Comparing Embedded Systems Development Methods

Zdeněk Havlice, Juraj Vízi and Veronika Szabóová
Dept. of Computers & Informatics, FEEI Technical University of Košice, Košice, Slovakia

Abstract

This paper represents different development models and techniques used in embedded systems software life cycle, their comparison and usage in different software systems. Special topic of development methodologies is described in radiotherapy medical project, which is fully using development models and techniques in daily work.

Keywords: *Embedded systems, development models, development techniques*

1. Introduction

The performance of embedded systems of these days is powerful and really comparable to personal computers (PCs). If we take into account embedded devices, such as smartphones and tablets, nowadays they are more than just devices for doing phone calls and sending text messages. They are like small portable computers with operating system and a lot of applications. Currently there are a lot of development methods and techniques for software development. Picking-up those, which suit the needs of a project best, is not always an easy task. Therefore it makes sense to have knowledge from different methodologies. By methodology we mean a composition of one of the software development models used with one or more techniques [1].

2. Development methods

Here are some of the most well known development methods:

- Waterfall
- Incremental
- Spiral

Waterfall model consists of phases which are executed sequentially. The next phase will not start until the previous one is finished completely. This model can be used in small projects, or in projects, which are some kind of stereotype, i.e. requirements of the developed system are stable and rarely change. Disadvantage of this model is high cost if any issue is found in late phases.

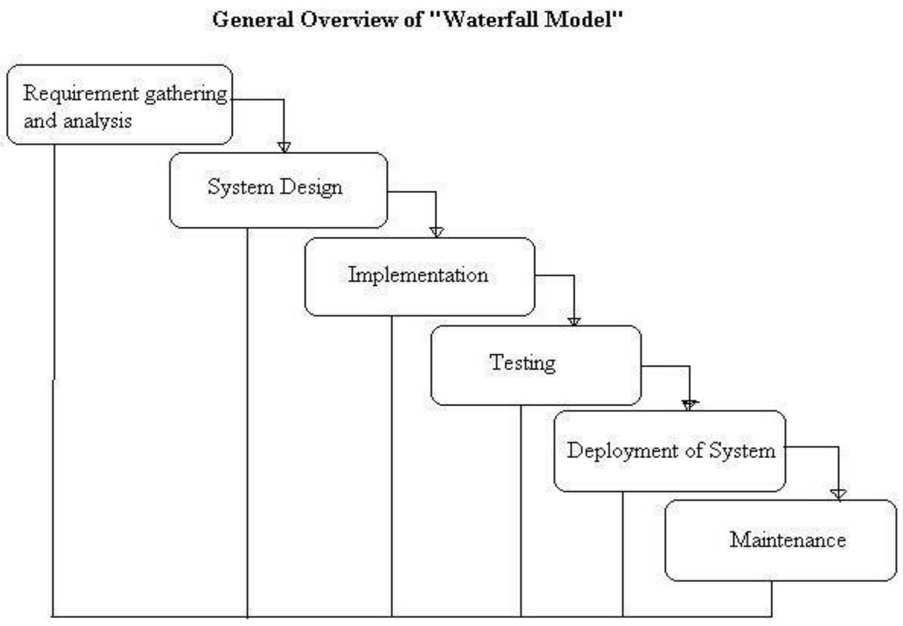
General Overview of "Waterfall Model"
Requirement gathering and analysis
System Design
Implementation
Testing
Deployment of System
Maintenance

Figure 1 Waterfall model [2]

The incremental model is an improved waterfall model with overlapping sections, which makes the project shorter in length as it would be in case of waterfall model. Incremental model leads to creating standalone fully running features - increments, which then needs to be integrated together. This model can be used in projects, where it is better to develop the whole product by these increments.

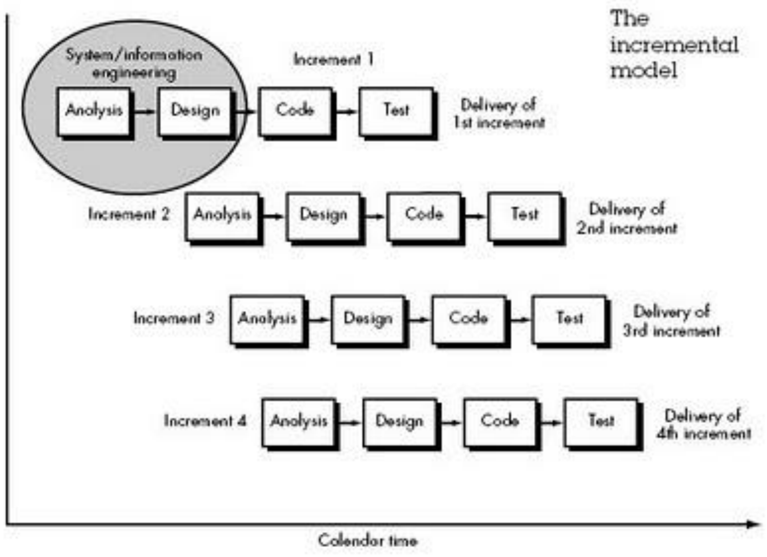
The incremental model
System/information engineering
Increment 1
Analysis
Design
Code
Test
Delivery of 1st increment
Increment 2
Delivery of 2nd increment
Increment 3
Delivery of 3rd increment
Increment 4
Delivery of 4th increment
Calendar time

Figure 2 The incremental model [3]

Spiral model is just another adaptation of incremental model. A prototype is created at the beginning, which is increased periodically. This model can be seen as a predecessor of SCRUM. In every period of time (let's call it iteration or sprint) something new is implemented, which is immediately reviewed with the customer at the iteration's end and new tasks are planned at the beginning of next iteration. The Advantage is quick response of the developed system. This model defines 3 roles in project team:

- Product owner – represents the customer, defines the requirements.
- Scrum master – dedicated person in scrum team. Scrum master is leading the meetings and is responsible for removing obstacles during development
- Scrum team – developers and testers

The spiral model as well as SCRUM has the following parts:

- Iteration planning – it is done at the beginning of the iteration, tasks are taken from product backlog and inserted into iteration backlog
- Stand-up meetings – daily meetings lead by scrum master to monitor the progress. Each team member has to answer 3 questions: what did he/she achieved yesterday, what will do today and if there are any obstacles.
- Review – it is done at the end of each iteration together with product owner. Scrum team can prepare a demo for this review.
- Retrospective meeting – sometimes it can be together with the review or separately. The goal of retrospective meeting is to tell what went well in previous iteration, what was wrong and what can be improved.

Spiral model or SCRUM can be used in projects, where new requirements are often coming from customer or the developed software product is complex and huge (size of code > 1 million lines of code).

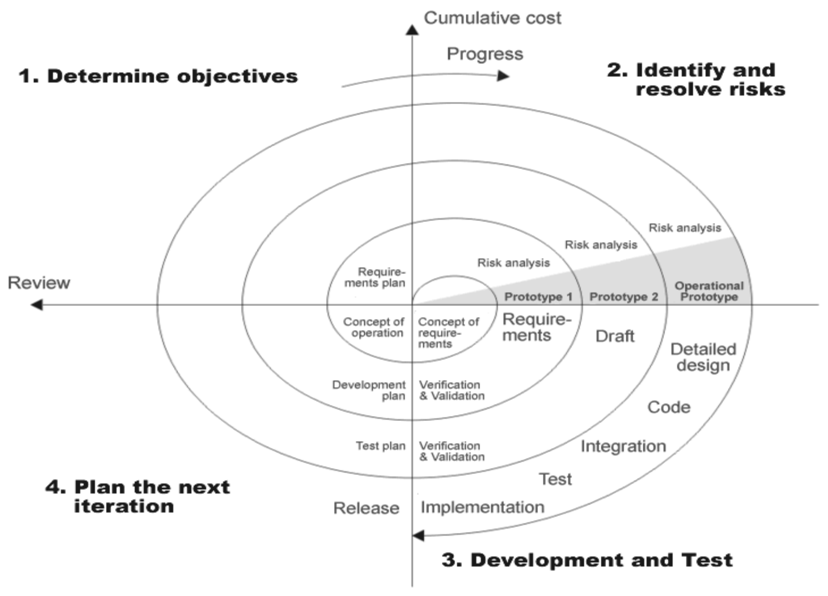

Figure 3 Spiral model [4]

3. Development techniques

Following development techniques will be explained:

- Prototyping
- Cleanroom
- Object-oriented

Prototyping technique is something what we have already seen with spiral model. It is a process of building a working replica of a system. The initial prototype created at the beginning project is periodically extended by new features. Advantages of prototyping are in regular checking of requirements (verification and validation) and early functionality. Disadvantages can be in poor documentation or making a feeling, that the developed system is almost finished.

Cleanroom technique can be used with all development models. The philosophy of cleanroom is to keep bug free software. In other words, don't start with implementing new feature until errors are in existing implementation. Advantage of cleanroom technique is having bug free software. Disadvantage is that the length of development can increase in time. Cleanroom technique can be supported by various tools, e.g. IBM Rational CharmNT – it is a bug tracking tool.

Object-oriented approach in development processes focuses on real-world objects, as it is in case of object-oriented programming. The system is data-oriented and functional complexity is lower [1]. This technique can be also supported by many tools, e.g. Enterprise Architect from Sparx, open source project StarUML or Power Designer from Sybase etc.

4. Development methods and techniques in medical software projects

This chapter is describing more in details development methods and techniques used in medical software projects, e.g. radiotherapy. This software project is created by a lot of critical requirements, which require highly sophisticated development processes. Commonly used development model in this branch is a V-model. Verification and validation of system's requirements can be explained using this model. In real projects, V-model is often applied as a combination of waterfall model with agile techniques, e.g. SCRUM. It means that the software product is developed during 2 or 4 weeks iterations and meanwhile the project milestones are strictly followed. Milestones are the items, which represents the waterfall model. Medical projects have usually these milestones:

- R1 – contract is signed and analysis of system requirements is established by project team.
- R2 – system requirements are clarified and the three main project documents are defined and released. After R2 the implementation can start.
- R3 – the product is developed in planned iterations with all processes defined by SCRUM (stand-up, iteration planning, iteration review, retrospective). In parallel with coding, product document can be updated. Coding involves not only

production code, but also unit tests and/or automated integration tests. Reaching R3 milestone means “feature freeze” for development team, i.e. the implementation of requirements, which are related to each other is stopped. This group of requirements is called feature. It’s a standalone functionality of a developed software system – something, which can be immediately shipped to customer. Then the focus of development turns from coding to testing. After R3 formal integration and system test rounds start. Found issues in software are discussed inside the project team and fixed.

- R4 – since R4 till R5 it’s more about testing, while till R3 the main scope was about coding. In this phase testers test rounds are planned and executed. This is space for cleanroom technique, i.e. found bugs are fixed. After reaching R4 milestone “code freeze” is applied, i.e. no more coding should be done.
- R5 – if this milestone is reached, it means that software passed the acceptance tests on customer’s side and can be used for the purpose it was created.

Before each milestone, there is a process called milestone review. It is executed by upper management (system and software team leaders, product management, quality manager, test leaders and technical leaders). A checklist is followed during this activity, which defines the specific steps of this review. The status of all steps says at the end if milestone is passed, passed with some conditions or failed.

The Figure 4 describes V-model. Every part of this model will be explained as it is handled in a radiotherapy project.

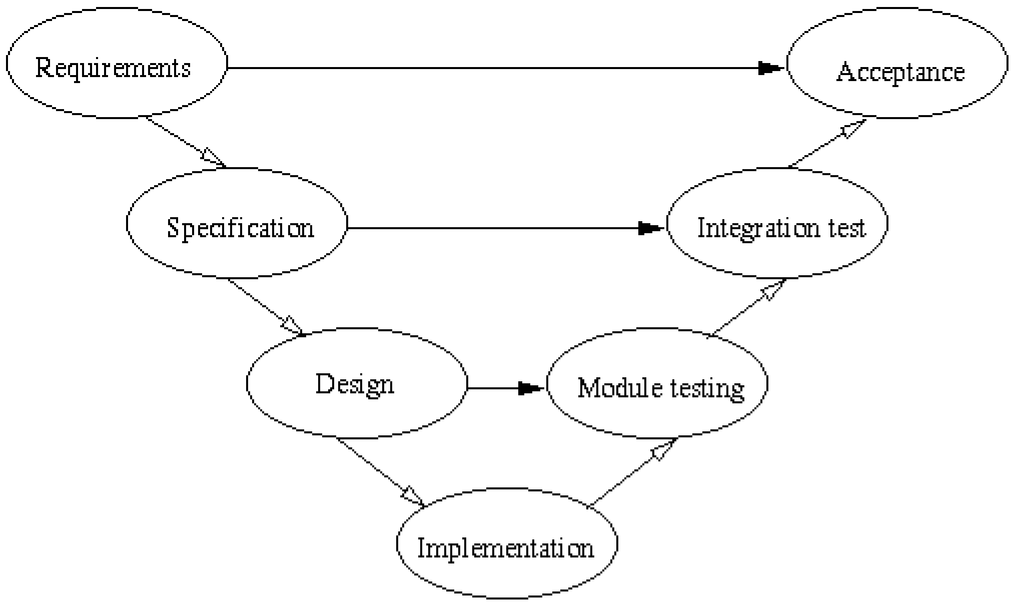

Figure 4 V-model [5]

The process begins in upper left corner (Requirements) and goes to upper right corner (Acceptance). In first phase all the requirements are collected and they are divided into smaller and more detailed requirements. In case of huge software projects (more than 1 million lines of source code) the requirements can be separated into different documents belonging to different components. This document is called Component Requirement Specification (CRS) and it contains all the requirements belonging to some specific component of developed software system. In radiotherapy the requirements in all documents are identified by keys. There are 2 kinds of keys:

- Hazard (or safety relevant) keys
- Non-hazard (or non-safety relevant) keys

Hazard keys identify requirements, which are safety-relevant. It means, that in case of any wrong behavior of system patient can be in danger or hardware can be damaged. For that reason hazard keys must follow more strictly the quality goals and regulations than non-hazard keys.

ISO 62304 (The Harmonized Standard for Medical Device Software Development) defines three safety classes [6]:

- 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

Compared to ISO 62304 we can definitely say that hazard keys include classes B and C, while non-hazard keys include class A.

The next phase in software life cycle is the design. This is the time, when the detailed architecture is designed using sophisticated tools, e.g. Enterprise Architect, which can generate source code from UML diagrams. This is called forward engineering. This kind of source can serve as a stub for coding itself. From point of documents describing the Design phase, it is the same situation as in case of Requirements phase. Several components can have their own design document. It is called Component Design Specification. What is important is the fact, that hazard keys from CRS documents are mapped into CDS documents. This process is called tracing. In general, one hazard CRS key can be traced into one or more hazard CDS keys. After finishing the detailed architecture of whole system, the implementation in a specific programming language can start.

The phase of testing especially in radiotherapy is much more complex than testing of non-medical software. It is because all of the safety-relevant requirements. There are different types of testing:

- Unit testing
- Integration testing
- System testing
- Acceptance testing

Unit tests are created for testing the smallest product of software product – a module (usually a method of a class). Unit tests are created and executed by programmers. In case of smaller team, one programmer can be responsible for production code and also for unit tests.

In case of bigger teams, this role can be divided between 2 programmers. One will be responsible for production code and second for writing unit test application. That's the principle of pair programming – one of agile techniques. If we are talking about validation than it means, that unit tests are validating the design specification. All unit test cases for any specific component can be found in document called Unit Test Procedure (UTPr). This document contains so called "UTPr keys".

Upper level of testing is called integration testing. Integration test cases are created by integration testers. The scope of integration test cases is a whole component and cooperation between components. The integration tests can be executed on a simulator. Real hardware is not required here. Integration tests are validating requirements described in CRS documents. The integration test cases are stored in Component Test Procedure (CTPr) documents. All CTPr documents of a whole system create Integration Test (IT) Plan. This document is maintained by a Test Leader.

System tests are located above integration tests on the right side of V-model. Sometimes these 2 kinds of tests can be merged, but in radiotherapy they are separated. System test cases are created by system testers and they are executed on a real hardware, i.e. the same, which is at customer (in hospital). The hospital is also the place, where acceptance tests are executed by experts, e.g. medical physicists and/or medical therapists.

Software medical projects (radiotherapy, particle therapy, ultrasound, computed tomography, etc) can't exists without quality assurance. This consists of all activities implemented in a quality system so that quality requirements for a product will be fulfilled [7]. In radiotherapy quality assurance includes people, tools and processes. There are dedicated people responsible for quality inside the projects. Processes, which belong under competencies of quality assurance responsible (QAR) are enlisted below:

- Adaptation of software processes for the needs of a project
- Regular checking of quality metrics – this task includes static and dynamic code analysis.
- Analyzing project risks and creating corrective and preventive actions
- Reporting about quality aspects
- Fulfilling the Quality goals in a project

5. Conclusion

The performance of actual embedded systems is comparable to performance of personal computers. This allows applying similar processes in software life cycle of embedded systems as they are used for personal computers. This paper represented different development models and techniques used in software processes. Reality proves that development models and techniques can be combined together to achieve the goals easier. As an example a radiotherapy project was presented, which uses waterfall methods together with agile techniques.

Acknowledgment

This work was supported by the Slovak Research and Development Agency under the contract No. APVV-0008-10 and by the Cultural and Educational Grant Agency of the Slovak Republic, Project No. 050TUKE-4/2013: "Integration of Software Quality Processes in Software Engineering Curricula for Informatics Master Study Program at Technical Universities Proposal of the Structure and Realization of Selected Software Engineering Courses".

References

[1] Reed Sorensen, "*Comparison of Software Development Methodologies*", Software Technology Support Center, http://www.google.com/url?q=http://www2.engr.arizona.edu/~ece473/readings/2-Comparison%2520of%2520Software%2520Development%2520Methodologies.doc&sa=U&ei=IlQSUfnEIYji4QTDqIGABQ&ved=0CBwQFjAD&usg=AFQjCNHT9A1Q5Lq4evled6M5y_PLepYp_Q, pp. 1-15

[2] What is Waterfall model – advantages, disadvantages and when to use it? http://istqbexamcertification.com/what-is-waterfall-model-advantages-disadvantages-and-when-to-use-it/#.UUN46Dd9Dng

[3] QTP Tutorials & Interview Questions – Incremental Model Diagram, http://qtp.blogspot.sk/2010/02/incremental-model-diagram.html

[4] Wikimedia Commons, File: Spiral model (Boehm, 1988).png http://commons.wikimedia.org/wiki/File:Spiral_model_%28Boehm,_1988%29.png

[5] The DisCo Home Page, http://disco.cs.tut.fi/tutorial/Tutorial.html

[6] ISO 62304: The Harmonized Standard for Medical Device Software Development, http://rdn-consulting.com/blog/2010/06/05/iso-62304-the-harmonized-standard-for-medical-device-software-development/

[7] Quality assurance, http://en.wikipedia.org/wiki/Quality_assurance