Ambler and M. Lines, *Disciplined Agile Delivery*. Upper Saddle River, New Jersey: Pearson, 2012.
- [18] S. Miura, "Industry Viewpoint on software," presented at the 11th Conference of the Global Harmonization Task Force, Washington, 2007.
- [19] ISO, "ISO/IEC 12207 Information technology -- Software life cycle processes," International Organisation for Standards, Geneva, Switzerland1995.
- [20] ISO, "ISO 12207:1995/AMD1:2002 Information technology - Software life cycle processes AMENDMENT 1," International Standards Organisation, Geneva Switzerland2002.
- [21] ISO, "ISO 12207:1995/AMD2:2004 Information technology - Software life cycle processes AMENDMENT 2," International Standards Organisation, Geneva Switzerland2004.
- [22] ISO, "ISO/IEC 14971:2007 Medical devices -- Application of risk management to medical devices," International Organisation for Standards, Geneva, Switzerland 2007.
- [23] ISO, "ISO/IEC 15288:2008 Systems and software engineering -- System life cycle processes," International Organisation for Standards, Geneva, Switzerland2008.
- [24] European Council, "Commission communication in the framework of the implementation of the Council Directive 93/42/EEC of 14 June 1993 concerning medical devices (Publication of titles and references of harmonised standards under the directive)," European Council, EU Bookshop2010.
- [25] C. Gerber. (2008). *Introduction into IEC 62304 Software life cycle for medical devices*. Available: <http://www.spiq.com/abs/JF200809IEC62304 SPIQ Rev004.pdf>
- [26] M. Klumper and E. Vollebregt, "The Regulation of Software for Medical Devices in Europe," *Journal of Medical Device Regulation*, vol. 7, pp. 5-13, 2010.
- [27] European Council, "Directive 2007/47/EC of the European Parliament and of the Council of 5 September 2007 amending Council Directive 90/385/EEC on the approximation of the laws of Member States relating to the active implantable medical devices, Council Directive 93/42/EEC concerning medical devices and Directive 98/8/EC concerning the placing of biocidal products on the Market. Official Journal of the European Union, 2007, L247, 21 (21 September 2007)." European Council, EU Bookshop2007.

- [28] P. T. H. Kim, "FDA and the regulation of medical software," presented at the Sixth Annual IEEE Symposium on Computer-Based Medical Systems, Michigan, 1993.
- [29] FDA, "21 CFR Part 880 Medical Devices; Medical Device Data Systems Final Rule.," *Federal Register* vol. 76, pp. 8637 - 8649, 2011.
- [30] *Food and Drug Cosmetics Act*, 1968.
- [31] C. Denger, R. L. Feldman, M. Host, C. Lindholm, and F. Schull, "A Snapshot of the State of Practice in Software Development for Medical Devices," presented at the First International Symposium on Empirical Software Engineering and Measurement, 2007. ESEM 2007, Madrid, 2007.
- [32] Embedded Forecasters, "Embedded Market Forecasters Survey (2010)," Embedded Market Forecasters, Ashland MA2010.
- [33] M. McHugh, F. McCaffery, and V. Casey, "Barriers to Adopting Agile Practices when Developing Medical Device Software," presented at the The 12th International SPICE Conference Process Improvement and Capability dEtermination, Palma, Majorca, 2012.
- [34] R. Rasmussen, T. Hughes, J. R. Jenks, and J. Skach, "Adopting Agile in an FDA Regulated Environment," presented at the Agile Conference, 2009 AGILE '09 Chicago, IL 2009.
- [35] P. A. Rottier and V. Rodrigues, "Agile Development in a Medical Device Company," presented at the Proceedings of the 11th AGILE Conference. AGILE '08., Toronto, 2008.
- [36] D. Vogel, "Agile Methods: Most are not ready for prime time in medical device software design and development," *DesignFax Online*, vol. 2006, 2006.
- [37] J. W. Spence, "There has to be a better way! [software development]," presented at the Proceedings to Agile Conference, 2005. , Denver, 2005.
- [38] K. Weyrauch, "What Are We Arguing About? A Framework for Defining Agile in our Organization," presented at the Proceedings of the conference on AGILE 2006, 2006.
- [39] L. Weiguo and F. Xiaomin, "Software Development Practice for FDA-Compliant Medical Devices," presented at the International Joint Conference on Computational Sciences and Optimization, 2009. , Sanya, Hainan, 2009.
- [40] S. Kemmis, R. McTaggart, and J. Retallick, "The action research planner," 2004.
- [41] O. Eikeland, "The validity of action research-validity in action research," 2006.
- [42] J. McNiff, *Action research: Principles and practice*: Routledge, 2013.
- [43] J. W. Creswell and D. L. Miller, "Determining validity in qualitative inquiry," *Theory into practice*, vol. 39, pp. 124-130, 2000.
- [44] M. Miles and A. M. Huberman, *Qualitative Data Analysis: An Expanded Sourcebook*. London: Sage, 1994.
- [45] B. Fitzgerald, G. Hartnett, and K. Conboy, "Customising agile methods software practices intel shannon," *European Journal of Information Systems*, vol. 15, pp. 200-213, 2006.
- [46] S. Easterbrook, J. Singer, M. A. Storey, and D. Damian, "Selecting Empirical Methods for Software Engineering Research," in *Guide to Advanced Emperical Software Engineering*, ed London: Springer, 2008.
- [47] C. Seal, *The Quality of Qualitative Research*. Thosuand Oaks CA: Sage Publications, 2000.