ETSN05 Lecture 3

Course information

- Req. expert & design expert meeting with SG tomorrow (Wednesday) 12:30-13:00 in E:1149
- Reviewers:
  - group 1: Martin
  - groups 2 and 3: Johan
- All groups have access to ePuss

Contents

- project management, planning
- documents, overview of the process
- reviews

Project goals

- produce the right product
- having the required quality
- delivered at the right time
- at the given cost

Shortcomings and risks

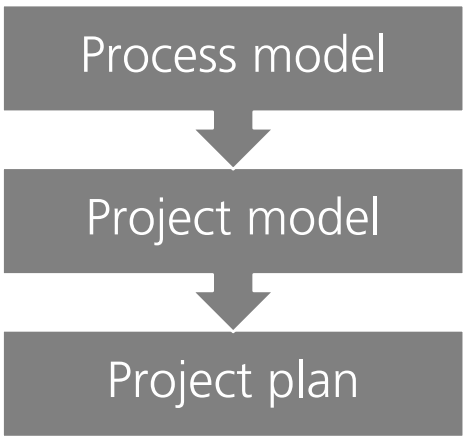
- two types of shortcomings:
  - financial: being late; being over budget
  - technical: does not meet requirements (functional and non-)
- three types of risks:
  - chaos and confusion: change in requirements, uncovering problems and errors
  - personnel: wrong people, too few people (or too many)
  - project environment: undefined methods, unknown quality, errors discovered late, insufficient management

Project management: main activities

- Planning, staffing, and organization
- Management
- Control

Planning

- Establish goals
- Define the organization
- Define standards and policies
- Define project milestones and deliverables
- Draw up project schedule
- Estimate project cost and define a budget
- Evaluate risks
- Document planning activities



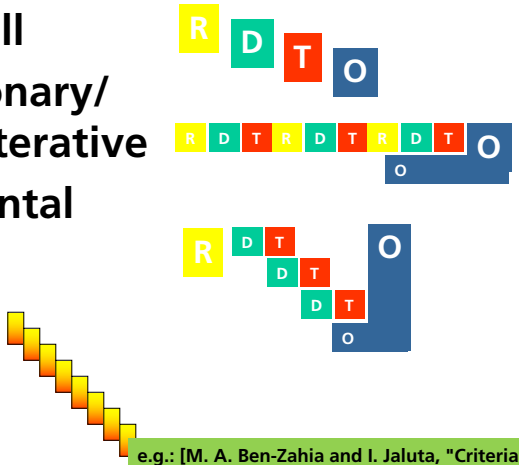
Waterfall, spiral, incremental, prototyping, agile,..

Gates, document templates, roles, tools…

Dates, persons, work breakdown…

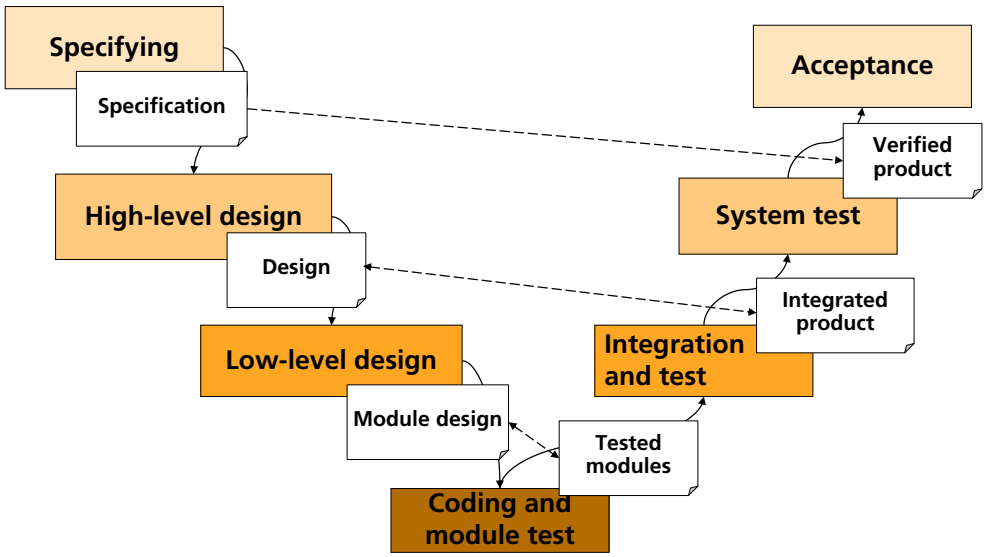
Some software process models

- Waterfall
- Evolutionary/ Spiral/ Iterative
- Incremental
- Agile



e.g.: [M. A. Ben-Zahia and I. Jaluta, "Criteria for selecting software development models," Global Summit on Computer & Information Technology (GSCIT), Sousse, 2014]

The software development cycle



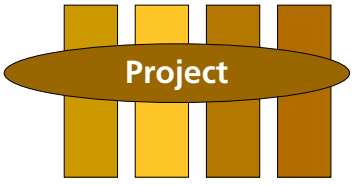
Milestones

- **Anchored in both technical tasks and business requirements**
- **Milestones are planned when activities are *complete***
- **Often, there are a deliverable or an internal document connected to the milestone**

- **Specific**
- **Measurable**
- **Responsible assigned**
- **Realistic**
- **Planned according to schedule**

Staffing and organization

- experience, education, training
- motivation and commitment
- team composition
  - line organization
  - matrix organization
  - function sub-teams



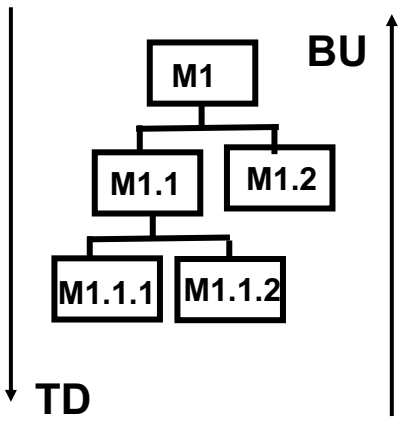
Estimation

- How much effort is required to complete an activity?
- How much calendar time is needed to complete an activity?
- What is the total cost of an activity

Costs:

- Hardware and software costs
- Travel and training costs
- Effort costs (often dominant)

Bottom-up vs. top-down



- You often start TD and divide when the uncertainty is too large
- BU may miss integration
- TD estimates may miss difficult modules

General estimation techniques

- Algorithmic cost modelling
- Expert judgement
- Estimation by analogy
- Parkinson's law
- Pricing to win

Don´t mix estimation, planning, and costing

- Estimation: The actual cost.
- Planning: How much should we plan for? E.g. add slack to remove risks?
- Costing: How much should it cost?

Magne Jørgensen, Practical Guidelines for Expert-Judgment-Based Software Effort Estimation, IEEE Software, Vol. 22, No. 3, pp. 57-63, 2005.

Combine estimation methods

- Useful combinations:
  - Top-down – Bottom-up
  - Analogy and linear regression
  - Expert judgment and formal methods
  - Expert judgment made by software professionals with different project experience
  - Different roles
- Ineffective combinations:
  - Expert judgment from experts with similar experience
  - Expert judgment from experts with same educational background and role
  - Estimation methods based same underlying principles

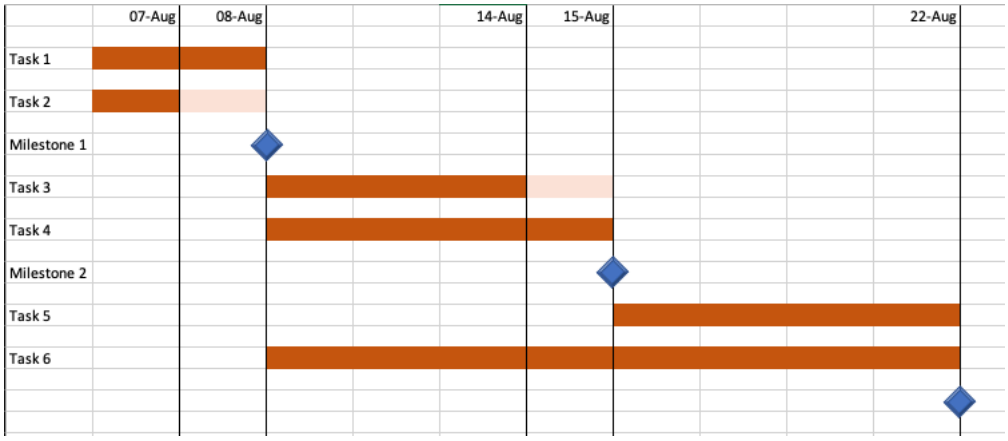
Magne Jørgensen, Practical Guidelines for Expert-Judgment-Based Software Effort Estimation, IEEE Software, Vol. 22, No. 3, pp. 57-63, 2005.

Ask for justification

- Acceptable:
  - Reference to effort of similar projects
  - Application of validated, organization-specific relationships
- Unacceptable:
  - "I believe the effort will be 5,250 hours"
  - Reference to output of from non-calibrated estimation model with no organization-specific data on its performance

Magne Jørgensen, Practical Guidelines for Expert-Judgment-Based Software Effort Estimation, IEEE Software, Vol. 22, No. 3, pp. 57-63, 2005.

Activity timeline (Gantt chart)



Management

- leadership
  - give enthusiasm and security
- follow up on personnel
  - give daily instructions
  - pursue good discipline
- delegate
- motivate
- support collaboration
- co-ordinate
- encourage communication
- solve conflicts
- stimulate improvement
- document decisions

Control

- establish reporting- and follow-up system
  - baselines
  - budget review
  - process model
  - independent review
  - verification and validation
  - quality assurance
  - configuration management
- analyse results
- initiate corrective activities
- reward obtained goals
- document control mechanisms

♦ **Project management**
♦ **Technical development**
♦ **Quality system**
♦ **Configuration management**

Other important activities

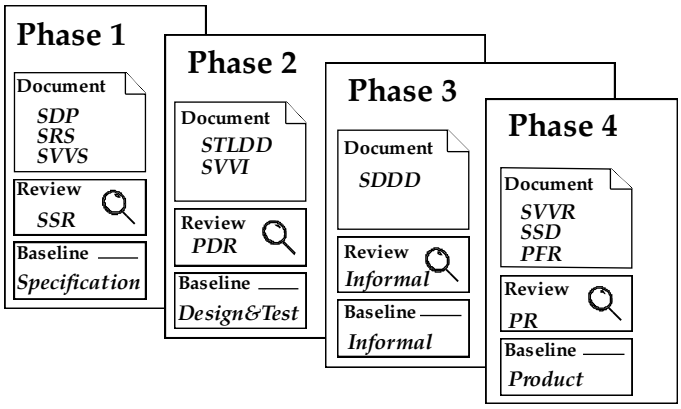
Project managers (PG) don't do all the job ...

- technical guidance (System group - SG)
- quality assurance in two parts:
  - verification & validation (TG and others through test and reviews)
  - acceptance "test"
- change management (CCB = PG + SG) (with help from others

Documentation from a project

- defines the product in terms of e.g. Requirements and design
- describes the product for the customer and the developers
- supports the users of the product, e.g. user-, administrator- and maintenance manuals (not included in course)
- Both product documents and project documents

Documents from different phases



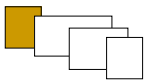
General rules for all documents

- follow PH if not stated otherwise in the SDP
- keep them in the document library
- no requirements on word processor and drawing tool
- consistent front pages
- all documents must have a configuration item number

Information transmission

- who is the target group
- which pre-requisites do they have
- what is your message
- does it say what you meant to say
- is it unambiguous and clear
- does is say from where you got the information

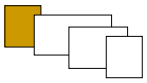
Phase 1: Specification



- decide on the project plan
- define and analyse the requirements to the product
- plan reviews and tests

- this phase ends with a formal review and a formal baseline

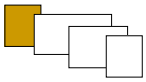
SDP



Responsible: PG

- careful plan for the development
  - time/resources per phase/week/activity/group
  - time plan (dates) for phases/documents/reviews
  - a calendar overview per week of tasks for each group
- description of personnel organization and lines of responsibility/authority
- description of support tools, methods and standards
- description of change management

SRS

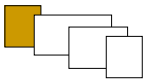


Software Requirements Specification
Responsible: SG (UG)

- analysis and identification of the customers ”needs”
- functional and quality requirements
- all terms defined explicitly
- for each service at least two examples of use (can be described using scenarios)
- structure of chapters like example SRS

- **Correctness**
- **Organized**
- **Consistent**
- **Verifyable**
- **Completeness**
- **Traceable**
- **Unambiguous**
- **Motivated**

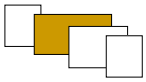
SVVS



Software Verification and Validation Specification
Responsible: TG

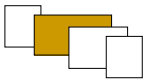
- reviews
- tests
  - what kind of test
  - a function test specification per service
  - a system test specification
  - all (testable) requirements must be referenced
- quality
  - evaluation of non-functional requirements

Phase 2: High-level design



- structure the software into high-level components
- create the design from every test case
- no detailed requirements from us
- the phase ends with a formal review and a formal baseline

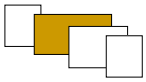
STLDD



Software Top-Level Design Document

Responsible: SG (UG)

- high-level design into components
- for all components it should say:
  - where it is placed in the system
  - its functionality
  - control- and data flow to and from the component
  - global data that are shared with other components
  - input and output



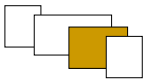
SVVI

Software Verification and Validation Instruction

Responsible: TG

- test instructions for every test case in SVVS
- every test instruction must be ”self contained”
  the difficult part is not to run the test - but
  to decide whether it went well or wrong
- test execution
  which test cases are run where?
  target machine/simulated environment

Test case x.y.z
–Pre-condition
–Setup
–Test case
–Follow-up
–Post-condition



Phase 3: Low-level design

All units/modules must be specified completely

- low-level design is followed by an informal review

SDDD

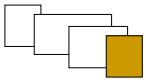
Software Detailed Design Document
Responsible: SG (UG)

- in this case the code
- (otherwise if “model driven development” divided up into chapters for:
  - system level
  - block level
  - process level)

Phase 4: Integration and system test

- test to ensure that the system satisfies the requirements
- acceptance tests show the customer that the system fulfils his/her needs
- collecting the experience from the project

- this phase ends with a formal review and a formal baseline

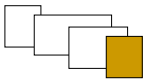


SVVR

Software Verification and Validation Report

Responsible: TG

- results of function tests for every service
- results of system tests
- review protocol from formal reviews
- the number of errors and types that were found by each review
- the number of errors and types that were found in dynamic testing
- the number of errors and types that were found in the various document
- errors that remain in the system
- comments on the results

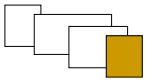


SSD

System Specification Document

Responsible: PG

- version description (delivered version, differences between versions, etc)
- possible differences against SRS (motivated)
- installation instructions (where are the files)



PFR

Project Final Report

Responsible: PG

- summary of experience from the project
- summary of metrics
- conclusions from the metrics
- error costs
- comments on what you would be able to improve on the project

Meetings

- project meetings: ~twice a week (but of course up to you)
- PG: meeting with the section manager - once a week
- meeting with the design expert – W4

Reviews

- basically it is simple: read the object of review in a systematic way
- that is: a manual technique with the goal of:
  - find mistakes
  - spread knowledge
  - get a basis for decisions
- these goals can be balanced in different ways in different kinds of review

Reviews

- What?
  - Requirements/design/ code/user interface
- How?

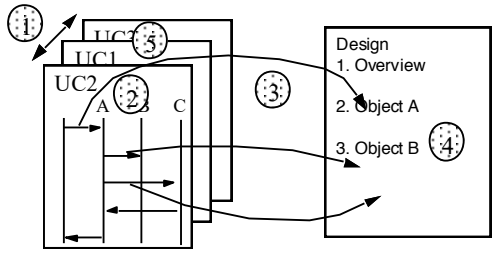
  Reading technique:
  - Ad hoc
  - Checklist
  - Scenario-based
- Who?
  - Number of reviewers
  - Roles
- When?
  - Continuously
  - Before decision

Review techniques

Check lists

- X hjkhsdf
- X jkljdk
- X hjdofqj
- X jkljdoöajd
- X lökasjldkj
- X lasjdflök

Use cases



Granskning med användningsfall

Example of review process

Roles:

- Author
- Moderator
- Reader
- Recorder
- Inspector
- Inspection process responsible

Steps:

1. Planning
2. Overview
3. Preparation
4. Meeting
5. Rework
6. Follow-up

Example review entry criteria

- previous documents must be reviews, e.g. The design is not reviewed until after the specification
- given standards have to be followed
- automatic controls must have been carried out and been approved (e.g. spelling checks)

Example of check list items

* Are all program variables initialized before their values are used?
* Is there any possibility of buffer overflows?
* Are all conditional statements correct?
* Is each loop certain to terminate?
* Are all output variables assigned a value before they are output?
* Can unexpected input result in errors?
* Are function parameters used in correct order?
* Is memory correctly de-allocated after it is no longer is used?

And more…

Error types

- error types for each document (see PH)
  - SRS
  - SVVP & SVVR
  - STLDD & SDDD
- general error types if the above do not apply
- error types from dynamic testing
- severity of errors: A, B, C

Reviews in your projects

- informal reviews - customer not present, done before every formal review
- formal reviews:
  - Software Specification Review (week 3)
    - SDP, SRS, SVVS
  - Preliminary Design Review (week 6)
    - STLDD, SVVI, monitor files (not on paper)
  - Product Review (week 10)
    - acceptance test
- the better informal review, the better formal review
- the document library is also reviewed at SSR, PDR, PR

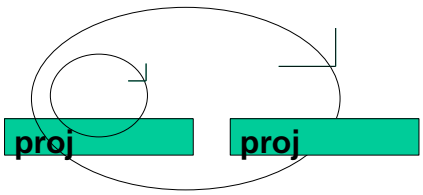
Data collection

- date for distribution of review material
- time for preparation
- date of review meeting
- end date for rework
- first review or re-review
- identity of review material (e.g. document number)
- size of review material
- names of participants
- number of reviewers
- duration of meeting
- number of errors divided by type and severity
- decisions (approved, corrected or re-review)

Compare the review protocol and problem reports

Use of information

- short term: process control (cf. project control)
  - example: estimation of remaining errors
- long term: process improvement



To do this week

- exercises 2
- prepare for lab 1 & 2
- PG: schedule review 1, SDP
- SG: SRS
- UG: SRS
- TG: SVVS
- informal review