

Quality Assurance Software Development Processes

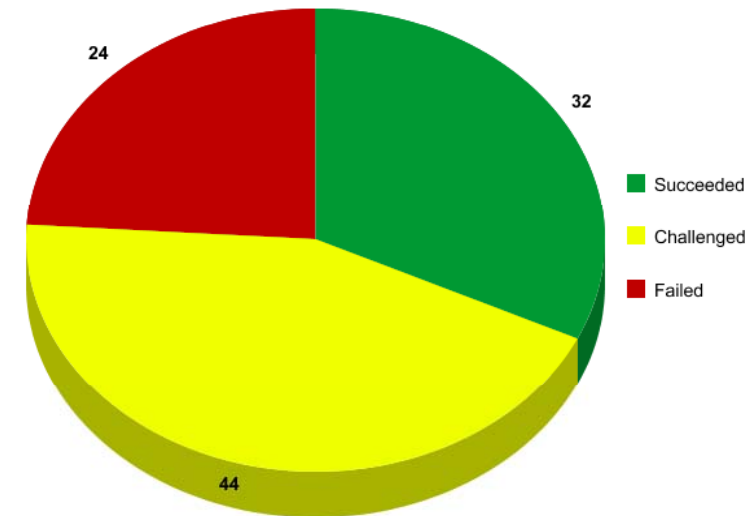
Part II - Lecture 2

The Standish "Chaos" Reports

Reports on statistics about IT projects (data for 2009)

- **32% of all projects succeeded** (delivered on time, on budget, with required features and functions)
- **44% are challenged** (late, over budget and/or with less than the required features and functions)
- **24% have failed** (cancelled prior to completion or delivered and never used)

Resolution of Projects



Among the suspected causes:
poor estimates and poor planning

<http://blog.standishgroup.com/>

Today's Outline

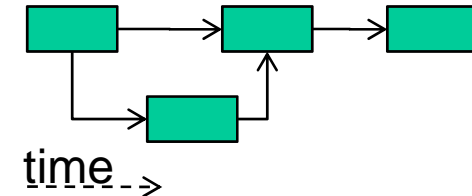
- Software Development Processes
 - What is it?
 - Phases of a process
 - Waterfall model
- Capability Maturity Model (CMM)

Software Development Processes

*He who fails to plan,
plans to fail
(Proverb)*

Software Development Process

Generic plan for a software project



1. **What** has to be done? (-> tasks/activities/steps)
 2. **Why** do a task? (-> outcomes, produced artifacts)
 3. **When** should it be done? (-> schedule)
 4. **Who** does it? (-> people, roles, responsibilities)
 5. **How** should it be done? (-> methods, standards, tools)
- Many different processes exist
 - No single process suitable for every project (no "one size fits all")
 - Using a process can improve the quality of the product

Phases of a Process 1

Simple old process model:

1. Write/change program (implementation phase)
2. Find defects, go back to 1.



Problem:

- Ad hoc changes over time mess up program structure!!!
- Result: further changes cost more and more.



Solution:

- A design phase in which overall program structure is defined
- Changes of the implementation are allowed, but have to follow the design

Phases of a Process 2

Improved process model:

1. Define a design for the program (**design** phase)
2. Write/change program (**implementation** phase)
3. Find defects, go back to 2.



Problem:

- Does the program do what the user wants?
- Result: program may be rejected by the user.



Solution:

- An **requirements** phase in which the requirements for the program are specified
- The design is defined so that the user's requirements are satisfied

Phases of a Process 3

Even better process model:

1. Analyze the requirements of the user (**requirements** phase)
2. Define a design for the program (**design** phase)
3. Write/change program (**implementation** phase)
4. Find defects, go back to 3.

You learned in the first half of the QA course:

- Testing needs to be planned and prepared
- Step 4 is important and should have its own phase
- In the **testing** phase, the product is systematically checked for defects

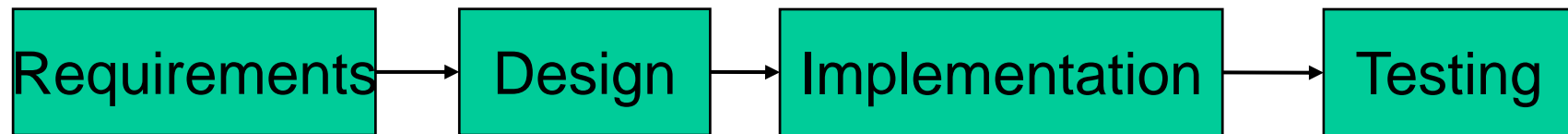
Software Development Lifecycle (SDLC)

Phase	Result
Requirements	Specification of the user requirements
Design	Specification of the overall program structure
Implementation	Executable program ("alpha" quality)
Testing	Executable program ("release" quality)

- These are just the most common phases!!!
- Other phases may include: deployment, operation, support, training, maintenance
- All the phases together are sometimes referred to as Software Development Life Cycle (SDLC)

Waterfall Model

- Old fashioned model of how a process should work
- Go through the phases one after another
- A.k.a. stagewise model, linear sequential model

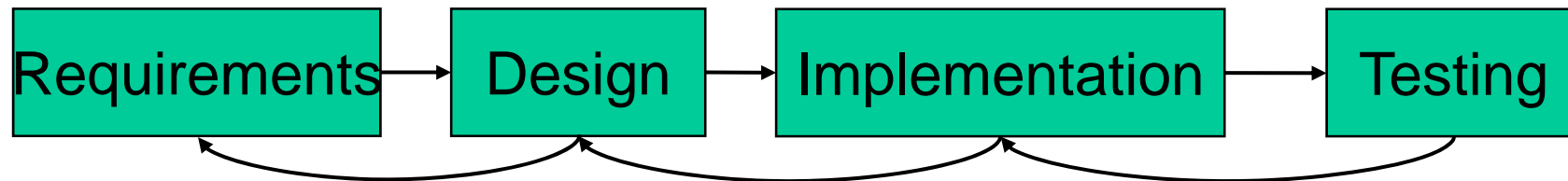


Problem:

- Often changes in artifacts of previous phases necessary
- In big systems many requirements become visible only after analysis phase or change over time
- Design flaws are often discovered during implementation and testing

Waterfall Model with Backflow

- Extends the stagewise model with possibility to go back to previous phase ("backflow")
- Defects from the previous phase can still be corrected



- Step towards *iterative* / *incremental* development (heavily used in most modern processes)
- Still very simplistic
- In reality activities of different phases often happen at the same time

The Capability Maturity Model (CMM)



*Never eat more than you can lift
(Miss Piggy)*

Capability Maturity Model (CMM)

How can we guarantee high-quality results?

Idea of CMM:

- A high-quality process yields a high-quality product
- Let's make our process high-quality!

CMM

- Describes "the key elements of an effective process"
 - Evolution from 'immature' process to 'mature, disciplined' process
 - **Key practices** for meeting goals for cost, schedule, functionality, and product quality
- Can be used to **measure** and **improve** the maturity of a process

Capability Maturity Model (CMM)

- Developed by the U.S. Department of Defence Software Engineering Institute (SEI) in the 1980s
- Has been continuously revised since then
- Motivation: objective evaluation of contractors for military software projects
- Used for many government projects
- Categorizes software development organizations into one of five process maturity levels
- Each maturity level defines:
 - A certain capability of producing quality software
 - **Key process areas** (what is done?)
 - Key practices (how is it done?)
- Level 1 worst, level 5 best

CMM Level 1: Initial

- No stable environment for developing and maintaining software
- Constant changes of the process make it unpredictable ("ad hoc").
 - Unpredictable cost, schedule, functionality, and product quality
 - Performance depends on individuals and varies with their innate skills, knowledge, and motivations ("heroic")
 - Irregular work schedule: long hours and deadline stress
- Most organizations are only level 1



CMM Level 2: Managed

- Effective management processes for software projects are institutionalized
 - Project planning
 - Project tracking and oversight
 - Requirements management
 - Configuration management (CM)
- Planning and managing new projects is based on experience with similar projects
- Successful practices from earlier projects can be repeated



CMM Level 3: Defined

- Process is standardized and documented by SE process group (SEPG)
 - Process definition, process focus
- Standard process can be tailored to the unique characteristics of a project
- Effective SE practices used
 - Software product engineering
 - Intergroup coordination
 - Peer reviews
- Training program
so that everybody can fulfill their roles



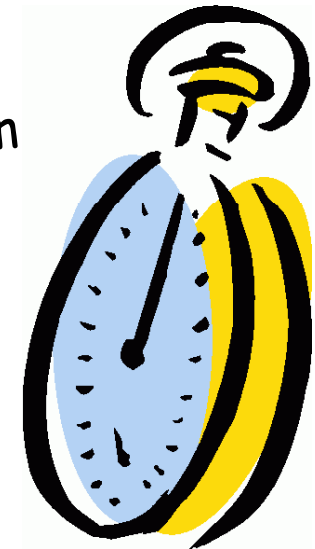
CMM Level 4: Quantitatively Managed

Quantitative process management

- The process is **instrumented** with well-defined measurements in organizational measurement program
- Productivity and quality are measured for all important activities
- Organization-wide **database** for collecting and analyzing the surveyed data
- Quantitative quality **goals** for both software products and processes

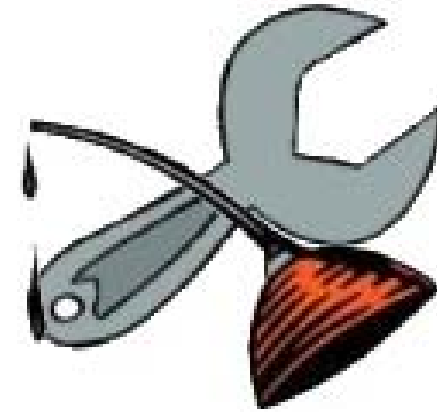
Software quality management:

- Process is controlled to operate within acceptable limits
- Predictably high quality








CMM Level 5: Optimizing

- Continuous process improvement
- Use surveyed data to analyse cost benefit of new technologies and process changes
 - Technology change management
 - Process change management
- Identify and disseminate innovation
- Defect prevention
 - Analyze causes of defects
 - Prevent known types of defects from recurring
 - Identify weaknesses and improve proactively
- Disseminate experience between projects



CMM Summary

Level		Characteristics	Key process areas
1 Initial		Unstable environment Unpredictable, ad hoc	None
2 Managed		Management processes Based on experience	Requirements management, project planning, tracking and oversight, CM
3 Defined		Standardized, docu- mented process Effective SE practices	Process definition & focus, training program, SE, peer review
4 Quant. Managed		Measurement program Predictably high quality	Quantitative process management, software quality management
5 Optimizing		Process improvement Analyze defects Disseminate experience	Technology & process change management, defect prevention

CMM Criticism

- CMM has failed to take over the world; most organizations still level 1
- Commercially more successful methodologies, e.g. RUP
- CMM is not a recipe or guarantee for success; it may just increase its probability
- Little validation of the cost savings below 4th level
- Too much bureaucratic overhead
 - Well suited for bureaucratic organizations, e.g. government agencies, large corporations
 - Focus on "perfectly completed forms" rather than application development, client needs or the market
 - Process may impede meeting schedule in cases where time to market with some product is more important than quality and functionality
- **Promoting process over substance**
(e.g. predictability over service provided to end users)

Other Quality Frameworks

SPICE (Software Process Improvement & Capability dEtermination)

- ISO answer to CMM
- Basically the 5 levels of CMM plus level 0 for incomplete/nonexistent process

ISO9000

- Family of ISO standards for quality management systems
- Certification of compliance possible, only pass or fail
- 8 quality management principles
 1. Focus on your customers
 2. Provide leadership
 3. Involve your people
 4. Use a process approach
 5. Take a systems approach
 6. Encourage continual improvement
 7. Get the facts before you decide
 8. Work with your suppliers



Today's Summary

- Software development processes define a **generic plan** for a software project
 - Processes are often divided into **phases**
 - Often **iterations** of phases are needed
- The quality of processes can be rated by **maturity models** like CMM
 - CMM defines hierarchy of 5 levels: initial, managed, defined, quantitatively managed, optimizing
 - Maturity models can help to improve a process

Reference:

Software Engineering Institute. Capability Maturity Model for Software, Version 1.1. Technical Report. 1993.

<http://www.sei.cmu.edu/reports/93tr024.pdf>

Quiz

1. What does a software development process define?
Name 5 aspects that are defined.
2. What are the 4 most common phases of a software development process?
3. What are the key characteristics of each of the 5 CMM maturity levels?