A Continuous Improvement Approach for Medical Device Software Development Companies

Dr. Özden Özcan-Top - Dr. Fergal Mc Caffery
Dr. Özden Özcan - Top

Dr. Fergal Mc Caffery

Dundalk Institute of Technology, Ireland
Outline

- Medical Device Software Development
- MDevSPICE®
- Challenges Associated Using MDevSPICE®
- The Continuous Improvement Approach
  - Studies in Theory
  - Studies in Practice
  - Experiences of Implementation
Medical Device Software Development

- Medical device software must meet the regulatory requirements of the region into which the device is being marketed for use.

- Safety and reliability are vital when it comes to safety critical software development.

- The role of regulations is to minimize the public health risks posed by unsafe or defective medical devices.
Medical Device Software Development

- The challenge is in the adherence to a large number of regulatory requirements specified in various international standards that can often become overwhelming
  - Difficulty in understanding what requirements are required
  - Evidences to be provided for audits
  - Finding efficient process solutions
MDevSPICE®

- MDevSPICE® is a medical device software process assessment framework.

- The purpose of development is to
  - integrate the regulatory requirements from the relevant medical device software standards
  - guide medical device software developers to produce medical software that will be safe and reliable
  - help medical companies better prepare for the demanding and costly regulatory audits

- It has 23 processes in total
  - Medical Device System Life Cycle Processes
  - Medical Device Software Life Cycle Processes
  - Support Processes

- It has been built upon 19 medical software development standards and guidelines
Some of the Standards and Guidelines within MDevSPICE®

**Life Cycle Processes**
- IEC 62304 - Medical Device Software Life Cycle Processes
- ISO/IEC 12207 Software Life Cycle Processes

**Management Requirements**
- ISO 14971 - Application of risk management to medical devices
- ISO 13485 Medical Device QMS Requirements

**Regulatory Requirements**
- FDA Title 21 CFR Quality System Regulations
- Medical Device Directives
<table>
<thead>
<tr>
<th>Process ID</th>
<th>DEV.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Software Architectural Design</td>
</tr>
</tbody>
</table>
| Purpose    | The purpose of the software architectural design process is to provide a design for the software that implements and can be verified against requirements.  
**NOTE:** Software Architectural Design is described in Clause 5.3 of IEC 62304. |
| Outcomes   | As a result of successful implementation of this process:  
- a) a software architectural design is developed and baselined that describes the software items, including SOUP, that will implement the software requirements [Class B,C];  
- b) In the case of SOUP items, all system hardware and software necessary to support the proper operation of the SOUP item is specified [Class B,C];  
- c) any segregation between software items that is necessary for risk control is identified [Class C];  
**NOTE3:** an example of segregation is to have software items execute on different processors. |
| Base Practices | **DEV.2.BP1: Describe software architecture.** Transform the software requirements into a software architectural design that describes the top-level structure and identifies its major software items. Ensure that software architecture implements system and software requirements including those relating to risk control. [Outcome: a]  
**NOTE from FDA on Validation:** Human factors engineering should be woven into the entire design and development process, including the device design requirements, analyses, and tests. Device safety and usability issues should be considered when developing flowcharts, state diagrams, prototyping tools, and test plans. Also, task and function analyses, risk analyses, prototype tests and reviews, and full usability tests should be performed. Participants from the user population should be included when applying these methodologies.  
The software design specification should include:  
- Software requirements specification, including predetermined criteria for acceptance of the software;  
- Software risk analysis;  
- Development procedures and coding guidelines (or other programming procedures);  
- Systems documentation (e.g., a narrative or a context diagram) that describes the systems context in which the program is intended to function, including the relationship of hardware, software, and the physical environment;  
- Hardware to be used;  
- Parameters to be measured or recorded…….  
**NOTE from FDA on premarket:** Architectural design may be provided also in the form of an Architecture Design Chart.  
**NOTE from 82304:** To reduce the RISK to an acceptable level, the architecture specification shall, where appropriate, make use of:  
- fail-safe functions,  
- diversity,  
- partitioning of functionality,  
- defensive design, e.g. plausibility checks  
**NOTE from FDA on SOUP:** Identify the expected design limitations of the SOUP Software. |
Process ID: DEV.2
Name: Software Architectural Design

Purpose: The purpose of the software architectural design process is to provide a design for the software that implements and can be verified against requirements.

NOTE: Software Architectural Design is described in Clause 5.3 of IEC 62304.

Outcomes: As a result of successful implementation of this process:

a) a software architectural design is developed and baselined that describes the software items, including SOUP, that will implement the software requirements [Class B,C];

b) In the case of SOUP items, all system hardware and software necessary to support the proper operation of the SOUP item is specified [Class B,C];

c) any segregation between software items that is necessary for risk control is identified [Class C];

NOTE3: an example of segregation is to have software items execute on different processors.

Base Practices: DEV.2.BP1: Describe software architecture. Transform the software requirements into a software architectural design that describes the top-level structure and identifies its major software items. Ensure that software architecture implements system and software requirements including those relating to risk control. [Outcome: a]

NOTE from FDA on Validation: Human factors engineering should be woven into the entire design and development process, including the device design requirements, analyses, and tests. Device safety and usability issues should be considered when developing flowcharts, state diagrams, prototyping tools, and test plans. Also, task and function analyses, risk analyses, prototype tests and reviews, and full usability tests should be performed. Participants from the user population should be included when applying these methodologies.

The software design specification should include:

- Software requirements specification, including predetermined criteria for acceptance of the software;
- Software risk analysis;
- Development procedures and coding guidelines (or other programming procedures);
- Systems documentation (e.g., a narrative or a context diagram) that describes the systems context in which the program is intended to function, including the relationship of hardware, software, and the physical environment;
- Hardware to be used;
- Parameters to be measured or recorded……..

NOTE from FDA on premarket: architectural design may be provided also in the form of an Architecture Design Chart.

NOTE from 82304: To reduce the RISK to an acceptable level, the architecture specification shall, where appropriate, make use of:

- fail-safe functions,
- diversity,
- partitioning of functionality,
- defensive design, e.g. plausibility checks

NOTE from FDA on SOUP: identify the expected design limitations of the SOUP Software.
<table>
<thead>
<tr>
<th>Process ID</th>
<th>DEV.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Software Architectural Design</td>
</tr>
</tbody>
</table>
| Purpose    | The purpose of the software architectural design process is to provide a design for the software that implements and can be verified against requirements.  
NOTE: Software Architectural Design is described in Clause 5.3 of IEC 62304. |
| Outcomes   | As a result of successful implementation of this process:  
a) a software architectural design is developed and baselined that describes the software items, including SOUP, that will implement the software requirements [Class B,C];  
b) In the case of SOUP items, all system hardware and software necessary to support the proper operation of the SOUP item is specified [Class B,C];  
c) any segregation between software items that is necessary for risk control is identified [Class C];  
NOTE3: an example of segregation is to have software items execute on different processors. |
| Base Practices | **DEV.2.BP1: Describe software architecture.** Transform the software requirements into a software architectural design that describes the top-level structure and identifies its major software items. Ensure that software architecture implements system and software requirements including those relating to risk control. [Outcome: a]  
**NOTE from FDA on Validation:** Human factors engineering should be woven into the entire design and development process, including the design device requirements, analyses, and tests. Device safety and usability issues should be considered when developing flowcharts, state diagrams, prototyping tools, and test plans. Also, task and function analyses, risk analyses, prototype tests and reviews, and full usability tests should be performed. Participants from the user population should be included when applying these methodologies.  
The software design specification should include:  
– Software requirements specification, including predetermined criteria for acceptance of the software;  
– Software risk analysis;  
– Development procedures and coding guidelines (or other programming procedures);  
– Systems documentation (e.g., a narrative or a context diagram) that describes the systems context in which the program is intended to function, including the relationship of hardware, software, and the physical environment;  
– Hardware to be used;  
– Parameters to be measured or recorded………

**NOTE from FDA on premarket:** architectural design may be provided also in the form of an Architecture Design Chart.  
**NOTE from 82304:** To reduce the RISK to an acceptable level, the architecture specification shall, where appropriate, make use of:  
– fail-safe functions,  
– diversity,  
– partitioning of functionality,  
– defensive design, e.g. plausibility checks  
**NOTE from FDA on SOUP:** identify the expected design limitations of the SOUP Software. |
Challenges Associated with Using MDevSPICE®

- MDevSPICE® was originally designed to be used with both of the traditional waterfall and V-model lifecycle models.
  - It is pretty straightforward to produce the necessary deliverables required to achieve regulatory audits with these models.
  - Verification, validation and risk assessments are particularly important in medical device software development.
  - This approach works well when there is high confidence in the requirements defined.

- But they’re also rigid and cannot accommodate changes easily.
- Being fast time to market and competitiveness.
The Solution we propose: Continuous Improvement Approach

▶ The First Phase: Studies in Theory
  - Evaluating the level of the regulatory compliance
  - Developing an agile practices repository
  - Extension of MDevSPICE®-with agile software development practices so that it can be more flexible and provide guidance on efficient sw development.

▶ The Second Phase: Studies in Practice
  - Introducing the continuous improvement philosophy to the companies and making it a part of their daily life
1st & 2nd Generation Agile Approaches
- eXtreme Programming
- Scrum
- DevOps
- Lean
- KANBAN
- Disciplined Agile Delivery
- Scaled Agile FramEwork

MDevSPICE covers
- IEC 62304:2006
- IEC 80002-1:2009
- IEC 80002-3:2014
- IEC 13485:2003
- FDA Guidances
- IEC 60601-1:2012
- IEC 62366:2007
- IEC 82304-1:2014

IN THEORY

AGILE INTEGRATED MDevSPICE

IN THE FIELD

Process Assessment based on MDevSPICE

Value Stream Mapping

Prioritization of Needs

Introducing Improvement KATA Technique

Implementation

Continuous Re-Assessments
# Scrum Example

<table>
<thead>
<tr>
<th>Scrum Events</th>
<th>Descriptions of the Events</th>
<th>MDevSPICE® Processes and Base Practices</th>
</tr>
</thead>
</table>
| Sprint Planning   | “The work to be performed in the Sprint is planned at the Sprint Planning. This plan is created by the collaborative work of the entire Scrum Team.” | PRO.1 Project Planning  
PRO.1.BP4: Define and maintain estimates for project attributes  
PRO.1.BP5: Define project activities and tasks.  
PRO.1.BP7: Identify and monitor project interfaces.                                                             |
| Daily Scrum       | “A 15-minute time-boxed event for the Development Team to synchronize activities and create a plan for the next 24 hours.” | PRO2. Project Assessment and Control  
PRO.2.BP3: Report progress of the project.  
PRO.2.BP4: Perform project review.                                                                               |
| Sprint Review     | “A meeting held at the end of the Sprint to inspect the Increment and adapt the Product Backlog. The timeline, budget, potential capabilities, and marketplace for the next anticipated release of the product are reviewed” | PRO2. Project Assessment and Control  
PRO.2.BP1: Monitor project attributes  
PRO.2.BP2: Monitor project interfaces  
PRO.2.BP3: Report progress of the project.  
PRO.2.BP4: Perform project review.  
PRO.2.BP5: Act to correct deviations.                                                                         |

## Mapped MDevSPICE® Processes

<table>
<thead>
<tr>
<th>Mapped MDevSPICE® Processes</th>
<th>Scrum Coverage Ratios</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. PRO.1 Project Planning</td>
<td>100%</td>
</tr>
<tr>
<td>2. PRO.2 Project Assessment and Control</td>
<td>90%</td>
</tr>
<tr>
<td>3. ENG.1 Stakeholder Requirements Definition</td>
<td>55%</td>
</tr>
<tr>
<td>4. ENG.2 System Requirements Analysis</td>
<td>71%</td>
</tr>
<tr>
<td>5. DEV.1 Software Requirements Analysis</td>
<td>33%</td>
</tr>
</tbody>
</table>

**Inverse**

1st & 2nd Generation 
Agile Approaches

eXtreme Programming 
Scrum 
DevOps 
Lean 
KANBAN 
Disciplined Agile Delivery 
Scaled Agile Framework
Why is there no 100 % coverage on requirements processes?

» «ENG.1.BP8: Establish stakeholder requirements baseline»
  - Every change on the product has to be made in a controlled way whether it is on the artifacts or the code

» Maintaining stakeholder requirements traceability to the sources of stakeholder need

» DEV.1.BP3: Determine the impact of the requirements have on the operating environment
  - It is important to determine the interfaces between the software requirements and other elements of the operating environment such as third party software.
## Studies in Practice: 1-Assessments

<table>
<thead>
<tr>
<th>Company</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Company A</strong></td>
<td>Company A develops medical applications for iOS, Android, Windows 8 and Web Browser. The products that they developed are classified as Class B. Small sized company including 7 people</td>
</tr>
<tr>
<td><strong>Company B</strong></td>
<td>Company B develops software that is currently not safety critical but the organisation has demands placed upon them from their industry as it has to be always accurate, reliable and consistent. It includes 50 employees.</td>
</tr>
<tr>
<td><strong>Company C</strong></td>
<td>Company C develops mobile and web medical applications to assist patients. Class B based on IEC 62304:2006. Small sized company with 10 people</td>
</tr>
<tr>
<td><strong>Company D</strong></td>
<td>Company D develops personalized safety critical applications for patients to support them in behavior change and improve patient engagement with healthcare practitioners. Large scale company employing more than 150 people across distributed environments.</td>
</tr>
</tbody>
</table>
Planning for Assessment

10 processes selected
- Project Planning,
- Project Monitoring,
- Stakeholder Requirements,
- System Requirements Analysis,
- System Architectural Design,
- Software Requirements Analysis,
- Software Architectural Design,
- Software Unit Implementation and Testing,
- Software Integration and Testing, Software System Testing
<table>
<thead>
<tr>
<th>Interviewed Staff</th>
<th>Company A</th>
<th>Company B</th>
<th>Company C</th>
<th>Company D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Chief Technical Officer and Chief Executive Officer</td>
<td>Chief Technical Officer</td>
<td>Chief Technical Officer and Software Architect</td>
<td>Program Manager and Product Manager</td>
</tr>
<tr>
<td>Total interview hours</td>
<td>6 hours</td>
<td>6 hours</td>
<td>6 hours</td>
<td>11 hours</td>
</tr>
<tr>
<td>Assessed project type</td>
<td>Web and mobile based decision support tool</td>
<td>Tool to monitor individual and team performance in real-time</td>
<td>Mobile and web based exercise guidance software</td>
<td>A Platform which provides an infrastructure for a many medical device software applications which they also develop in-house.</td>
</tr>
<tr>
<td>Number of the Assessed Processes</td>
<td>7</td>
<td>6</td>
<td>23</td>
<td>23</td>
</tr>
<tr>
<td>Assessed Processes</td>
<td>Project Planning - Project Monitoring - Stakeholder Requirements Definition - Software Architectural Design - Software Unit Implementation and Testing - Software Integration and Testing - Software System Testing</td>
<td>Project Planning - Project Monitoring - Stakeholder Requirements Definition - System Requirements Analysis - System Architectural Design - Software Requirements Analysis</td>
<td>All MDevSPICE processes</td>
<td>All MDevSPICE processes</td>
</tr>
</tbody>
</table>
Summary of the Issues Found

- Lack of formal evidences
- Progress against project objectives is not signed off
- Estimations are performed in an Ad-Hoc way
- Major issues in requirements management,
- No traceability established between the (stakeholder) requirements and the code base
- Changes to requirements are not tracked
- Customer involvement was quite limited
Company A-B-C

- 3rd party functionality and the risks associated with that functionality at the software requirements, system requirements and software architecture stages were not taken into account and often overlooked
- Software tested manually
Issues cont’d

➢ Company D

- Established quality management system is in place
- JIRA Software and JAMA are effectively used with integration to establish the full traceability and the follow-on work that needs to be done
- Evidences are developed for audits
- Customer involvement was limited
- Automated unit and regression tests are in place
- Effective management scheme for the program level
- Seeing potential bottlenecks in architecture 2-3 sprints ahead by having sufficient visibility of the backlog
- Reduce the effort that is required to produce technical documentation
Studies in Practice - 2-Value Stream Mapping*

- VSM is one of the lean practices used in the Toyota Production System
- We identify the value stream of a product or feature to specify where to direct our attention for process improvement
- It is the process of identifying various process blocks at a decent level of detail within the product delivery

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lead Time (LT)</td>
<td>Total time to complete a process block with all interruptions</td>
</tr>
<tr>
<td>Process Time (PT)</td>
<td>Actual time to complete a process block without any interruptions</td>
</tr>
<tr>
<td>Percent Complete and Accurate (C/A %)</td>
<td>The proportion of time a process receives something from an upstream process that does not require rework</td>
</tr>
</tbody>
</table>

VSM Implementation - Company A, a Single Feature

Process Time: 10-15 minutes
Lead Time: 2 weeks
C/A%: 50%

High Level Requirements Analysis: 1-2 hours
Protoyping: 1-1.5 days
Requirements Analysis: 0.5 days
Requirements and Cost Approval: 1 day

Design and Development with Team: 0.5 days
Writing the Test Cases: 1 day
Development Outsourced: 2 weeks
Final Testing and Quality Assurance Indoors: 0.5 day

Total Process Time: 2 weeks, 5 days
Total Lead Time: 12.5 weeks
Studies in Practice – 3- Prioritization

- List of the Issues - both from MDevSPICE assessment and VSM-done
- Goals-done
- Improvement Backlog
  - Priority list at the project level and at the organizational level

<table>
<thead>
<tr>
<th>Goal</th>
<th>Focus Process</th>
<th>Challenge</th>
<th>Size (S-M-L-XL)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Improvement KATA*

- «Finding information to support a theory is never a problem, but testing the theory and then taking the correct action is hard»
- It is important that we do not over invest.
- Most of the agile methods provide heuristic approach for implementation not the rules or steps.
- Values > Principles > Practices---require judgement
- That’s why the EXPERIMENTATION is very important

* Mike Rother – KATA http://www-personal.umich.edu/~mrother/Homepage.html
Improvement KATA- Meaning:

«*Way of Doing*»

IK is a general-purpose framework and a set of practice routines for reaching goals where the path to the goal is uncertain.

*Kata* are structured routines that you practice deliberately, especially at the beginning, so their pattern becomes a habit and leaves you with new abilities.

**THE FOUR STEPS OF THE IMPROVEMENT KATA MODEL**

A systematic, scientific pattern of working

1. Understand the Direction or Challenge
2. Grasp the Current Condition
3. Establish the Next Target Condition
4. Iterate Toward the Target Condition

**Figure by:**
Mike Rother
The facts

➢ The experimental approach takes longer but has

➢ Advantages of
  - Showing measureable benefits faster
  - Reducing the risk of change
  - People to learn themselves what it Works
  - Improvement continous after the change agents leave from the organization
Focus Process:

Target Condition
Achieve by:

Challenge:

Current Condition

PDCA Cycles Record

Obstacles Parking Lot

By Mike Rother – KATA http://www-personal.umich.edu/~mrother/Homepage.html
## An Example

<table>
<thead>
<tr>
<th>Focus Process:</th>
<th>Challenge:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stakeholder Req. Man Obtain requirements</td>
<td>Managing Different Priorities</td>
</tr>
</tbody>
</table>

### Target Condition

Achieve by: **One week**

- Understanding the characteristics of the stakeholders
- Deciding who are accountable for elicitation of the requirements
- Deciding how to interact with stakeholders

### Current Condition

**One product, different customers, limited resources and changing priorities**

Forgetting a stakeholder who has an impact on the requirements and figuring this out in later phases of the project

### PDCA Cycles Record

Stakeholder Maps to understand who the stakeholders of the project are

<table>
<thead>
<tr>
<th>INFLUENCE</th>
<th>KEEP SATISFIED</th>
<th>MANAGE CLOSELY</th>
<th>KEEP INFORMED</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOW</td>
<td>INTEREST</td>
<td>HIGH</td>
<td></td>
</tr>
</tbody>
</table>

### Obstacles Parking Lot

- KEEP SATISFIED
- MANAGE CLOSELY
- KEEP INFORMED
Conclusion

- In the medical device development domain, we are seeking ways to be more flexible and adaptive, and ensure that medical device development companies build safe and high quality software and comply with the regulatory requirements.

- In this paper, we shared our experiences from developing such an environment with the «continuous improvement approach»
Thank you
ozden.ozcantop@dkit.ie

This research is supported by the Science Foundation Ireland Research Centres Programme, through Lero - the Irish Software Research Centre (http://www.lero.ie) grant 10/CE/I1855 & 13/RC/2094.