

<https://www.halvorsen.blog>



Week Assignment

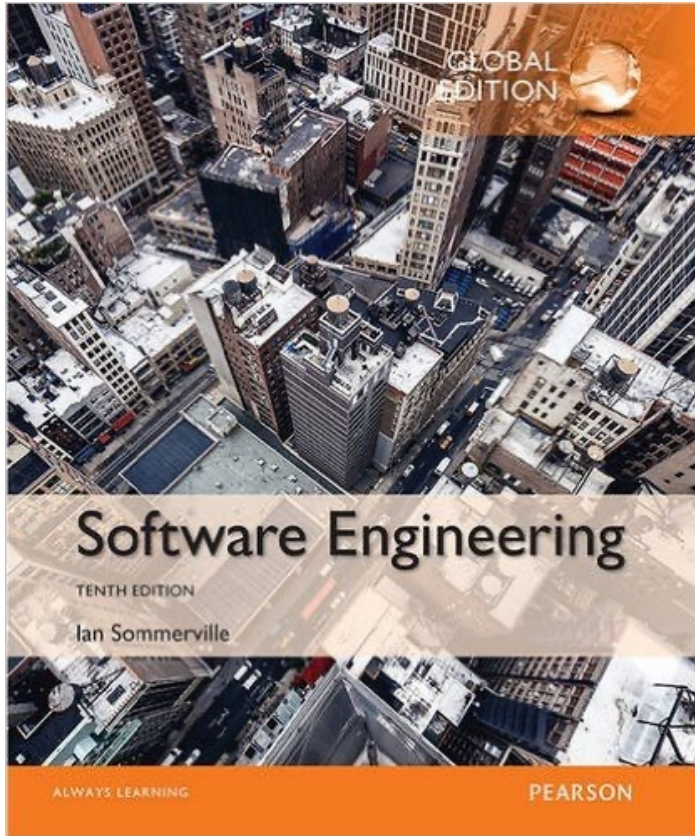
Project Management & Requirements Analysis

Hans-Petter Halvorsen

Week Assignment

1. Create Software Requirements & Design (SRS/SDD->SRD) Document(s)
2. Create/Update Project Plan/Gantt Chart
3. Start/Cont. using Azure DevOps
4. Start Coding/Implementation

Textbooks (Topics this Week)



Software Engineering, Ian Sommerville

Ch.4: Requirements Engineering

Ch.22: Project Management

Video: An Introduction to Requirements Engineering

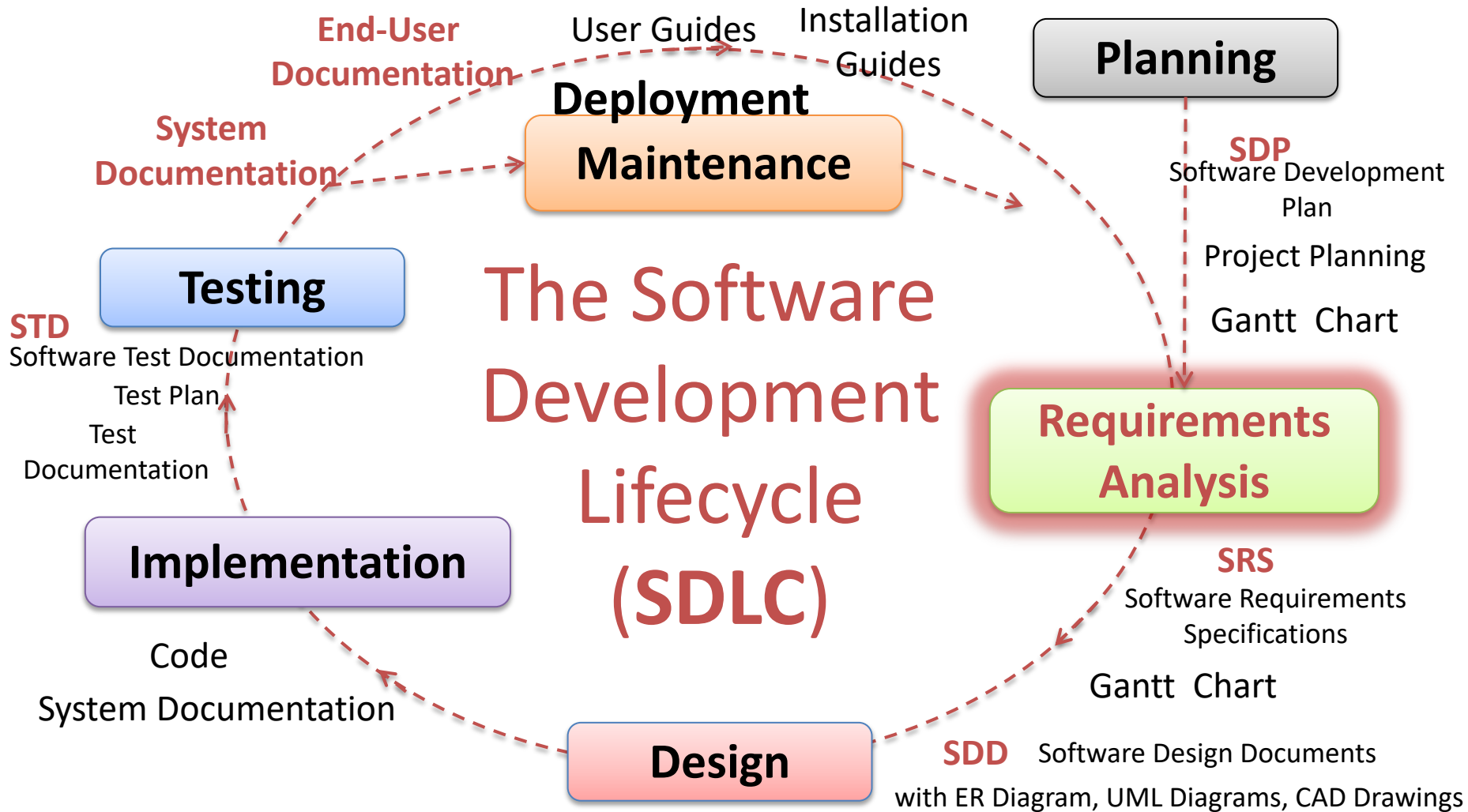
<https://youtu.be/Ec0s0z5uXQ8>



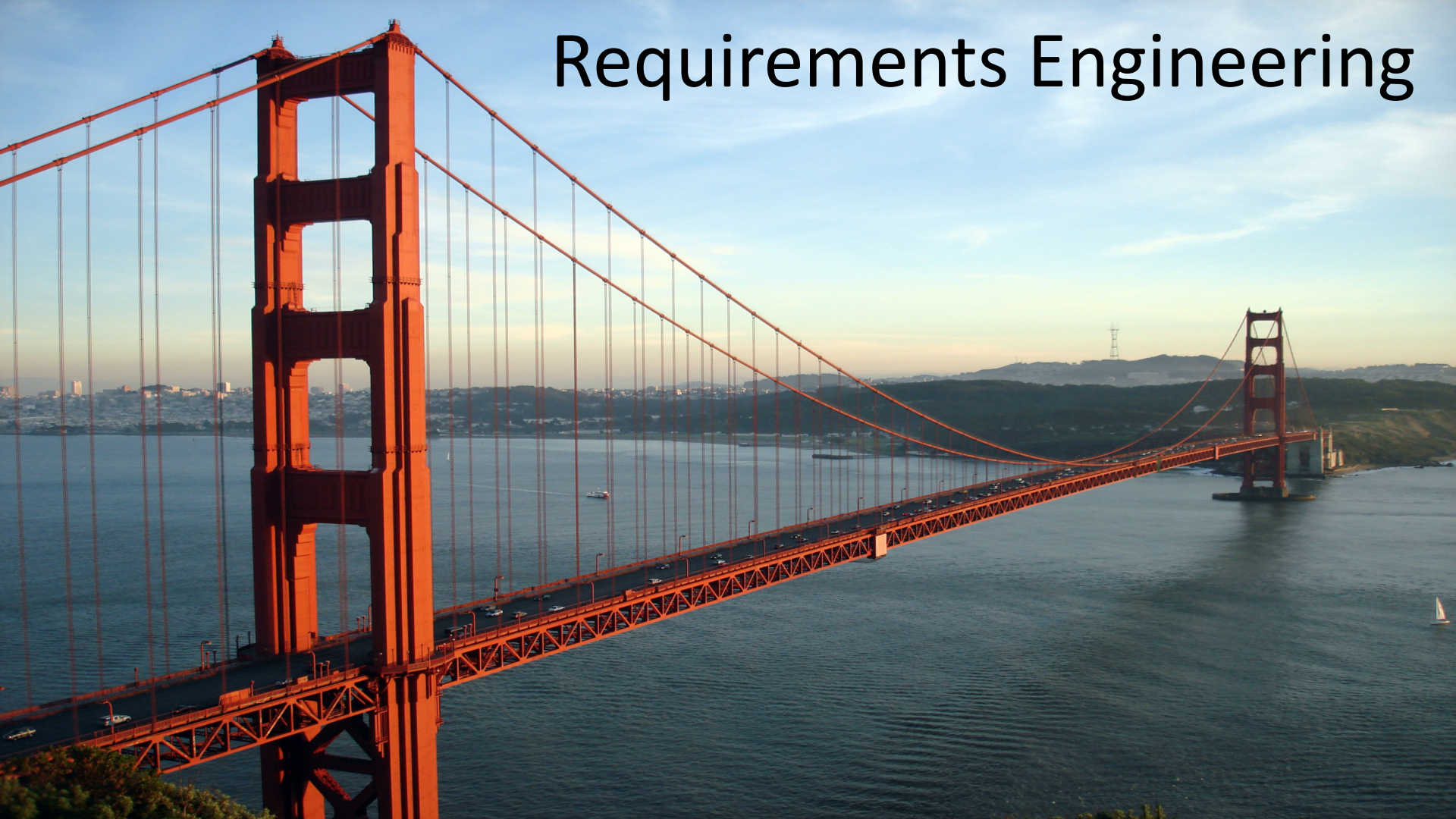
Project Management & Requirements Analysis

Hans-Petter Halvorsen

[Table of Contents](#)

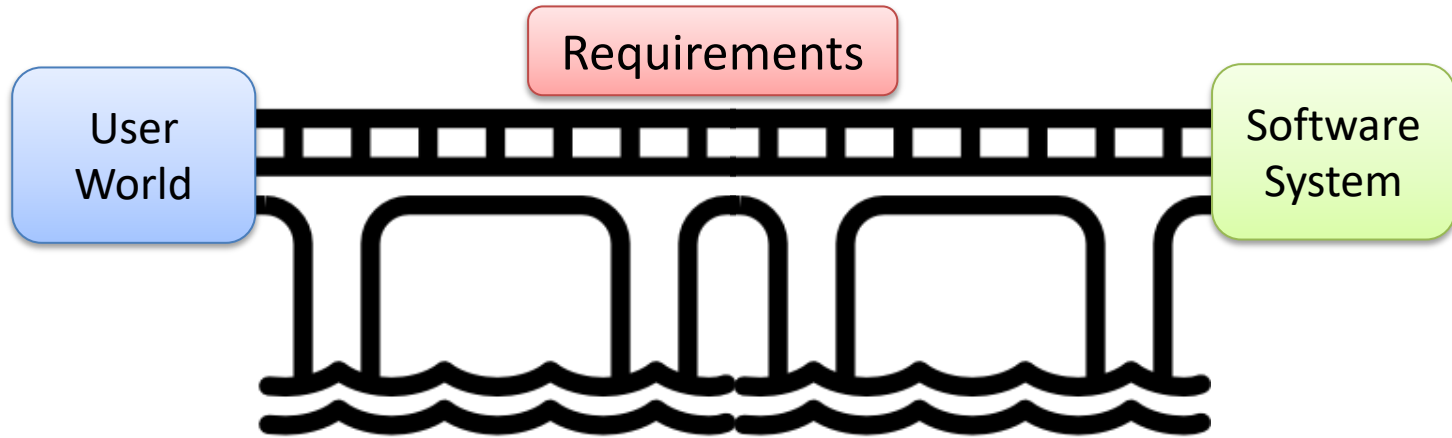


Requirements Engineering



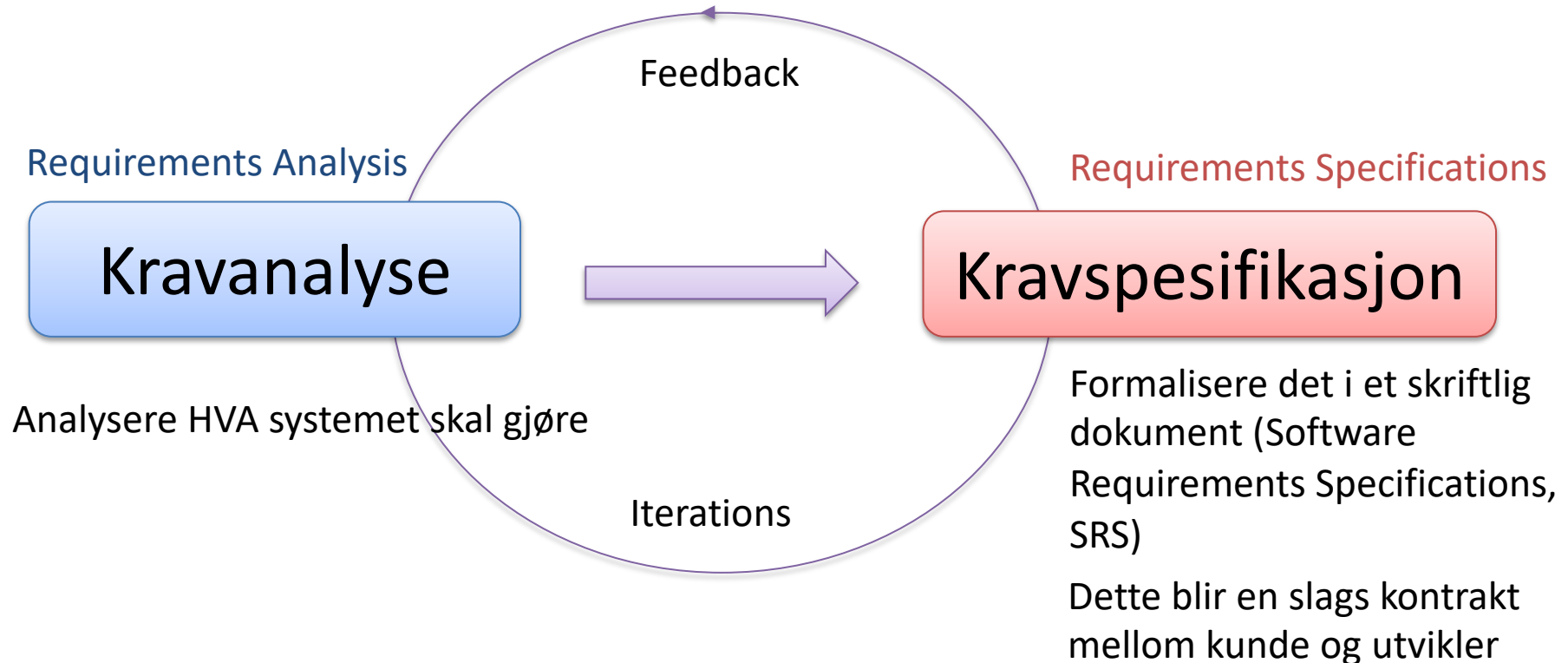
What is Requirements Engineering?

Requirements is the bridge between the real world and the software system



An Introduction to Requirements Engineering: <https://youtu.be/Ec0s0z5uXQ8>

Requirements Engineering





World's Funniest Engineering Fails Ever:
<https://www.youtube.com/watch?v=MUF1tMlnSJw>

Requirements Engineering

– Det er enkelt å glemme å spesifisere at bilen skal ha bremseser.

Når kunder spesifiserer hva de skal ha, blir enkelte ting tatt for gitt.

Hvordan sikrer du at de får kvaliteten de forventer?



Requirements Engineering



What the Customer got



What the Customer really needed

Reality?



You dont make good software using this approach!

Still, with Agile we start to implement code even if we dont have all the details at hand.



How the customer explained it



How the Project Leader understood it



How the Analyst designed it



How the Programmer wrote it



How the Business Consultant described it



How the project was documented



What operations installed



How the customer was billed



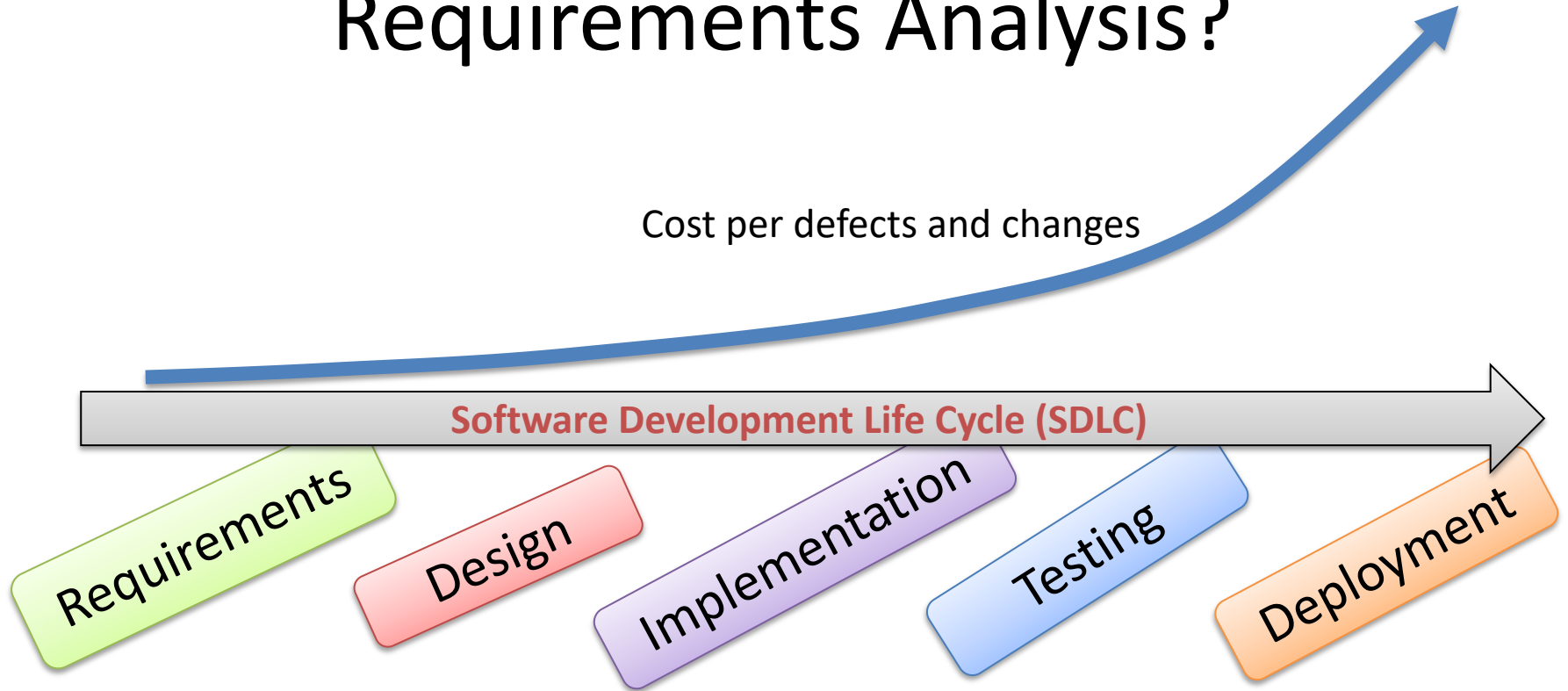
How it was supported



What the customer really needed

Take the Requirements Analysis seriously !!

Why spend time on Requirements Analysis?





Software Requirements & Design

Hans-Petter Halvorsen

[Table of Contents](#)

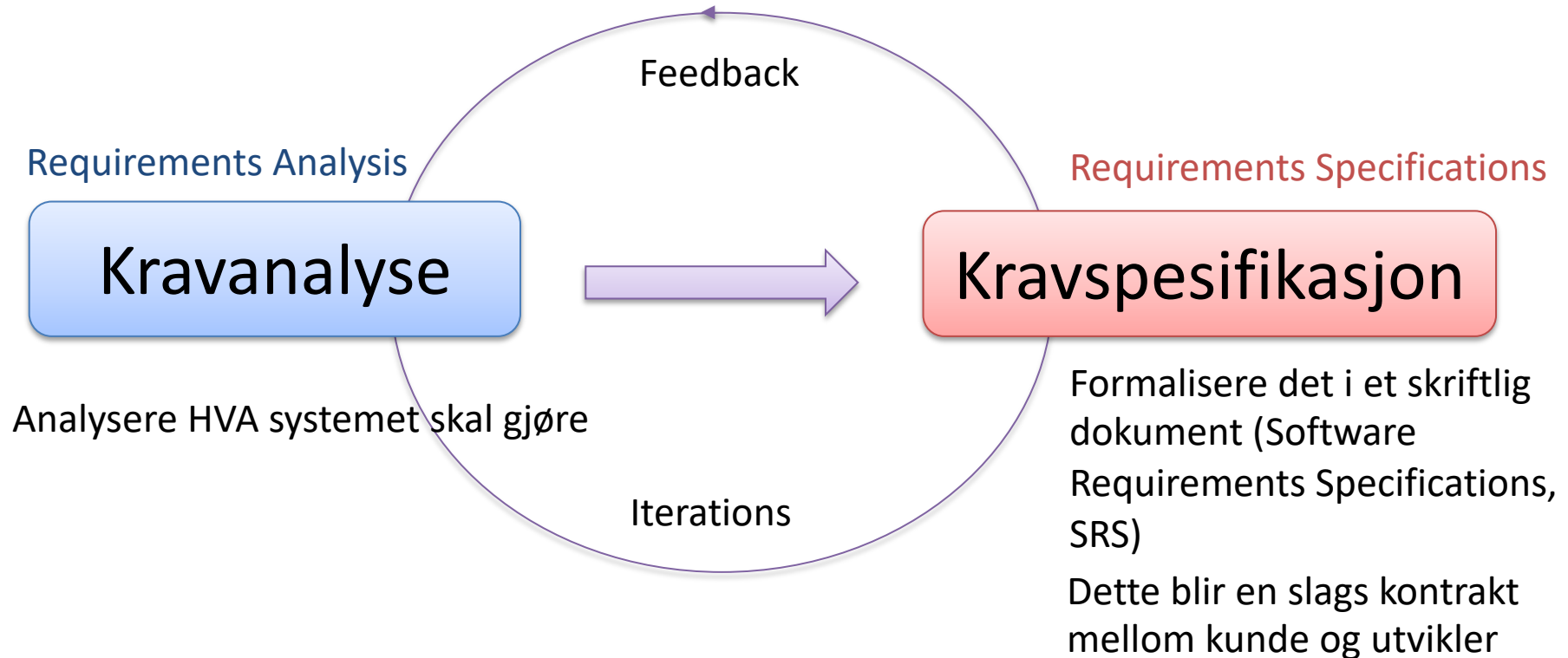
Software Requirements & Design

- Based on the Project Specifications and the Brainstorming session, create **High-Level** and **Detailed Requirements** for your Software
- Create Software Requirements Specifications (SRS)/Software Design Documents (SDD) -> **Software Requirements and Design Document (SRD)**

Note! We will add even more details the upcoming weeks, i.e., Database design, UML modelling, such as Class diagrams, etc.)

See Next Slides for more details...

Requirements Engineering



Software Requirements & Design

Requirements (WHAT):

- **WHAT** the system should do
- Describes what the system should do with Words and Figures, etc.
- **SRS** – Software Requirements Specification Document

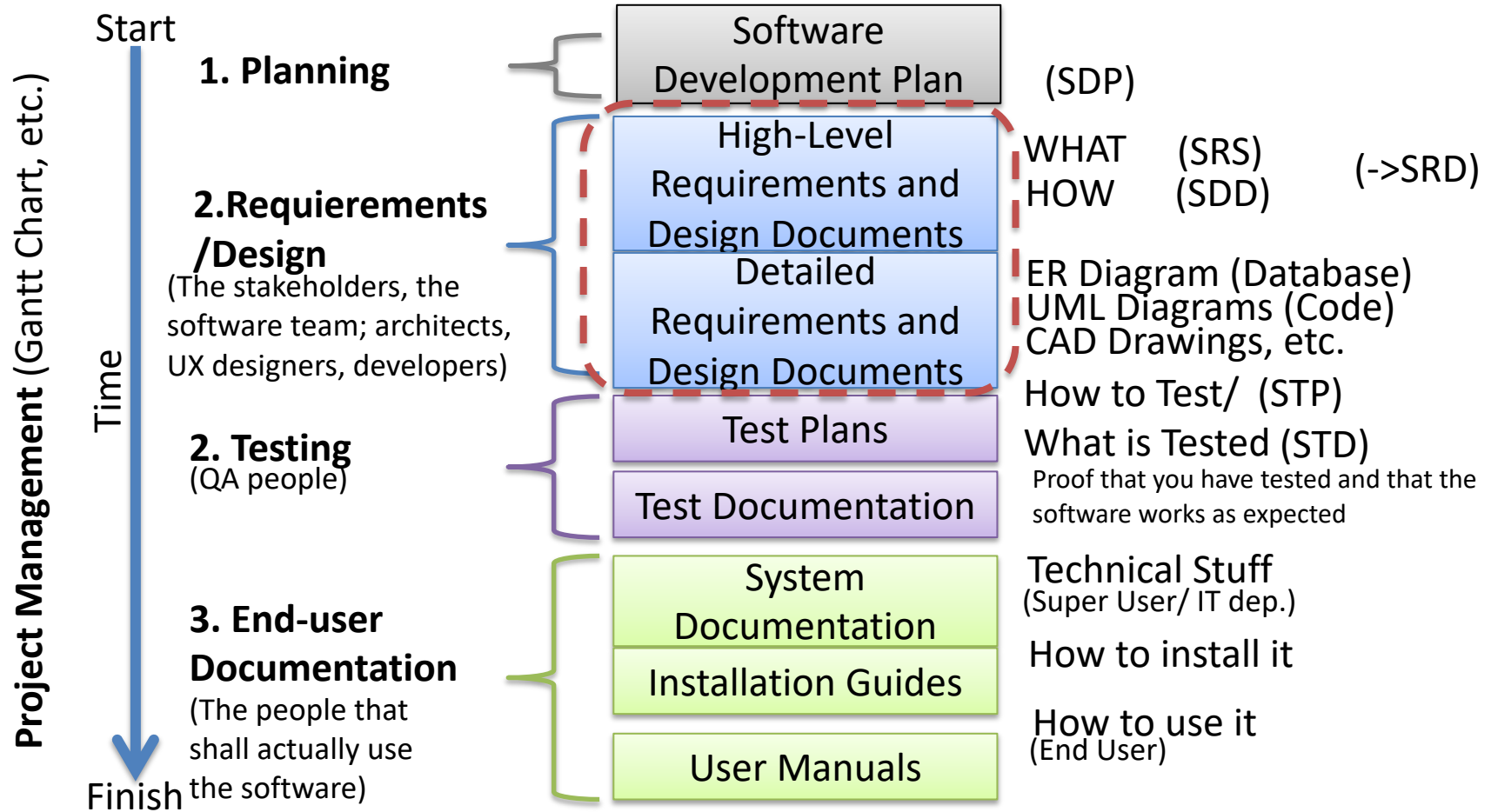
Software Design (HOW):

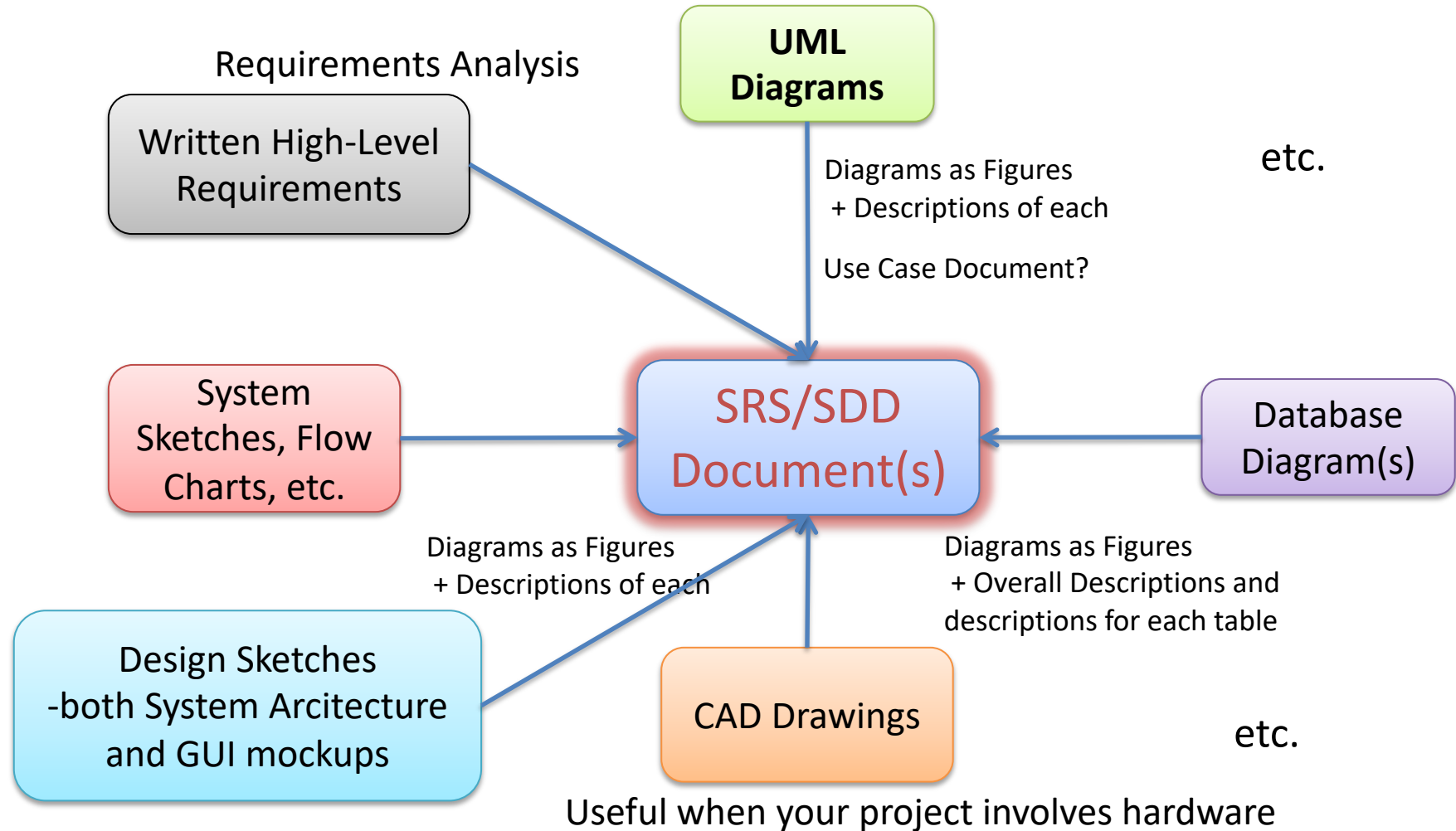
- **HOW** it should do it
- Examples: GUI Design, UML, ER diagram, CAD, etc.
- **SDD** – Software Design Document

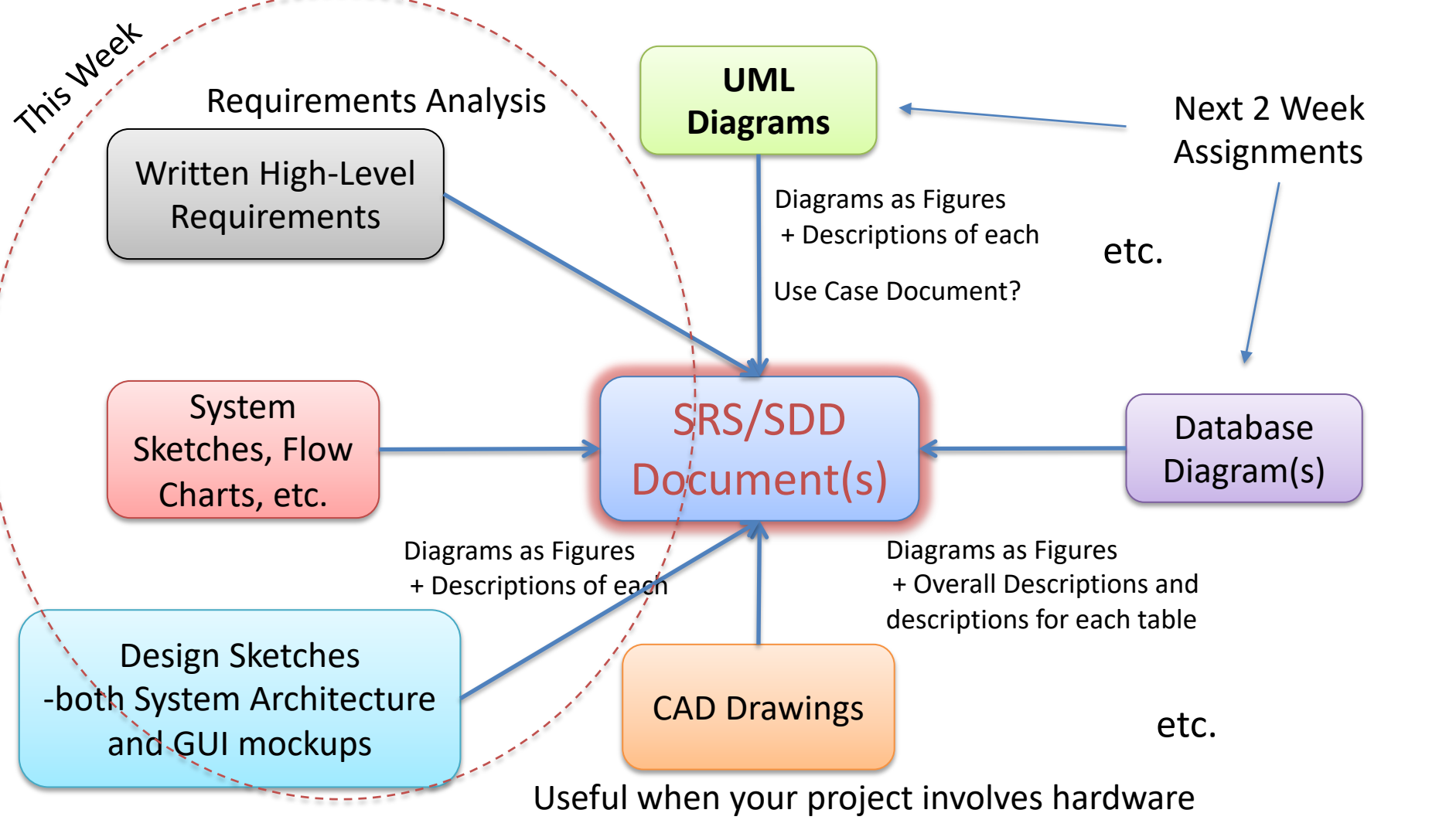
Note! Many don't separate SRS and SDD documents, but include everything in a Requirements & Design Document (**SRD** document).

In practice, requirements and design are inseparable.

Typical Software Documentation







The Structure of the SRS Document

Example 1

Chapter	Description
Preface	This should define the expected readership of the document and describe its version history, including a rationale for the creation of a new version and a summary of the changes made in each version. Les: Forord: Kort om bakgrunn for dokumentet, takke personer som har bidratt, oversikt over eventuelle endringer, osv.
Introduction	This should describe the need for the system. It should briefly describe the system's functions and explain how it will work with other systems. It should also describe how the system fits into the overall business or strategic objectives of the organization commissioning the software.
Glossary	This should define the technical terms used in the document. You should not make assumptions about the experience or expertise of the reader.
User requirements definition	Here, you describe the services provided for the user. The nonfunctional system requirements should also be described in this section. This description may use natural language, diagrams, or other notations that are understandable to customers. Product and process standards that must be followed should be specified.
System architecture	This chapter should present a high-level overview of the anticipated system architecture, showing the distribution of functions across system modules. Architectural components that are reused should be highlighted.
System requirements specification	This should describe the functional and nonfunctional requirements in interfaces to other systems may be defined.
System models	This might include graphical system models showing the relationship models are object models, data-flow models, or semantic data models.
System evolution	This should describe the fundamental assumptions on which the system is based on. This section is useful for system designers as it may help them avoid
Appendices	These should provide detailed, specific information that is related to the requirements define the minimal and optimal configurations for the system and the relationships between data.
Index	Several indexes to the document may be included. As well as a normal alphabetic index, there may be an index of diagrams, an index of functions, and so on.

Dette eksemplet er basert på IEEE sin standard om "requirements documents" og er litt omstendelig og bruker litt innfløkt språk, m.m. Mer detaljer om denne finnes i Kap.4 og i referanselista til Sommerville-boka. En slikt omstendelig standard brukes i mer formelle og omstendelige statlige prosjekter, m.m. Jeg anbefaler en lettere variant ifm. vårt prosjekt, se eksempel 2 og 3 på neste slides

The Structure of the SRS Document

- A. **System Overview** (brief description of what the software system will do)
- B. **Technical Requirements** (Functional requirement, Non-functional requirements, User-interface specification, User task flow, Input/output and other data specifications, Interface specifications to other systems)
- C. Acceptance Criteria/Interaction Scenarios
- D. Validation/Verification
- E. Requirements Considerations (Assumption made about the software, End users, Existing systems, Environment, Limitations)
- F. Other Information...

Appendix B contains lots of SRS examples and detailed descriptions

SRD Example

A mix of SRS and SDD

- Introduction
 - System Overview
 - Introduction, Description of the system, Problem Description, Sketches of the system
 - Technical Requirements
 - Functional requirement, Non-functional requirements, User-interface specification, User task flow, Input/output and other data specifications, Interface specifications to other systems
 - Architecture
 - The technical architecture of the system, system sketches, etc.
 - Database
 - Database modelling and detailed descriptions
 - UML
 - Use Case Diagrams, Sequence Diagrams, Class Diagrams
- etc.

Many don't separate SRS and SDD documents, but include everything in a Requirements & Design Document (**SRD** document). In practice, requirements and design are inseparable.

SRD Example

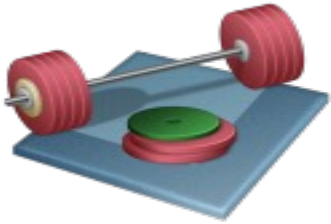
A mix of SRS and SDD

- System Overview
 - Introduction, Description of the system, Problem Description, Sketches of the system
 - Technical Requirements
 - Functional requirement, Non-functional requirements, User-interface specification, User task flow, Input/output and other data specifications, Interface specifications to other systems
 - Architecture
 - The technical architecture of the system, system sketches, etc.
 - Database
 - Database modelling and detailed descriptions
 - UML
 - Use Case Diagrams, Sequence Diagrams, Class Diagrams
- etc.

This Week!

Data & Cyber Security and GDPR

- **GDPR** - General Data Protection Regulation
- Handling Data Security and GDPR regulations (data protection and privacy) needs to be a part of the Requirements, Design and the final Solution.
- **Data & Cyber Security** Issues regarding your Software. What can/should you do to protect your Software?
- Make sure to include these Topics within your **SRD** document (and later in your Software Test Plan)
 - What do you need to do in order to follow the GDPR regulations?
 - How can you implement GDPR in your Software?
 - How can you secure your Software against threats and vulnerabilities?



Start creating the SRD document for your Project

Use the Brainstorming Notes and SRS/SRD Examples as the foundation for your SRD document

Functional and Non-Functional Requirements

Functional Requirements

- Statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations.
- May state what the system should not do.

Non-Functional Requirements

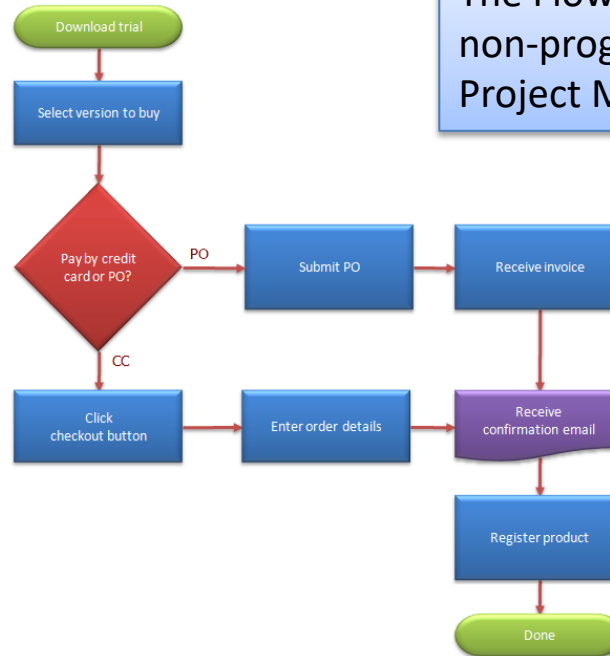
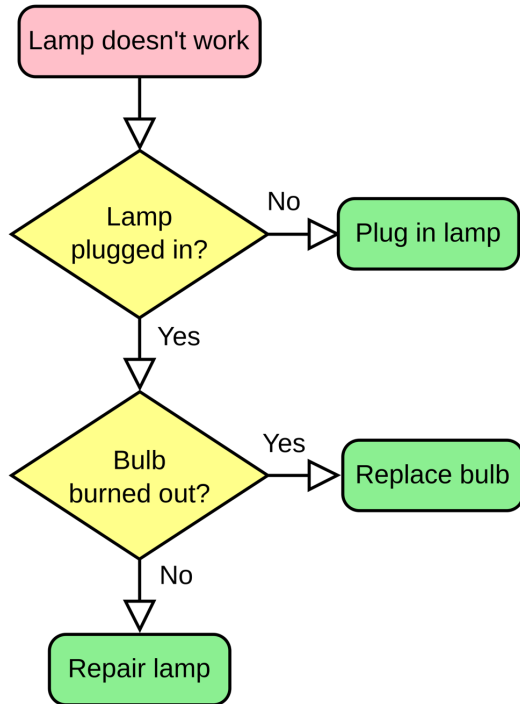
- Constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, standards, etc.
- Often apply to the system as a whole rather than individual features or services.

System Sketches

- You need to make one or more system sketches at different levels and for different users
- In Introduction Chapter
 - A basic sketch with few technical details (System Overview sketch)
 - Should be understood by all kind of readers
- In Architecture Chapter
 - One ore more sketches with more details (Technical Architecture Sketch(es))
 - For readers with more technical knowledge

Flow Chart Example

High-Level Flow Charts makes it easy to see how the system shall work

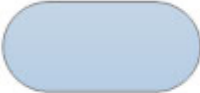

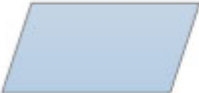




The Flow Charts should be understood by non-programmers like the Stakeholders, Project Managers and Customers

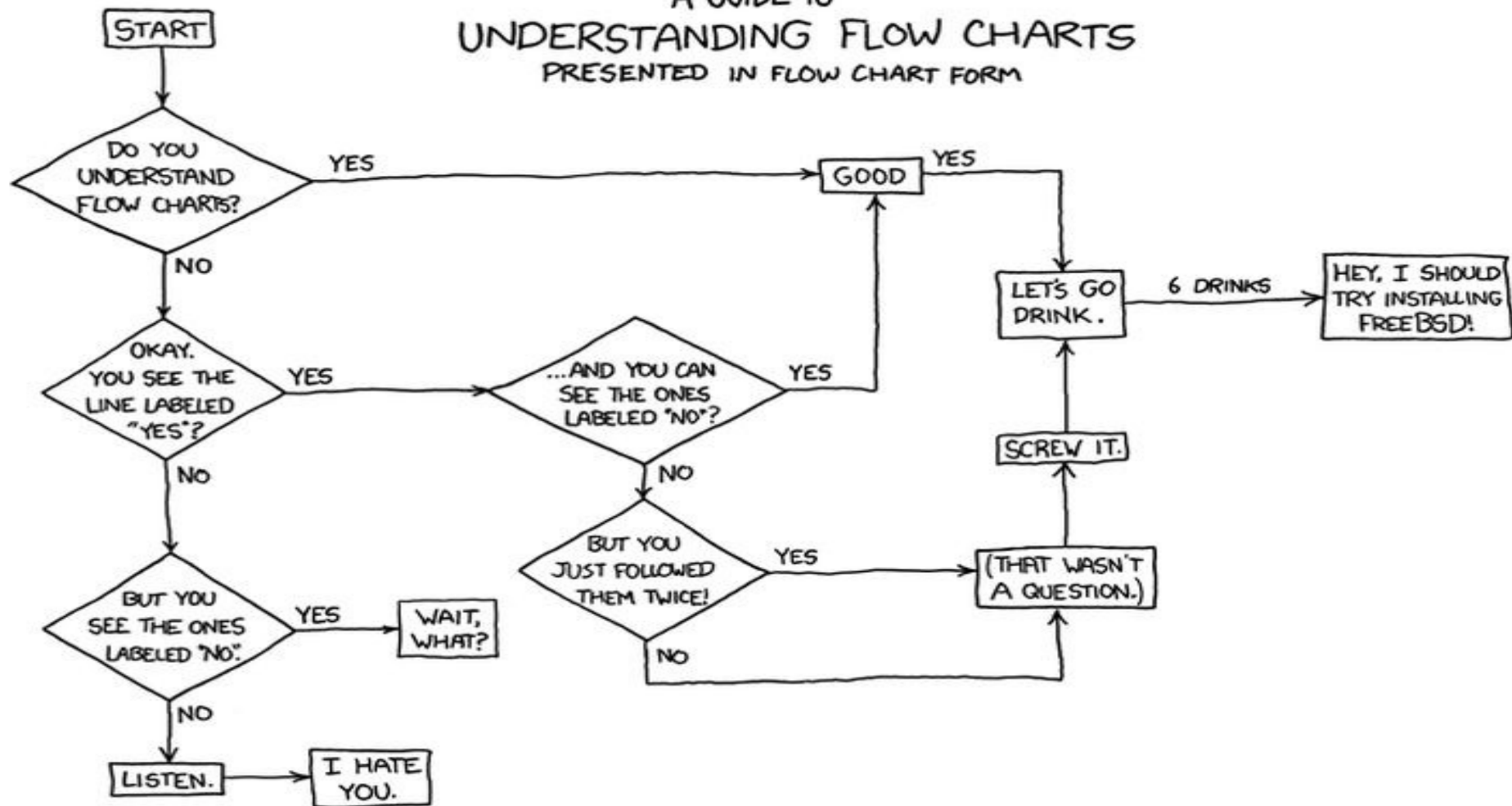
Note! Later we will create more detailed diagrams using **UML** Modelling

MS Visio or MS PowerPoint has built in features for creating Flow Charts

Flow Chart Symbols

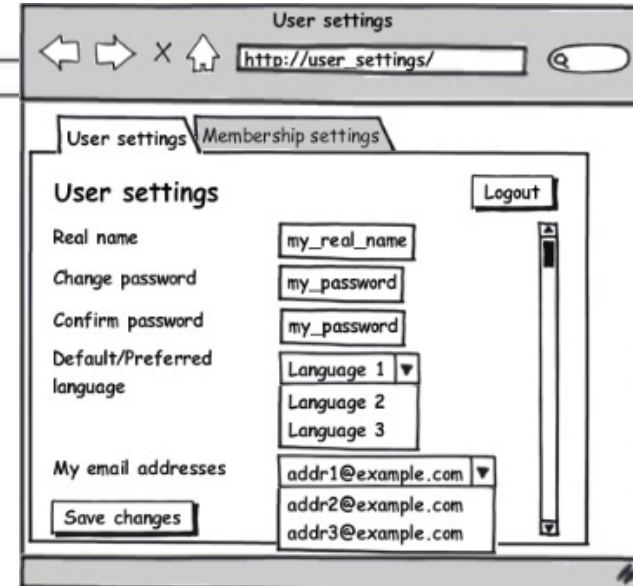
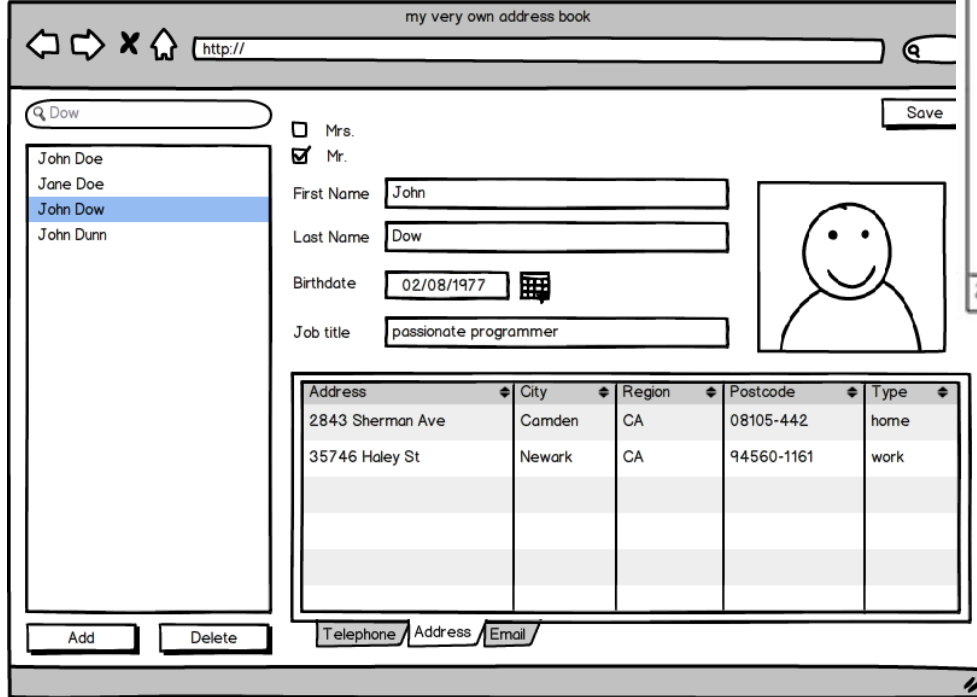
Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision

A GUIDE TO UNDERSTANDING FLOW CHARTS PRESENTED IN FLOW CHART FORM



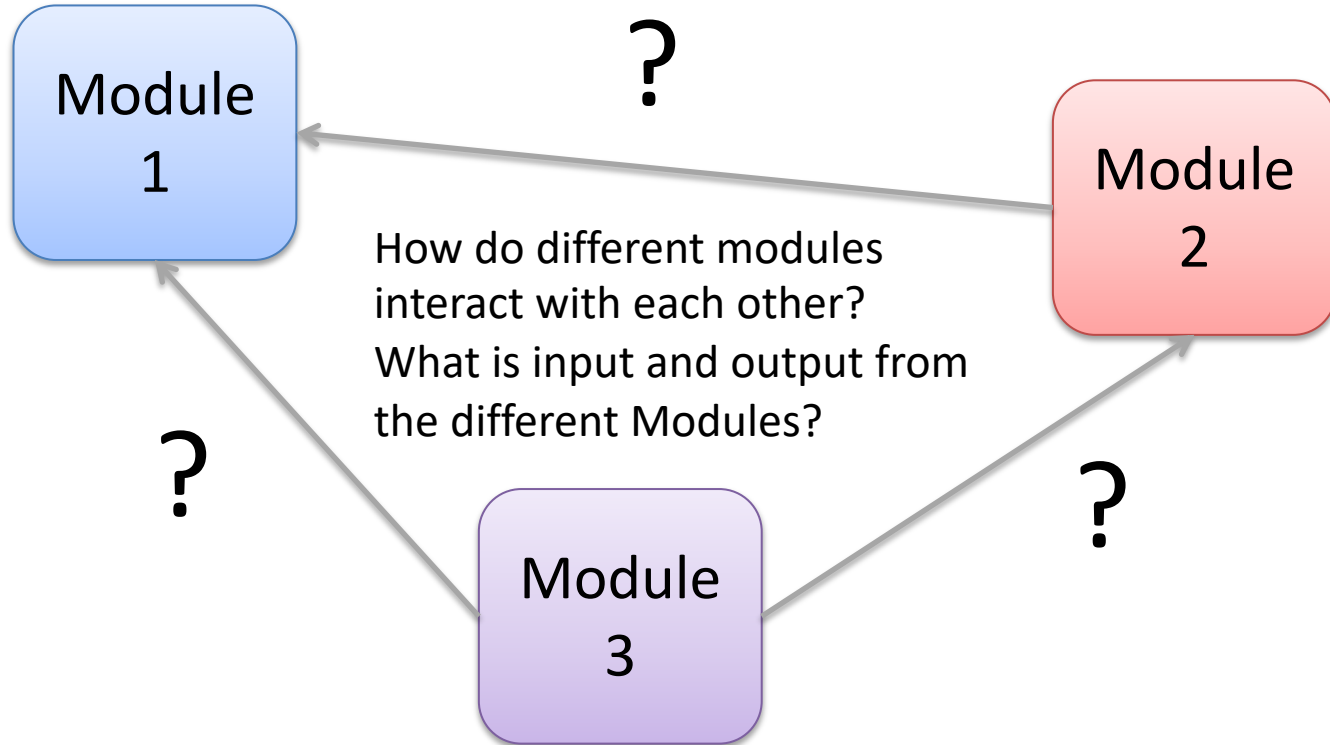
GUI Design Sketches

“Mockups”



You should also start creating some GUI sketches that you can in the SDD/SRD document

Interface specifications



Check List

- The **standards and guidelines** available in the organization are followed in the document.
- Will the requirements meet the **customer's need**?
- Are all **functional, nonfunctional and interface requirements** captured for the system?
- Is each requirement **detailed enough** and supported by necessary diagrams, figures, data and use cases so that all the stakeholders get their necessary input from the requirement?
- Are there requirements that **conflict with each other**?
- Are the requirements **verifiable/testable**?
- How much **dependency** does the requirement have on other requirements?
- Are the requirements validated and verified by all the **stakeholders** (including all members of the Development Team)?



Project Plan (Gantt Chart)

Project Plan (Gantt Chart)

- Create/Update the Gantt Chart using MS Project for your Software Project. **It should be included in the Software Development Plan (SDP)**
- Important Milestones, Deadlines and Meetings should be part of the Project Plan (see the course schedule)
 - Alpha Release (Sprint Iteration 1)
 - Beta Release (Sprint Iteration 2)
 - RC Release (Sprint Iteration 3)
 - RTM Release (Sprint Iteration 4)
- Use the Software Requirements and Design document(s) as background information when creating the Gantt Chart.
- Break Requirements down to Tasks and Subtasks and set who is Responsible for each of the Tasks + Time Estimate

See Next Slides for more details...

Project Task Estimation

How many hours does it take to do a specific Task?

- The Features and Requirements need to be broken down into manageable Tasks by the team
- Each Tasks then needs to be Estimated (Hours)
- In the beginning of the project, we make roughly estimates
- Then week by week we break it into more details and are able to do more precise estimations

Note!! Each Task should have only one Responsible Person



Azure DevOps

Azure DevOps

- Add your Releases (Alpha, Beta, RC, RTM) as **Iterations** in the system. You should also add **Areas** and a structured **Folder Structure**
- Get an overview of **Work Items** in Azure DevOps.
- Add your High-level Software Requirements and Design Items as Work Items in Azure DevOps (**Product Backlog**).
- Select some of them to be part of Sprint 1/Alpha (**Sprint Backlog**). Make a rough estimate for each task.

See Next Slides for more details...

Azure DevOps - New Project

My organizations

- A** alarmsystem
- B** bachelor-v17-imsephi
- C** CheckpointAS
- E** ees17
- O** olavd
- R** Rutor
- S** software-usn
- S** systemutviklingogdokumentasj...

Related pages

- [What's new in DevOps](#)
- [Documentation](#)
- [Get help](#)

- [+ New organization](#)
- [Organization settings](#)

software-usn

Projects My work items My pull requests

E EnvironmentalPublicHealth

All projects

- E** EnvironmentalPublicHealth
- SE** Software Engineering
- T** TestProject
This is a test project used to test the functionality in VSTS

Create new project

Project name *

MySoftware

Description

Visibility



Public

Anyone on the internet can view the project. Certain features like TFVC are not supported.



Private

Only people you give access to will be able to view this project.

Public projects are disabled for your organization. You can turn on public visibility with [organization policies](#).

Advanced

Version control ?

Team Foundation Version Control

Work item process ?

Scrum

Make sure to select these settings!!!

Create

Cancel

<https://dev.azure.com>

My Project



MySoftware



Full access to Azure Pipelines for Stakeholders: Give Stakeholders full access to Azure Pipelines for private projects. [Learn more](#)

Try it!

Not now

Overview

Summary

Dashboards

Wiki

Boards

Repos

Pipelines

Test Plans

Artifacts

Project settings



MySoftware

Private

Invite



About this project

Like 0



Development of MySoftware

Project stats

Last 7 days

Repos

1

Changesets by 1 authors

Members 1

Invite members to MySoftware

Search and add users to your project

Add users or groups *

Search users

Add to team(s) *

MySoftware Team

Add

Cancel

Azure DevOps in Visual Studio

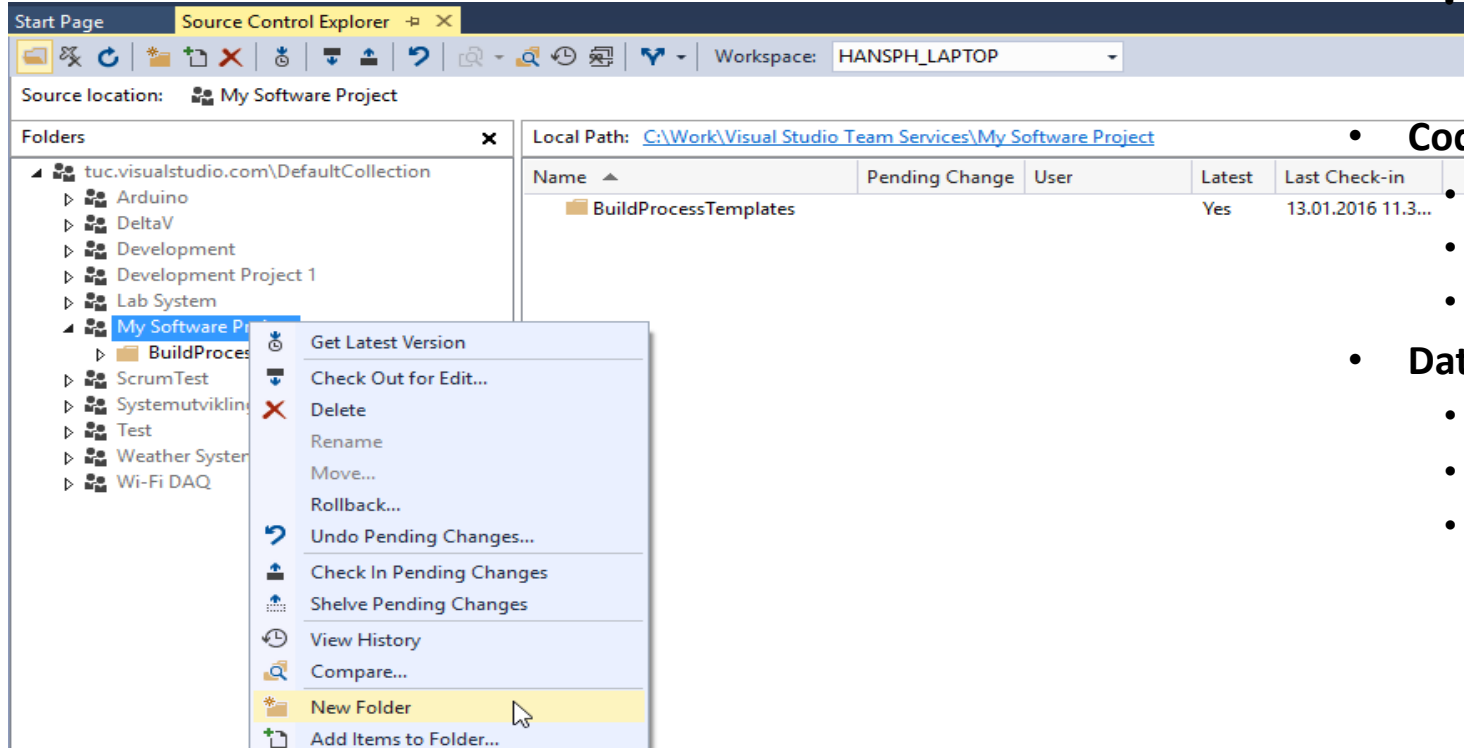
The screenshot displays the Visual Studio interface with the following components:

- Source Control Explorer:** Shows a folder tree for 'Development' with subfolders like 'BuildProcessTemplates', 'Code', 'Documents', and 'Project Management'. A callout box points to this structure with the text: "Create a good Folder structure for your Documents and the Source Code".
- Team Explorer:** Shows 'Pending Changes' for 'Development' with a table of changes.
- Pending Changes Panel:** Shows a table of pending changes with columns for Name, Pending Change, User, Latest, and Last Check-in.

Name	Pending Change	User	Latest	Last Check-in
BuildProcessTemplates			Yes	2013-04-26 12:...
Code			Yes	2013-05-31 12:...
Documents			Yes	2013-06-20 10:...
Project Management			Yes	2014-01-15 9:2...

Folder Structure Example

- My Project
 - Documents
 - Process Documents
 - Product Documents
 - System Documents
 - User Documents



- Code
 - Desktop
 - Web
 - Server-side
- Database
 - Tools
 - Design
 - Scripts
 - Functions
 - Scripts
 - Stored Procedures
 - Tables
 - Triggers
 - Views

Areas & Iterations (Sprints)

Project Settings > Project configuration

General

Boards

This project is currently using the Scrum process. To customize your work item types, [go to the process customization page.](#)

Overview

Iterations

Areas

Teams

Create and manage the areas for this project. These areas will be used by teams to determine what shows up on the team's backlog and what work items the team is responsible for. [Learn more about customizing areas and iterations](#)

Security

To select areas for the team, go to [the default team's settings.](#)

Notifications

Service hooks

New New child + -

Project Settings > Project configuration

Dashboards

Areas

Teams

General

Boards

This project is currently using the Scrum process. To customize your work item types, [go to the process customization page.](#)

Boards

MySoftware

MySoft

Overview

Iterations

Areas

Create and manage the iterations for this project. These iterations will be used by teams for iteration planning (sprint planning). [Learn more about customizing areas and iterations](#)

Project configuration

Database

Teams

Security

Notifications

Service hooks

Dashboards

Team configuration

Stored Procedures

GitHub connections

Tables

Desktop

DesktopApp1

Documentation

Pipelines

Service connections

Boards

Project configuration

Team configuration

GitHub connections

Pipelines

Service connections

New New child + -

Iterations Start Date End Date

MySoftware

Alpha

Beta

Beta1

The different software modules could be divided into different Areas.

It is important to have a good structure from the beginning!!

Create Iterations (in Scrum they use Sprints) for the different releases, milestones (internal and external); e.g., Alpha, Beta, RC, RTM

Create Product Backlog Items

Azure DevOps

software-usn / MySoftware / Boards / Backlogs

MySoftware

Overview

Boards

Work Items

Boards

Backlogs

Sprints

MySoftware Team

+ New Work Item

Product Ba...

Add to top

	Order	Work Item Type	Title	State	Effort	Value Area
--	-------	----------------	-------	-------	--------	------------

+ 1	Product Back...	The system should store the data in a Dataabse	...	New		Business
-----	-----------------	--	-----	-----	--	----------

What is the Product Backlog?

- A Term Used in Agile/Scrum
- The Product Backlog is an ordered list of everything that might be needed in the product -> Requirements
- It is the single source of requirements for any changes to be made to the product.



Coding and Implementation

Coding and Implementation

- Start Planning the code structure of your Application(s)/ Module(s).
- Install necessary Software
- Get an overview of the software platforms, programming languages you shall use, etc.
- Consider start creating the main shell for your application (both code and GUI). (Test that you can communicate with a Database, etc., DB Design starts next week)
- It is important that we always have a working software (so it can be reviewed, tested, etc. during the whole project)! This one of the basic feature of Scrum

See Next Slides for more details...

ASP.NET Core

Web Page: <https://halvorsen.blog/documents/programming/web/aspnet>

Videos:

- ASP.NET Core – Introduction
<https://youtu.be/zkOtiBcwo8s>
- ASP.NET Core – Database Communication
<https://youtu.be/0Ta3dQ3rxzs>
- ASP.NET Core - Database CRUD Application
<https://youtu.be/k5TCZDwTYcE>
- ASP.NET Core – Class Library
<https://youtu.be/emUiMd1zRrY>
- ASP.NET Core – Charts
<https://youtu.be/mksUls9fx-Q>
- ASP.NET Core – Session Data
https://youtu.be/I0SQ_XAoFvA

Web Programming ASP.NET Core

Hans-Petter Halvorsen



<https://www.halvorsen.blog>

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: <https://www.halvorsen.blog>

