Software Engineering - Lecture 2

Ossama Edbali

January 13, 2015

1 Software Processes

A software process defines a structured set of activities to develop software systems. It guarantees good quality software. This creates a number of process steps that must be part of all software projects within them.

The basic steps in all software processes are:

- Feasibility and planning of the projects
- Requirement analysis
- Design
- Implementation and testing (unit or system)
- Operation and maintenance

There is no standard process model because software products are very different. One must choose a suitable process model that best fits its requirements:

- maximizes quality
- minimizes time
- minimizes cost

Elements of Software Process

There are 3 main elements in Software Processes:

- **Development process**, that specifies all engineering activities
- **Project management process**, that specifies how to plan and control those activities
- **Software configuration control process**

Software Process Models

There are various software process models but the most important are:

- **Waterfall**
- **Spiral**
- **Iterative and incremental**

Waterfall model

WM is a simple linear and rigid Document Driven Process (DDP - documents must be accurately maintained) in which every phase must be completed (frozen) before moving to the next one. There are a number of variations and adaptations of this process model. A completed phase means that it must release document that describe it.

The process is as follows:

- Requirements definition
- Software and system design
- Programming and unit testing
- Integration and system testing
- Operation and maintenance

Verification and validation

At the end of each phase we must apply these two rules:

- **Verification**, The process of evaluating the output of a phase to determine whether the products satisfy the conditions imposed at the start of that phase. *Is the product correct?*
- **Validation**, The process of evaluating the output of a phase to determine whether the products satisfy the requirements of the project. *Is the product doing the right thing? (i.e. predictable outcomes)*

Feedback in WM

This is an improved version of WM since it applies an iteration process at each phase. For example if the unit testing failed one can redesign the software. This applies to all phases in the waterfall model. However one must produce a huge quantity of documents for each iteration (heavyweight task). Moreover it affects negatively the budget and time limits.

Downsides

Such a rigid and linear process has many downsides that leads to a high risk “all or nothing” situation in which the client sees the product just at the very end of the whole process (he/she can refuse it or change requirements obviously). Clients are active just in the requirement phase (the most ambiguous one).

Advantages

When the requirements are well-understood and changes will be fairly limited during the design process.

Rapid Prototyping during requirements analysis

During the requirements analysis one can build a simple prototype of the software to give an idea to the customer how the final product should be.

Some advantages are:

- Get, at minimal cost, some idea about the requirement
- Early feedback to the clients

Disadvantages are:

- It is not useful for project when there are changes in requirements later in the project
- An unstable/badly implemented prototype often becomes the final product
- Requires extensive customer collaboration

Spiral model

Each loop in the spiral (see lecture notes) represents a phase in the process e.g. project inception in the innermost loop, requirements capture in the next loop, system design after that, etc...

In this model risks are heavily considered in each phase.

The sectors in each phase are:

- Objective setting
- Risk assessment and reduction
- Development and validation
- Planning

The advantages of such model are that it gives importance risks, it is realistic in terms of handling the unclear requirements.

The disadvantages are that it needs technical staff for the risk management sector.

Iterative and incremental models

The idea in this family of models is that software should be developed in increments. It is feature-based, thus software should be developed in increments. Each increment comes with a new release.

Each increment involves taking the next task from the list and dealing with that task in 3 phases:

- Design the implementation for this task
- Implement this task
- Analyse the result and update the project control list

Advantages of such a model are that it supports the problem of requirements changing and provides feedback to the client.

A disadvantage is that it includes the extra costs of having to redesign or reimplement existing parts of the syste