Software Process

CS307 Professional Computing

What is a software process?

- A set of activities or tasks that are followed to develop or enhance some software.
- Common activities in all software processes are:
  - Analysis and specification: determining what the system should do functionally and technically and what constraints may apply.
  - Development - production of the software system
  - Testing - checking that the software is what the customer wants and that it functions as specified
  - Enhancement and maintenance - changing the software in response to changing demands, bugs that are found, etc.

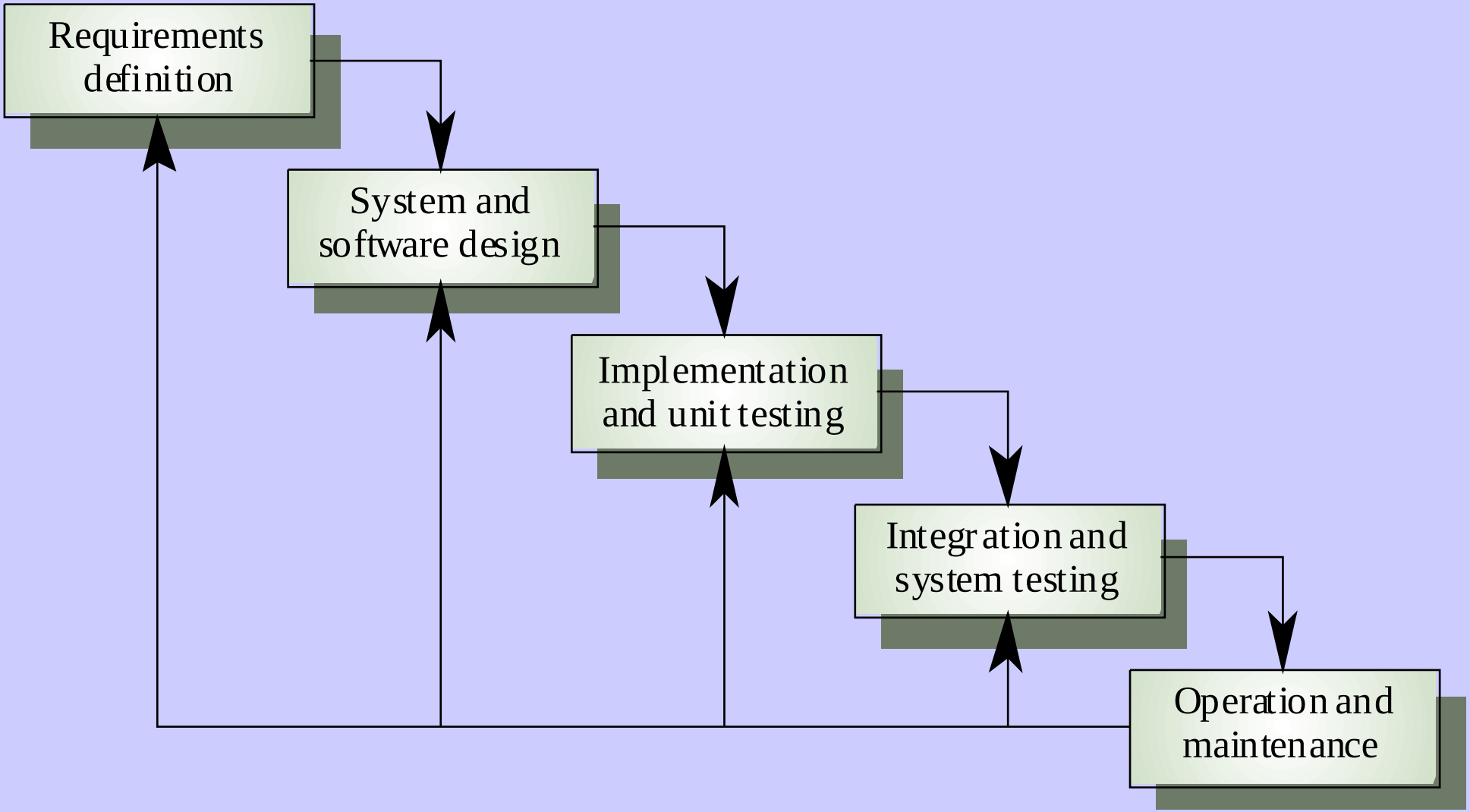
What is a software process model?

- A formalized specification of the software process, the tasks that will be undertaken and the items that will be produced.
- Mostly this is about common sense and codified best practice.
- Each organization will tailor the software process to their individual needs.

What is a software process model? (cont...)

- Historically, several generic software process models have been described but most are based on either
  - The waterfall model
    - Separate and distinct phases of specification and development
  - Iterative/evolutionary model
    - Each step is iterated until some point and steps can take place in parallel.
  - Formal systems development
    - A mathematical system model is formally transformed to an implementation

Waterfall model



Waterfall model problems

- The drawback of the waterfall model is the difficulty of accommodating change after the process is underway
- Inflexible partitioning of the project into distinct stages
- This makes it difficult to respond to changing customer requirements
- Therefore, this model is only appropriate when the requirements are well-understood

Iterative/Evolutionary Process

- Each step is iterated until an agreed point.
  - Organized around minor and major milestones.
- Steps can proceed in parallel in a limited fashion.
- Prototypes are used to explore requirements and design.

Iterative/Evolutionary process

- Problems
  - Have to agree on where to stop iterating.
  - Possibility that delivery times slip.
  - Easy acceptance of new requirements may lead to feature creep.
- Applicability
  - General

Two Processes Described

- We are going to look at two different styles of process methodology
- First is based on process based on that used at Microsoft
- Second is based on process used in some more established Open Source projects (e.g. Mozilla)

Microsoft Process

- Teams are organized into following roles:
  - Product Unit Manager
    - The manager – ultimate business and technical responsibility
  - Project Manager
    - Interface with customer, writes specs, deals with developers and test
  - Software Test Engineer
    - Very capable engineers – write tests, make sure product does what it should
  - Software Development Engineer
    - Writes the software
  - Designers
    - UI type things.
  - Misc: Marketing, Legal, etc.

Microsoft Process

- Software lifecycle is arranged around a series of Milestones
- M0 – inception phase: investigations on requirements, feasibility, technical issues, etc.
  - Early specs may be written at this stage.
  - Specs are written by PMs with developers and testers
  - Project plans are developed by PM, Development and Test
  - Developers may prototype at this stage

Microsoft Process

- M1 – coding phase
  - Specs are refined and “frozen”
  - Development of initial product is done
  - Test frameworks are written and product starts being tested.
  - Product is eventually released as an Alpha
    - Either internally as “dog food” or to select customers for feedback
    - Focus groups, usability trials, etc.
- M2 – coding phase iteration
  - Specs are updated with enhancements from customer and user feedback
  - Developers implement changes
  - Testers implement new tests – scalability and stress tests conducted
  - Product is released as another Alpha or Beta

Microsoft Process

- M3 – product release
  - Product is stabilised and all bugs resolved
  - Packaging and installation written (installation started in M1)
  - Product is released as a series of release candidates RC1, RC2, etc.
  - Product goes “gold” - eventually shipped
  - Everyone has a party.
- Lots of things about this process
  - Flexibility (what is done in what Milestone – how many Milestones)
  - Iteration, product can change radically between milestones depending on feedback, change in circumstances, etc.
  - Nightly builds – software is built every night and stable releases released often so that people can test

Open Source Process

- Very similar to Microsoft Process in many ways
- Requirements are gathered in a number of different ways
  - Typically from the developer who starts the project
  - Numbers of customers voting on a feature
  - Direct contribution of a feature (patches or code submission)
  - Project administrators or leads may decide on what feature gets accepted but mostly down to developers.
- Roles are collapsed into single individuals
  - Larger projects have distinct roles e.g. OpenOffice.org

Open Source Process II

- Code is worked on for some release milestone
  - Code is released essentially once it has been checked in
- Product is stabilized around official releases
- Communication between developers and users largely done by email
- Projects more structured if there is industry participation (Netscape for Mozilla, Sun Microsystems for OpenOffice.org)
- System works incredibly well. According to the theory of Engineering practice – it shouldn't!

How does this apply to the project?

- We will be adopting a Microsoft-style approach to the process.
- Teams will be organized around
  - Program Managers (2)
  - Developers (3)
  - Testers (2)
- Generally all the team can participate in gathering requirements but only Program Managers really need to.
- Each of the deliverables are effectively mini-milestones.
- Projects will be marked on effective implementation of the process as much as the end product.