Standalone Software as an Active Medical Device

Martin McHugh, Fergal McCaffery, and Valentine Casey

Regulated Software Research Group, Dundalk Institute of Technology & Lero,
Dundalk Co. Louth, Ireland
{Martin.McHugh,Fergal.McCaffery,Val.Casey}@dkit.ie

Abstract. With the release of the latest European Medical Device Directive (MDD) standalone software can now be classified as an active medical device. Consequently the methods used to ensure device safety and reliability needs to be reviewed. IEC 62304 is the current software development lifecycle framework followed by medical device software developers but important processes are beyond the scope of IEC 62304. These processes are covered by additional standards. However since the MDD became mandatory these additional standards are not comprehensive enough to ensure the reliability of an active medical device consisting of only software. By employing software process improvement techniques this software can be developed and validated to ensure it performs the required task in a safe and reliable way.

Keywords: Medical Device Standards, IEC 62304, MDD (2007/47/EC), Software Process Improvement.

1 Introduction

The use of technology within healthcare is on the rise particularly [1] with the advent of healthcare software applications for use with smartphones such as “Medscape” and “Radiology 2.0” for the Apple iPhone. The iTunes App Store has a section containing over two hundred and thirty applications for use within healthcare [2]. Failures in software used within healthcare can have costly and deadly consequences. This occurred in 2000 when twenty one Panamanian teletherapy patients received lethal doses of radiation therapy due to faulty software [3]. The Food and Drugs Administration (FDA) record all product recalls and in the period from 1st November 2009 to 1st November 2010 seventy eight devices were recalled due to software related problems [4].

An increasing number of tasks within the medical profession are being transferred to automated software driven devices. This can be seen in USB blood glucose meters, where traditionally the measurement was taken by a clinician and the results manually recorded either via pen and paper or entered into an electronic health record (EHR), whereas now the sample is placed on the USB glucose meter and the device automatically records the results and once connected to a computer automatically updates the patient’s EHR [5].

Medical devices intended for use within the European Union must have a CE conformance mark [6]. To achieve this conformance mark audits are performed on these devices to ensure their safety and reliability by notified bodies within each country. Within the Republic of Ireland the National Standards Authority of Ireland (NSAI) is responsible for ensuring conformity before awarding a CE mark. These devices typically needed to satisfy standards which include: EN ISO 13485:2003 (medical device quality management standard) [7], EN ISO 14971:2009(medical device risk management standard) [8] and the medical device product level standard IEC 60601-1 [9, 10]. The original directive (MDD 93/42/EEC) historically defined a medical devices as being hardware with or without a software element [11]. However since the enforcement of European Medical Device Directive (MDD 2007/47/EC) in March 2010 [12], standalone software can now be classified as an active medical device [13] and consequently the standards used to ensure conformance to the CE mark need to be reviewed and if necessary amended. As a result of this update to the MDD, medical device software is required to be developed through adopting best practice software development practices. This essentially means adhering to the medical device software lifecycle process standard IEC 62304 [14] and the set of aligned medical device standards and technical reports e.g. IEC 62366 [15] and IEC TR 80002-1 [16].

Software process improvement (SPI) is not a new concept but is becoming increasingly important in the area of medical device software development. The ISO 15504-5 [17] standard also known as SPICE (Software Process Improvement Capability dEtermination) is recognised as a source of best practices for software development projects. SPICE was not developed for any specific sector of the software industry so it is general in its approach.

IEC62304:2006 is a software development lifecycle for use within the medical device software development domain and is derived from the generic software lifecycle process ISO 12207 [18] [19]. IEC 62304 is a harmonised standard since November 2008 with the following European Council medical device standards: MDD (1993/42/EEC); AIMD (1990/385/EEC); and IVDD (1998/79/EC) [20]. Within this paper we examine IEC 62304:2006 to determine if all the requirements of the MDD (2007/47/EC) are satisfied and if not, what framework must be applied in order for software development projects to meet the CE conformance requirements.

The remainder of this paper is structured as follows:

In section 2, the revision of the MDD (2007/47/EC) is examined to see what affect this amendment to the MDD (1993/42/EEC) has on the classification of a medical device and how this classification effects the development of medical device software. Section 3 discusses the importance of SPI. Additionally the history of IEC62304:2006 is analysed and how it is has evolved from the ISO 12207 and SW68 standards [19] along with what processes are included in IEC 62304:2006. Section 4 examines the existing medical device software development standard IEC 62304:2006 to determine if it is comprehensive enough to satisfy the MDD amendment in relation to standalone software now being defined as an active medical device. Section 5 discusses practices beyond the scope of the IEC62304:2006 standard that are required to satisfy the definition of software in the MDD (2007/47/EC). Additionally, we determine whether missing practices may be resolved by amending or extended, the existing processes or if there is a need to develop a new standalone medical device software lifecycle standard. Finally section 6 provides the conclusions from this research and plans to progress this work further.

2 Medical Device Directive (2007/47/EC)

All medical devices intended for use within the European Union must conform to the current MDD. The MDD (2007/47/EC) is the current directive and was released on October 11th 2007. However it only became mandatory for CE compliance on March 21st 2010 [21]. The MDD (2007/47/EC) is harmonised 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 [13]:

*"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 above definition as to what constitutes a medical device, standalone software can be considered a medical device. Whilst MDD (1993/42/EEC) did allow for software to be seen as a medical device [11] it did not extend to standalone software being recognised as an active medical device. MDD (2007/47/EC) Annex IX Section 1.4 defines an active medical device as [13]:

"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

2.1 Classification Rules

Annex IX section III (Classification) within the MDD (2007/47/EC) categorizes medical devices into one of four categories [13]:

- Class I devices are non-invasive devices unless they are used for the purpose of channelling blood or tissue or unless they are intended for use on wounds which have the dermis breached and can only heal by secondary intent, e.g. wheelchairs, bandages, incontinence pads. Also invasive devices that are not connected to an active medical device are classified as Class I e.g. tongue depressor.
- Class IIa devices are surgically invasive devices for transient use unless they control, diagnose, monitor or correct a defect of the heart of central circulatory system e.g. transfusion equipment, storage and transport of donor organs.

- Class IIb devices are surgically invasive devices which are implantable or intended for long term use unless they come into contact with the heart e.g. haemodialysis, dressings for chronic extensive ulcerated wounds.
- Class III devices are used for supporting or sustaining life and devices which potentially pose an unreasonable risk of illness or injury e.g. pacemakers and heart valves.

This classification is based on the risk to the patient's safety, ranging from low risk to high risk. The higher the risk to the patient's safety, the greater the level of assessment required to achieve the CE conformance mark. If software is part of a medical device it assumes the classification of the overall device. If standalone software is an active medical device then the device is classified based on the risk the device places on the patient or a third party according to MDD (2007/47/EC) Annex IX Section III [13].

The MDD (2007/47/EEC) has wider reaching consequences, devices that historically were not classified as medical devices and not subject to conformance standards are now being classified as medical devices. This occurs when the device is connected to an active medical device. An example of this is in the visual display unit (VDUs) that display results from a medical device are now classified as being a medical device [22].

3 Software Process Improvement & IEC 62304:2006

3.1 Software Process Improvement

SPI is an important element of any software development project. SPI is a continuous cyclic path of improvement by performing assessments, implementing recommendations of those assessments and beginning the cycle again [23]. All software development projects are a series of processes. The processes need to be understood and improved where possible. There are four primary objectives of SPI [24]:

- To attain an understanding of current software development practices;
- Select areas to focus on to achieve the greatest long term benefits;
- Add value to the organisation developing the software rather than solely to a specific development project;
- To grow by combining effective processes with skilled and motivated people.

Private industry has greatly benefited from SPI. Hughes Aircraft invested $400,000 to develop software process improvement within their company. This investment resulted in an annual saving of $2,000,000 to the company. Similarly safety is improved by employing SPI practices. An improvement process was undertaken by the group that develops on-board software for the Space Shuttle at IBM Huston. Early detection rate of errors rose from 48% to 80% as a result of using SPI [25].

A recent survey carried out by Embedded Market Surveys analysed software development projects in both embedded industry and the medical device industry [26]. The survey found that projects within embedded industry on average over run by 47%

whilst software development projects within medical device production over run by 54.4%. The survey also found that 9.7% of medical device projects were cancelled with the reasons being cited:

- Incomplete or vague requirements;
- Insufficient time;
- Insufficient resources;
- Design Complexity.

These issues can be overcome by employing an effective software development process which can then reduce project over runs, cost can be reduced and the number of projects cancelled can be decreased. SPI frameworks such as ISO 15504-5, and CMMI® [27] and international standards such as ISO\IEC 12207 or IEC 62304 provide guidance on how to help address all of the above areas either directly or by using normative reference to additional standards.

3.2 IEC 62304:2006 Medical Device Software – Software Lifecycle

IEC 62304:2006 is derived from ISO/IEC 12207, by the Association of Advancement of Medical Instrumentation (AAMI) and from the American National Standards ANSI/AMMI SW68:2001 [19].

Whilst ISO/IEC 12207 is not domain specific it is seen as being very comprehensive in its approach and this is reflected from the standards utilising the core principle of ISO/IEC 12207 e.g. ISO IEC 15504-5:2006 and ISO/IEC 90003. IEC 62304 was developed between 2002 and 2006 by the joint working group ISO/TC 210 and IEC Sub-Committee 62A [28]. IEC 62304 was created as a software development standard for lifecycle processes for the purpose of safe design and maintenance of medical device software.

Software developed using the IEC 62304:2006 standard is founded upon the assumption that the software is developed in accordance with a quality management standard (ISO 13485:2006), a risk management standard (ISO/IEC 14971) and a product level standard (EN 60601-1) [9].

IEC 62304:2006 section 3.24 defines a Software Lifecycle Model as [14]:

A conceptual structure spanning the life of the software from definition of its requirements to its release for manufacturing which:

- ➢ *Identifies the process, activities and tasks involved in development of a software product*
- ➢ *describes the sequence of and dependency between activities and tasks*
- ➢ *identifies the milestones at which the completeness of specified deliverables is verified*

This definition is based on the definition within ISO/IEC 12207:1995, definition 3.11 [18].

This standard provides a framework of process with activities and tasks. A process is divided into activities and the activities are further divided into tasks. The processes within IEC 62304:2006 for the development of software for medical devices are [14]:

- Quality Management System
- Software Safety Classification
- Software Development Process which includes;
  - Software Development Planning
  - Software Requirements Analysis
  - Software Architectural Design
  - Software Detailed Design
  - Software unit implementation and verification
  - Software Integration and testing
  - Software system testing
  - Software Release
- Software Maintenance Process
- Risk Management Process
- Software Configuration Management Process
- Software Problem Resolution Process

These processes are represented graphically in Figure 1 and it can be seen how these processes fit into IEC 62304:2006.

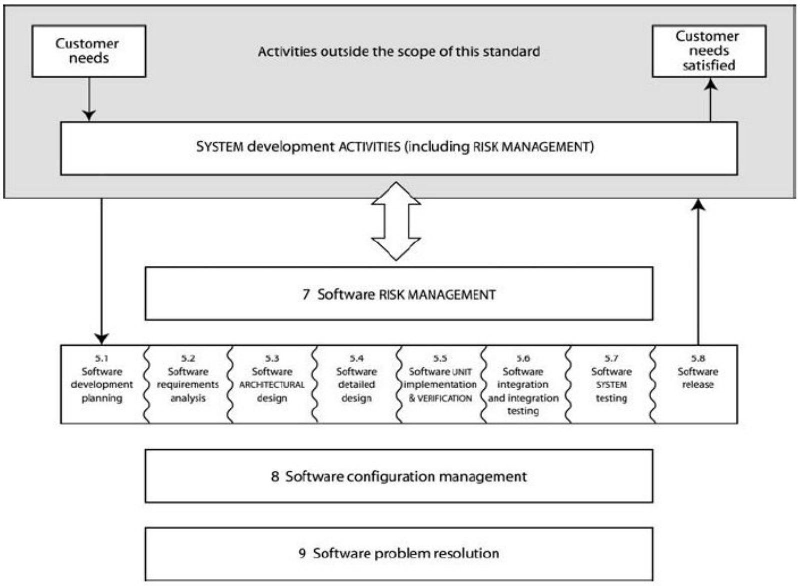
Customer needs
Activities outside the scope of this standard
Customer needs satisfied
SYSTEM development ACTIVITIES (including RISK MANAGEMENT)
7 Software RISK MANAGEMENT
5.1 Software development planning
5.2 Software requirements analysis
5.3 Software ARCHITECTURAL design
5.4 Software detailed design
5.5 Software UNIT implementation & VERIFICATION
5.6 Software integration and integration testing
5.7 Software SYSTEM testing
5.8 Software release
8 Software configuration management
9 Software problem resolution

Fig. 1. Overview of Software Development Processes and Activities [14]

Within IEC 62304:2006 Section 4.3 [14], software is classified according to the severity of potential harm, into one of three categories similar to that of the medical device classification:

- Class A: No injury or damage to health is possible
- Class B: Non-serious injury is possible
- Class C: Death or Serious Injury is possible

These classifications are subject to the medical device risk management standard ISO 14971:2007. As the risk management process is covered by ISO 14971:2007, IEC 62304:2006 makes normative reference to it. While making some additions which are required for the identification of software factors related to hazards.

Safety critical standalone software systems can be divided into items running a different software element each with its own safety classification. These items can be further sub divided into additional software elements. The overall software system assumes the highest classification contained within all of the software elements. For example if a software system contains five software elements, four of which may be classified as Class A, but one may be classified as Class C and therefore the overall device receives a classification of Class C [29]. This can be seen in figure 2.

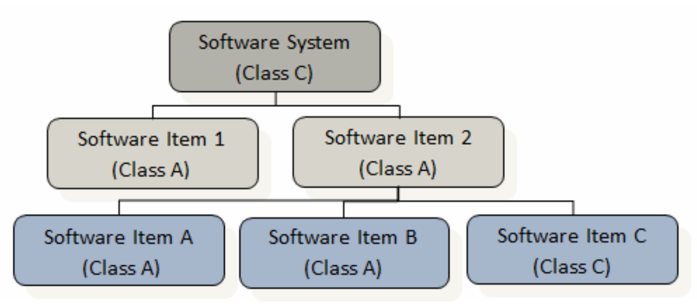
Software System (Class C)
Software Item 1 (Class A)
Software Item 2 (Class A)
Software Item A (Class A)
Software Item B (Class A)
Software Item C (Class C)

Fig. 2. Classification of software items within complete software system

4 Problems with Existing Conformance Standards

As discussed in section two, the definition of what constitutes a medical device has changed under MDD (2007/47/EC), however the methods used to test these devices has not changed in response to the latest directive amendment.

The current standard IEC 62304:2006 operates in conjunction with additional standards such as EN ISO 14971:2007 and EN ISO 13485:2003 to ensure overall conformance of the device. A number of processes have explicitly been defined as being beyond the scope of IEC 62304:2006

Figure 1 shows that there are processes beyond the scope of IEC 62304:2006 such as:

- Customer needs (Requirements elicitation)
- System Development Activities including risk management
- Customer needs satisfied (Validation and Final Release)

IEC 62304 was developed prior to MDD (2007/47/EC) and relies on additional standards to ensure the safety and reliability of the overall system. For this standard to be seen as a comprehensive method of ensuring the safety and reliability of standalone software as an active medical device these processes need to be brought into the scope.

Processes beyond the scope of IEC 62304 prior to MDD (2007/47/EC) were performed within other standards, but these standards do not address the specific aspects relating to developing standalone software as an active medical device. Essentially IEC 62304 was designed for medical device software that would always be part of a traditional medical device system (i.e. hardware and software) and therefore the focus was only on the software component and the system level processes and practices are not addressed. However, as the MDD defines standalone software as an active medical device the medical device system consists solely of software. Therefore to meet the requirements of the revised MDD system level processes and practices need to be handled and this is not currently possible using IEC 62304 in isolation.

5 Resolving the Problems

Research performed by the authors as (part of the Regulated Software Research Group (RSRG)) in Dundalk Institute of Technology (DkIT) has identified the following list of processes that need to be completed to ensure the development of safe and reliable software as an active medical device:

- Software System Project Management Planning;
- Software System Requirements Elicitation;
- Software System Requirements Analysis;
- Software System Architectural Design;
- Software System Design;
- Software System Construction;
- Software System Integration;
- Software System Testing;
- Software System Release;
- Software System Installation;
- Software System Maintenance.

These processes correspond to the key stages required for medical device software development. Each process has been designed to address the requirements of developing software as a standalone system. The following supporting processes have also been identified:

- Validation;
- Configuration Management;
- Change Request Management;
- Problem Resolution Management.

IEC 62304:2006 incorporates the majority of these processes. Processes not covered by IEC 62304:2006 include: Software System Requirements Elicitation; System Installation; Validation; and System Final Release (Software System Release Process only deals with software Release and not final release). The processes not covered by IEC 62304:2006 are however covered by ISO/IEC 15504-5 but as discussed ISO/IEC 15504-5 is not domain specific. Consequently, the processes not included within IEC 62304:2006 need to be reviewed and tailored to suit the needs of software being developed as an active medical device.

The processes outside of the scope of IEC 62304:2006 prior to the release of MDD (2007/47/EC) were covered by standards such as EN ISO 13485:2003, ISO 14971:2007 and EN 60601-1:1990 etc. Currently within IEC 62304:2006 there is no method of validating a complete system that only incorporates software. This results in the standard not being capable of ensuring the safety and reliability of software that is an active medical device.

Device manufacturers within other industries have had similar problems and as a result they developed standards which meet their needs and strove towards best practice rather than simply reaching conformance requirements. These include Automotive Spice and Spice for Space, both of which allow for process assessment and provide an improved software development path [23]. Similarly within the medical device industry Medi SPICE is being developed to provide a basis for medical device software process assessment and improvement. Medi SPICE aims to provide developers of medical device software with a complete lifecycle to develop medical device software. This includes conformance with regulatory requirements and the processes beyond the scope of IEC 62304. It is therefore envisaged that it can be used to help achieve CE compliance of standalone software as an active medical device. The work we present in this paper is being factored into Medi SPICE so that both embedded medical device software development and standalone medical device software will be covered [30].

6 Conclusions

For a medical device manufacturer to be successful, it must produce a safe, reliable device that conforms to the regulatory standards of the market into which they are selling their devices i.e. MDD (2007/47/EC) within the European Union. Upon achieving conformance medical device manufactures can market and sell their products within a particular market. Conformance to these standards ensures that only safe and reliable products are used. The MDD (2007/47/EC) has incorporated a number of changes in comparison to MDD (1993/42/EEC). In terms of this paper the most significant change within MDD (2007/47/EC) is the ability of software to be now seen as an active medical device. But since this amendment the question of whether or not the existing standards used to regulate this software are sufficient must be answered.

IEC 62304:2006 is the current standard used by software developers developing software for medical devices. This standard is part of the harmonised standards within MDD (2007/47/EC). IEC 62304:2006 provides a framework of lifecycle processes. However there are important system processes that are beyond the scope of IEC 62304:2006

The processes outside of the scope of IEC 62304:2006 are primarily system processes which reference ISO/IEC 12207. As ISO/IEC 12207 is a generic software lifecycle rather than being domain specific it can be considered broad and does not fully cater for the needs of medical device software development. These processes need to be tailored to suit the needs of medical device software. This will be achieved through the release of Medi SPICE.

Acknowledgements. 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 03/CE2/I303_1.

References

1. Blacksmith Enterprise, `http://www.blackenterprise.com/2010/09/30/healthcare-it-on-the-rise/`
2. PCMag.com, `http://www.pcmag.com/article2/0,2817,2343550,00.asp`
3. Borrás, C.: Overexposure of Radiation Theraphy in Panama. Articles and special reports, 173–187 (2006)
4. U.S. Department of Health & Human Service, `http://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfres/res.cfm`
5. Diabetes. Bayer Contour USB Glucose Meter (2010), `http://www.diabeticlive.com/glucose-meters/bayer-contour-usb-glucose-meter/`
6. British Standards Online (America), `http://www.bsiamerica.com/en-us/Sectors-and-Services/Industry-sectors/Healthcare-and-medical-devices/CE-marking-for-medical-devices/`
7. EN ISO 13485:2003. Medical Device: Quality Management Systems. Requirements for the Regulatory Process, July 24 (2003)
8. EN ISO 14971:2009. Medical Devices. Application of Risk management to medical devices, July 31 (2009)
9. EN 60601-1. Medical Electrical Equipment. General requirements for basic safety and essential performance. Collateral standard. Usability, May 31 (2010)
10. Emergo Group, `http://www.bsiamerica.com/en-us/Sectors-and-Services/Industry-sectors/Healthcare-and-medical-devices/CE-marking-for-medical-devices`
11. European Council, Council Directive 93/42/EEC Concerning Medical Devices, June 14 (1993)
12. Webb, K.: Changes to the Medical Device Legislation in Europe- The Impact of Directive, 2007/47/EC (2007), `http://www.slideshare.net/mediqol/changes-to-the-medical-device-legislation-in-europe-presentation`
13. European Council 2007, Council Directive 2007/47/EC (September 2007)

14. ANSI/AAMI/IEC 62304. Medical device Software - Software life cycle processes. Association for the Advancement of Medical Instrumentation, July 19 (2006)
15. IEC 62366. Medical Devices - Application of usability engineering to medical devices, November 13 (2007)
16. AAMI/IEC TIR 80002-1:2009. Medical Device Software 1: Guidance on the application of ISO 14971 to Medical Device Software, May 31 (2010)
17. ISO/IEC 15504-5:2006. Information Technology. Process Assessment. An exemplar process assessment model, April 28 (2006)
18. ISO/IEC 12207:1995. Information Technology. Software Lifecycle Processes, February 28 (1995)
19. Qmed, http://www.qmed.com/consultants/19204/global-perspectives-north-america-iec-62304-questions-and-answers?quicktabs_5=1
20. Eagles, S.: International Standards and EU regulation of medical device software – An update (2009)
21. Emergo Group, http://www.emergogroup.com/newsletters/directive-2007-47-ec-sep2007
22. ComputerWorld. FCC mobile network plan could revolutionize health care (2010), http://www.computerworld.com/s/article/9174429/FCC_mobile_network_plan_could_revolutionize_health_care
23. McCaffery, F., Coleman, G.: The need for a software process improvement model for the Medical Device Industry. In: International Review on Computers and Software, vol. 2, pp. 10–15 (2007)
24. Wiegers, K.: Software Process Improvement: Ten Traps to avoid, http://www.processimpact.com
25. Humphrey, W.S.: Introduction to Software Process Improvement. Technical Report CMU/SEI-92-TR-007 ESC-TR-92-007
26. Embedded Forecasters - Embedded Market Forecasters Survey (2010)
27. CMMI Product Team, Capability Maturity Model® Integration for Development Version 1.2, Software Engineering Institute (2006)
28. Miura, S.: Industry View Point on software In: 11th Conference of the Global Harmonisation Task Force (2007)
29. Gerber, C.: Introduction to IEC 62304 Software Life cycle for medical devices, September 4 (2008)
30. McCaffery, F., Dorling, A.: Medi SPICE: An Overview. Journal of Software Maintenance and Evolution: Research and Practice 22, 255–267 (2010)