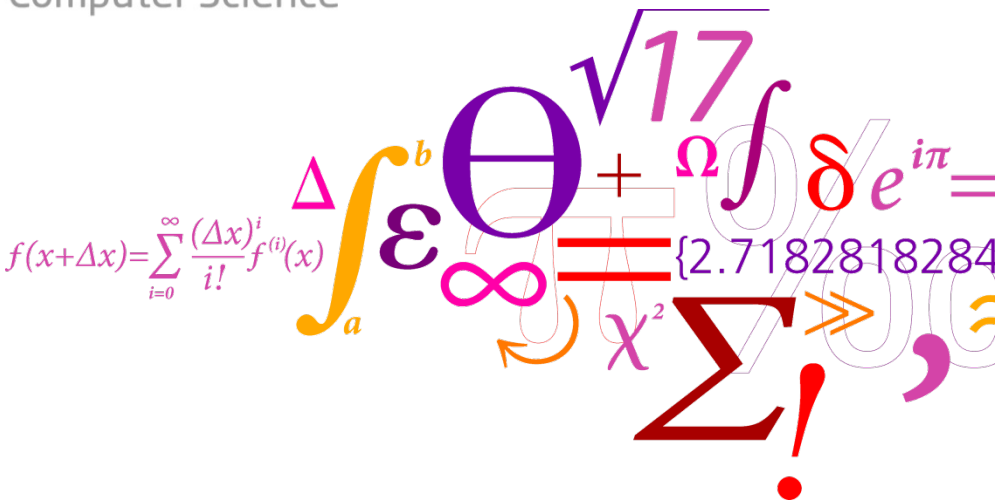
Software Engineering 2

A practical course in software engineering

Ekkart Kindler

DTU Compute
Department of Applied Mathematics and Computer Science



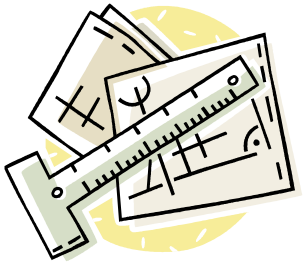
Specifying Software

Recap from Lecture 6

Goals (of "software documents"):

- Defining what the software should do (before it is really there)
- **C**ustomer and **D**eveloper agree on what should be delivered
- Effort (resources and time) can be planned based on that (contract will be based on that).

NB: Writing them is part of the process of understanding what the software should do!



Specifying Software

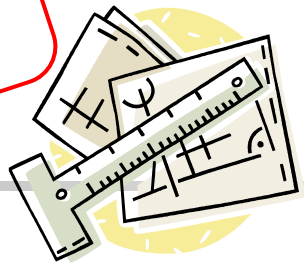
DTU Compute
Department of Applied Mathematics and Computer Science
Ekkart Kindler

what

- Project Definition
- Requirements Specification
  - rough
  - detailed
- Systems specification
- Complete Models
- Implementation, Documentation
  Handbook

Might concern both "what" or "how"

Actually, handbook is "what"; it could be part of the requirements specification.



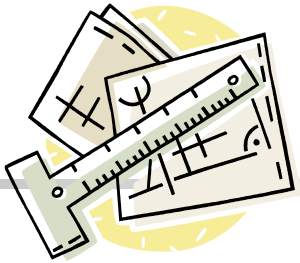
how

Specifying Software

DTU Compute
Department of Applied Mathematics and Computer Science
Ekkart Kindler

rough

- Project Definition
- Requirements Specification
  - rough
  - detailed
- Systems specification
- Complete Models
- Implementation, Documentation Handbook



detailed

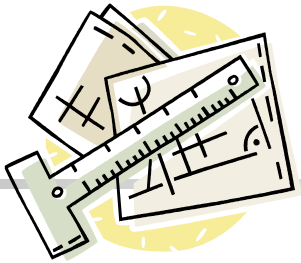
Specifying Software

DTU Compute
Department of Applied Mathematics and Computer Science
Ekkart Kindler

low cost

- Project Definition
- Requirements Specification
  - rough
  - detailed
- Systems specification
- Complete Models
- Implementation, Documentation Handbook

Maintainability

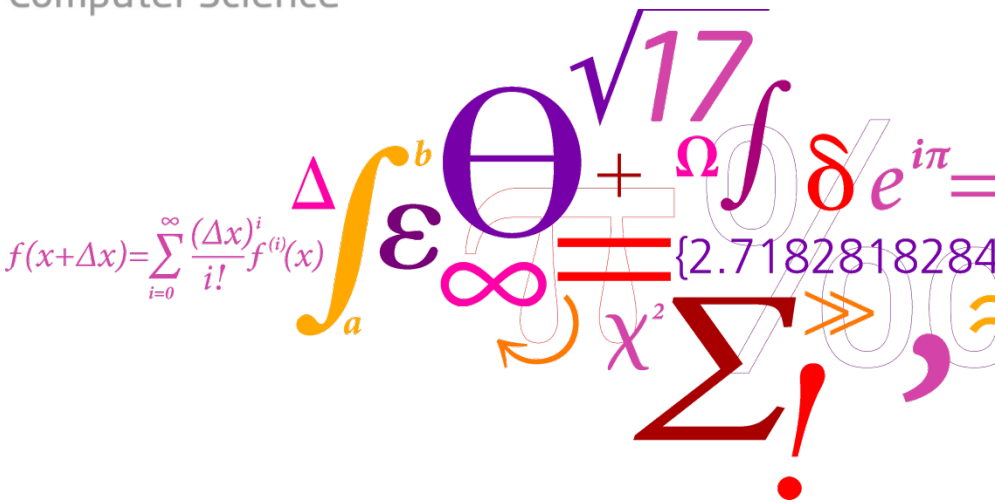


high cost

Possible Outline and Contents of your Systems Specification

Based on what you have already (Project Vision, Requirements docs, models, ...);

DTU Compute
Department of Applied Mathematics and Computer Science



1. Introduction

- Context & background
- Motivation & goals

- Partners of project
- Audience of this document

"What" (is assumed to) exist already?

"Why" do we need something else? And idea of "what" the new software will allow you to do?

2. Product Use

- Stakeholders / users
- Scenarios and examples

"fluffy" part of your project vision; could contain screenshots for getting a better feeling

- "Overview" of main concepts and functions

Could wrap up with an overview of discussing the **main terms** (by contrast to Glossary, this is written as running text).

3. Scope

On high level of abstraction; in more detail in Requirements

In particular: What is **not** covered (out of scope)

So that we can refer to them later (numbers → traceablility)

4. Requirements

- Overview main components
- Functional requirements
- Non-functional requiremensts

From end-user perspective

Use cases / activity diagrams (for non-editor functionality)

Class diagrams of domain models for editors (CRUD)

In our case, you can use User Stories to structure the functionalities

Prioritized with MSCW (scope)

5. Graphical User Interface

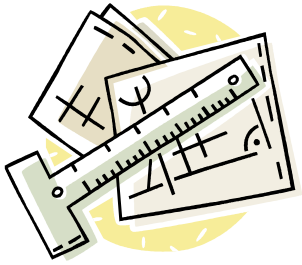
Only from the end-user ("what") perspective.

Part of GUI could discussed before requirements (e.g. Product Use): Easier to understand requirements

You do not need to discuss the parts of the GUI, which comes with the used infrastructure (e.g. openHAB); but, a glimps of it might help understand the functionality.

Up to here, ”why” and ”what” only.

Next, comes ”how”! → Software / Systems Specification.



6. Architecture Overview

Main components and interfaces between them (in particular, for editors, distinguish, Model from View and Controllers)

→ Component diagrams
→ Sequence diagrams (for important interfaces)

In order to undertstand your architecture, you might need to explain some aspects of the used.

7. Detailed design

For automatically generated editors: name used technology and focus on aspects that need to be added manually.

For some objects, a state maching of its life-cycle might be relevant (e.g. devices, …)

For important interactions: Sequence diagrams

If not covered in requirements (domain model): Data models for all permanently stored information and data used in interfaces

8. Realization

- Used platform and technology
- Use of technology-specific features in the design

Some things which you do not know yet, you can submit with the final submission

Don't forget stuff that is not programmed (but configured)!

Interesting implementation details (things that were not straight-forward)

Appendices

e.g.
Glossary
Complete example
Complete models
...

Even though you have a Glossary, make sure that the main part contains a discussion of the main concepts and terms!

Examples

DTU Compute
Department of Applied Mathematics and Computer Science
Ekkart Kindler

Have a look at the course's material page for examples of system specifications.