Software Engineering

Course Introduction

Hans-Petter Halvorsen

What is Software Engineering?

Software Engineering is the profession of the Development and Management of High Quality Software Systems within given Time and Cost frames

Software Engineering

Main purpose with this course:

> “To take you beyond Programming to actually Engineering Software”

Planning, Requirements Analysis, Software Design and Architecture, Software Testing, Software Deployment, etc.

Learning Goals

- You will be familiar with the most important features of different System Development Processes,
- Making the transition to working life easier
- You will learn tools to communicate different aspects of an abstract system. Both to customers and employees
- You will learn how to organize a Software Development Project

Learning Goals - Details

We will learn how to build good (i.e. high quality) software, which includes:

- Requirements Specification (Requirements Engineering/Analysis)
- Technical or Architectural Design
- Good User Experience (UX)
- Improved Code Quality and Implementation
- Testing
- System Documentation and User Documentation
- Deployment

Why Software Engineering?

- There are many differences between a one-person programming and large software system development.
- The degree of complexities between these two approaches make it necessary to bring more discipline into the development process.
- Modern software engineering is very complex and there are large numbers of failures in many software projects and defects encountered in the software products.
- All infrastructure for human livings rely on Software today (Traffic Systems, Financial systems)
- That's why Software Engineering is needed

Software Engineering Course

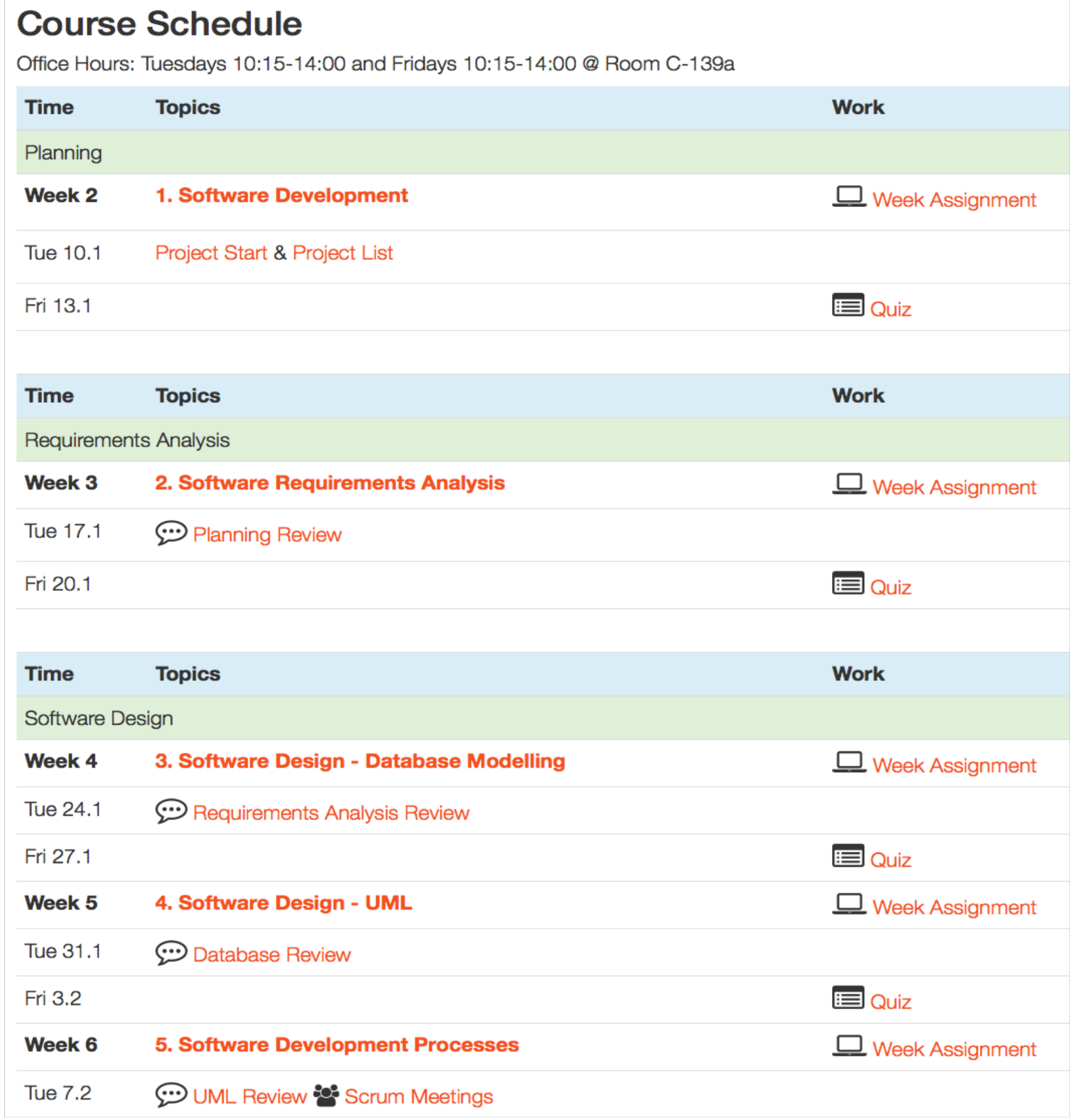
Canvas:

https://usn.instructure.com

Course Information, Course Schedule, Teaching material, Videos, etc.

https://www.halvorsen.blog/documents/teaching/courses/software_engineering.php

Course Schedule



Course Schedule

Office Hours: Tuesdays 10:15-14:00 and Fridays 10:15-14:00 @ Room C-139a

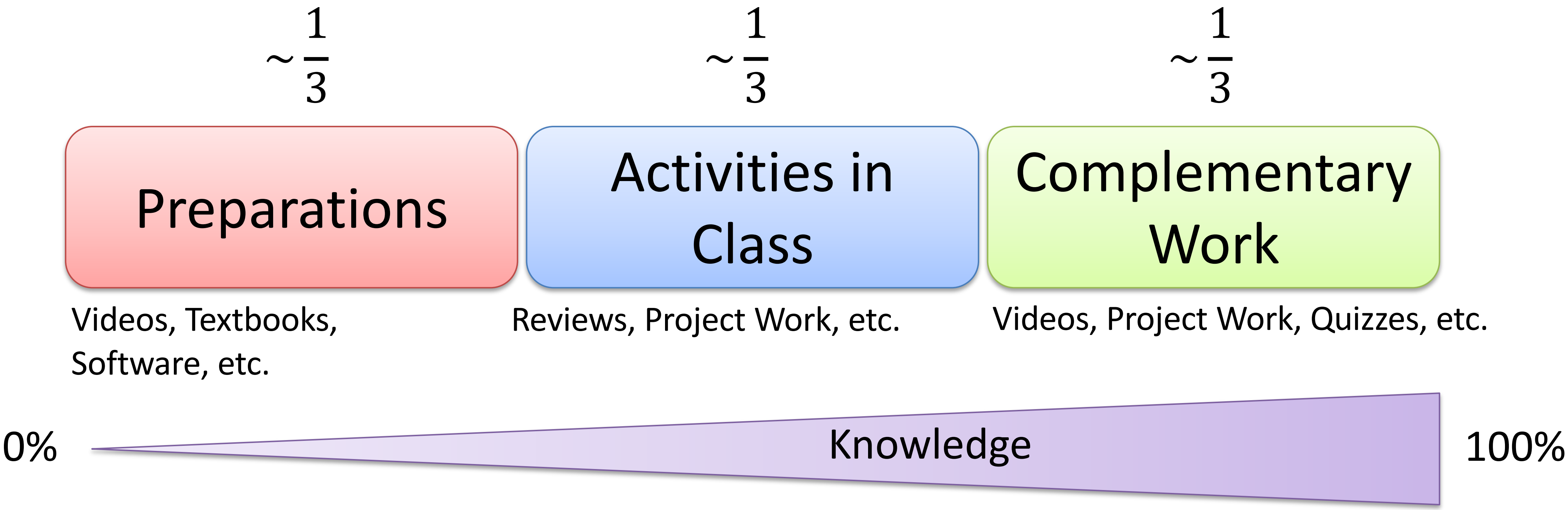
Time	Topics	Work
Planning		
Week 2	**1. Software Development**	Week Assignment
Tue 10.1	Project Start & Project List	
Fri 13.1		Quiz

Time	Topics	Work
Requirements Analysis		
Week 3	**2. Software Requirements Analysis**	Week Assignment
Tue 17.1	Planning Review	
Fri 20.1		Quiz

Time	Topics	Work
Software Design		
Week 4	**3. Software Design - Database Modelling**	Week Assignment
Tue 24.1	Requirements Analysis Review	
Fri 27.1		Quiz
Week 5	**4. Software Design - UML**	Week Assignment
Tue 31.1	Database Review	
Fri 3.2		Quiz
Week 6	**5. Software Development Processes**	Week Assignment
Tue 7.2	UML Review Scrum Meetings	

https://www.halvorsen.blog/documents/teaching/courses/software_engineering.php

The Learning Process



$\sim\frac{1}{3}$

Preparations

Videos, Textbooks, Software, etc.

$\sim\frac{1}{3}$

Activities in Class

Reviews, Project Work, etc.

$\sim\frac{1}{3}$

Complementary Work

Videos, Project Work, Quizzes, etc.

0%

Knowledge

100%

Most of the learning takes place outside the classroom: Do not forget the gain by being well prepared when you get to the classroom, as well as work with the material afterwards. Learning takes place where you do preparations and process the material afterwards. You need to work actively with the contents.

Software Engineering Course

- This is a very practical course with few traditional lectures
- Instead there are lots of practical work, both individual and in teams.
- Most of the theory and practical examples are provided as videos and well-written text books
- In class we will work with the Project and so-called Week Assignments.

Topics

- Software Planning, Project Management
- Requirements Engineering/Analysis
- Database Modeling
- UML (Unified Modeling Language)
- Software Development Processes (Waterfall, Agile Development, Scrum)
- Software Platforms (Desktop, Mobile, Web, Cloud, ..)
- Software Architecture
- Software Implementation
- Source Code Control and Bug Tracking
- Software Testing
- Software Documentation
- Software Deployment and Maintenance

Software

We will use many tools you are already familiar with, and use more advanced features in these tools

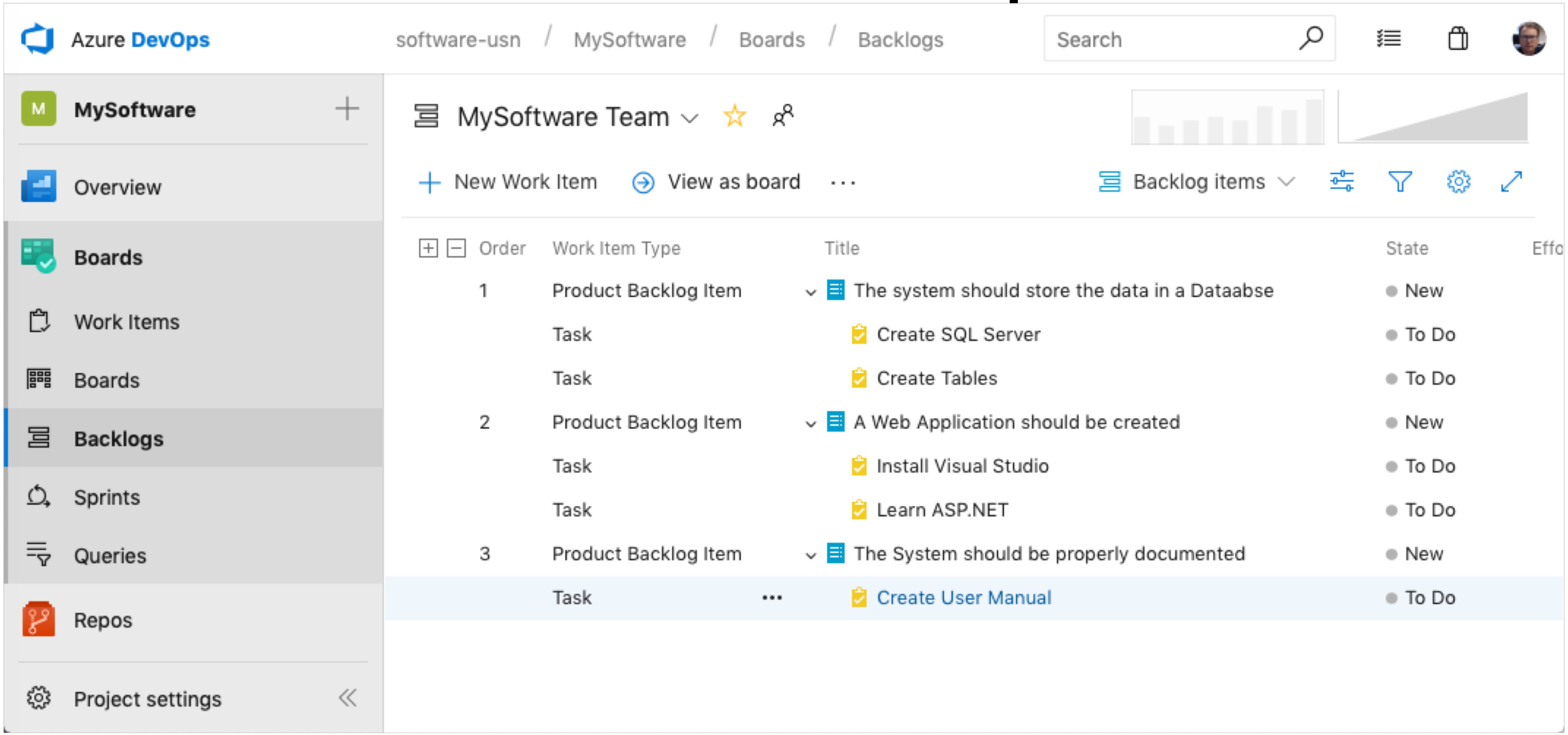
- Office 365, including MS Project
- Visual Studio (Enterprise Edition)
  - C#
  - ASP.NET (Framework for Web Development)
- Azure DevOps (previously Visual Studio Team Services)
- ERwin Database Modeling Tool
- SQL Server
- Microsoft Azure (Windows running in the Cloud)
- …

Web and Cloud

- We will focus on modern Web Applications and Cloud Platforms in this course
- We will use ASP.NET
- ASP.NET is a Web Framework for creating Web Applications developed Microsoft and integrated in Visual Studio
- Since you are already familiar with Win Forms, the transmission to ASP.NET Web Forms should be easy. We will use C# when creating ASP.NET Web Applications also, and the programming model is very similar
- We will also use Microsoft Azure

Azure DevOps

www.visualstudio.com



Order	Work Item Type	Title	State	Effo
1	Product Backlog Item	The system should store the data in a Dataabse	New	
	Task	Create SQL Server	To Do	
	Task	Create Tables	To Do	
2	Product Backlog Item	A Web Application should be created	New	
	Task	Install Visual Studio	To Do	
	Task	Learn ASP.NET	To Do	
3	Product Backlog Item	The System should be properly documented	New	
	Task	Create User Manual	To Do	

Software Engineering Tool for Project Management, Scrum, Source Code Control, Bug Tracking, Collaboration, etc.

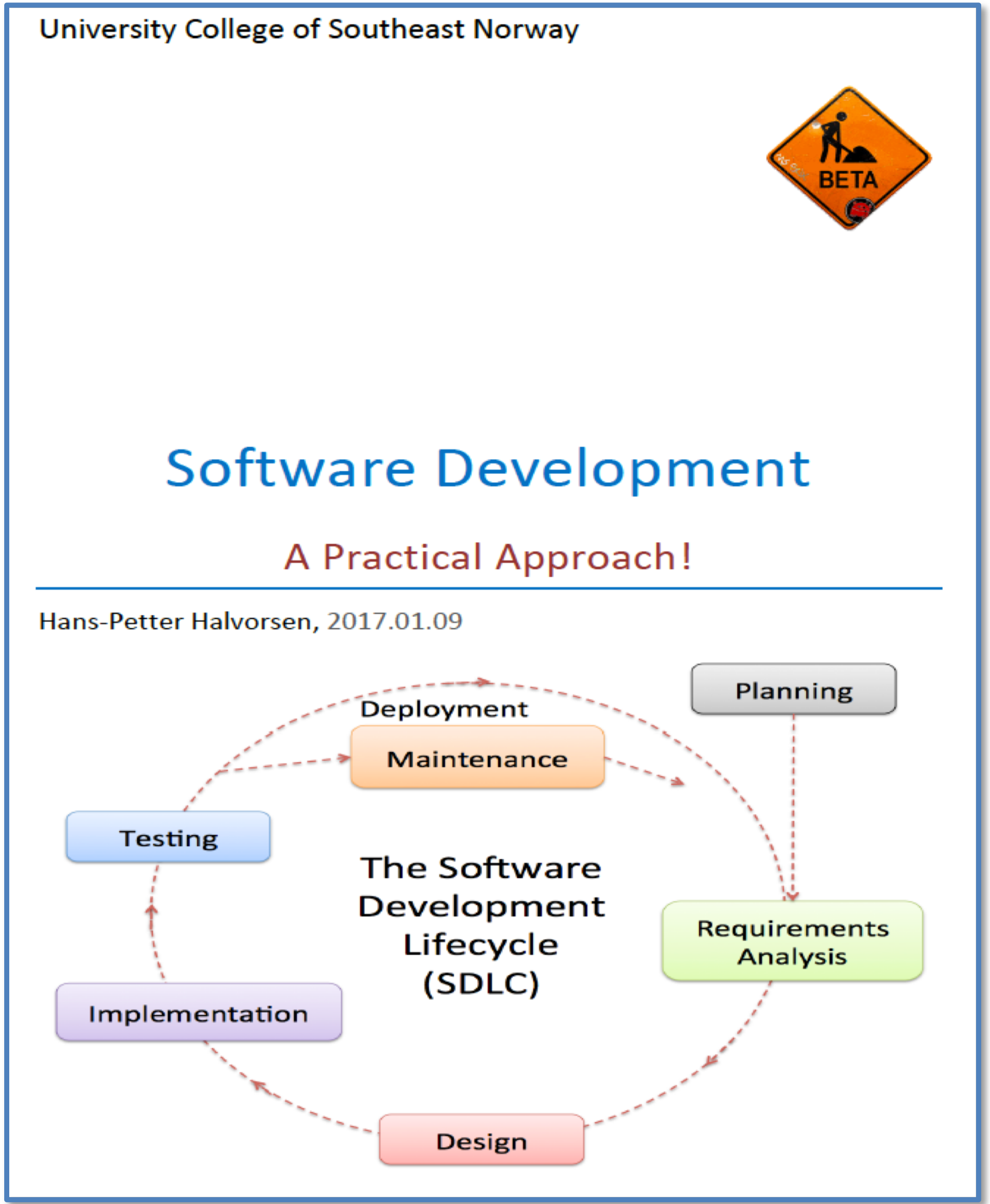
Azure DevOps

- Formerly “Visual Studio Team Services” (VSTS)
- Azure DevOps is a Source Code Control (SCC), Bug Tracking, Project Management, and Team Collaboration platform from Microsoft
- Integrated with Visual Studio
- Web-based Project Management, including Scrum
- Free for up to 5 users

www.visualstudio.com

Literature/Textbooks

- Software Development - A Practical Approach
  Halvorsen, Hans-Petter, 2018
- Software Engineering
  I. Sommerville, 10 ed.: Pearson, 2015
- Essentials of Software Engineering
  Frank Tsui; Orlando Karam; Barbara Bernal, 4 ed., Jones & Bartlett Learning



University College of Southeast Norway

BETA

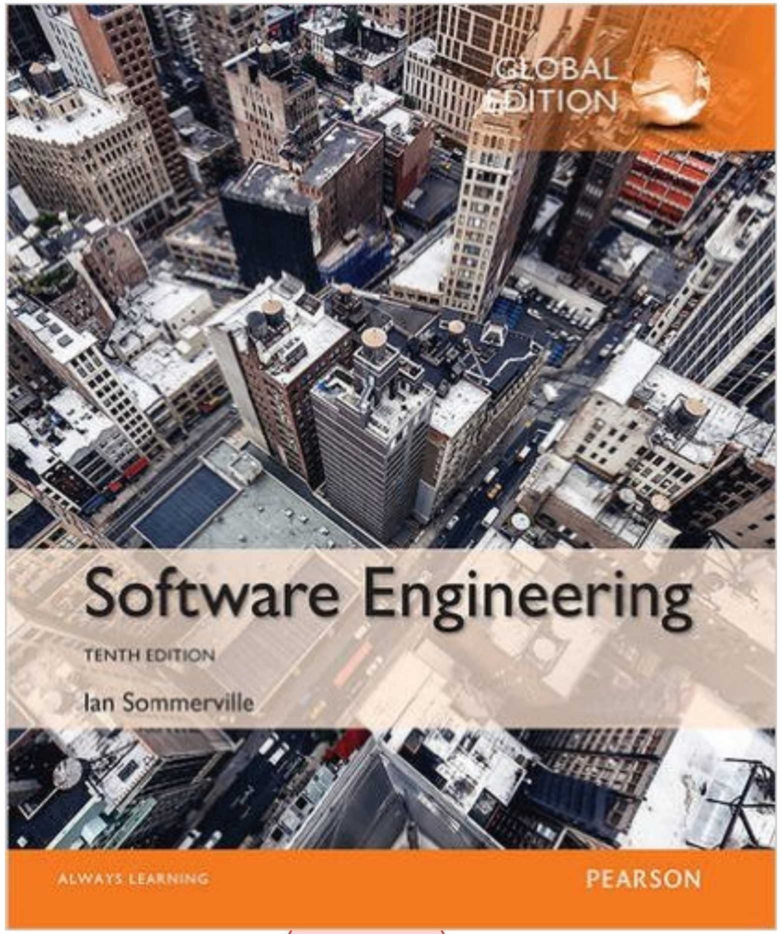
Software Development

A Practical Approach!

Hans-Petter Halvorsen, 2017.01.09

Free Download
(PDF)

Books

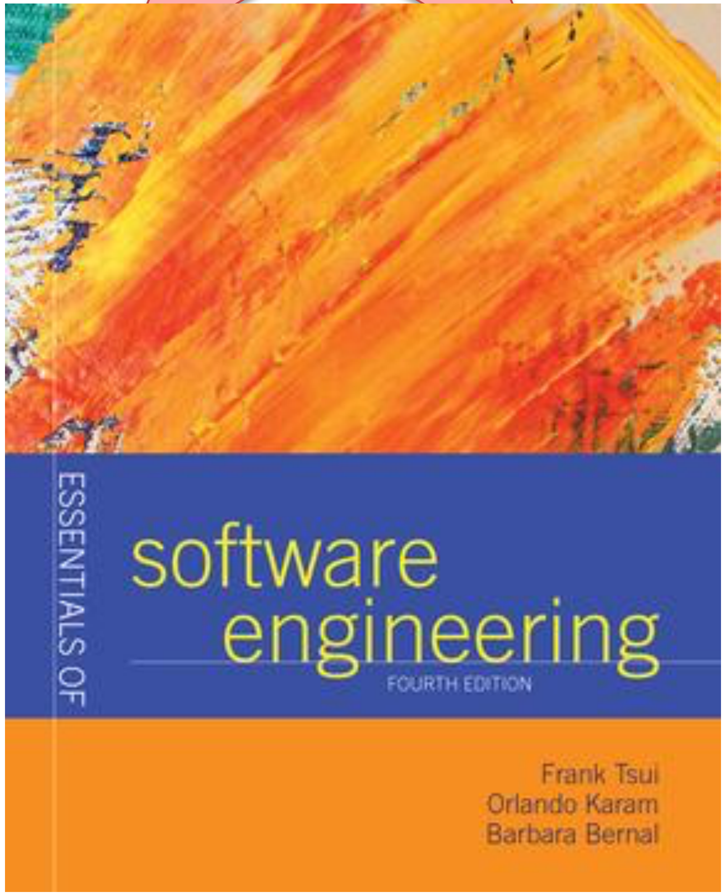


Software Engineering

I. Sommerville, 10 ed.: Pearson, 2015

Recommended to **Buy**!

Available Online!



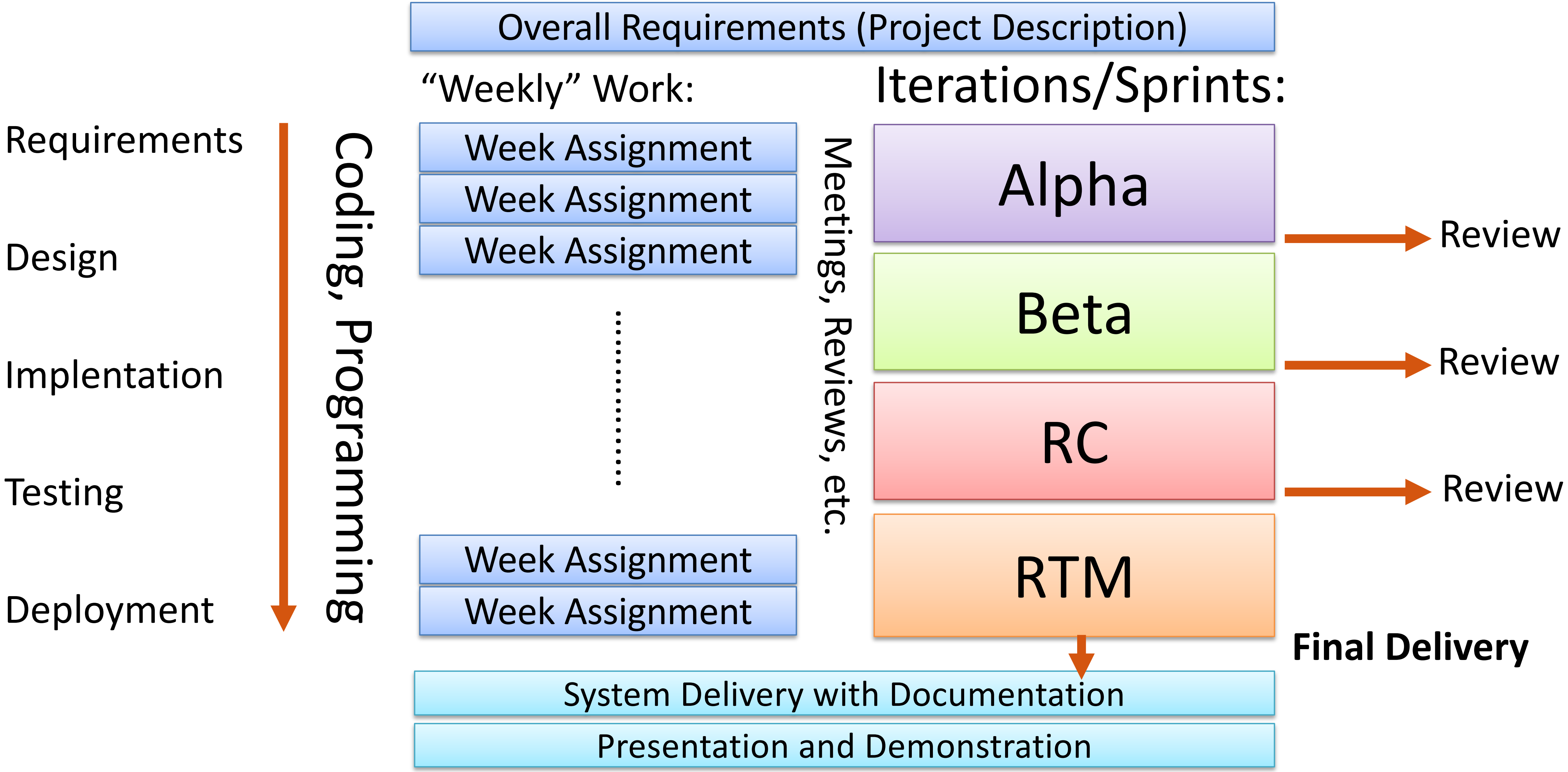
Essentials of Software Engineering

Frank Tsui; Orlando Karam; Barbara Bernal, 4 ed., Jones & Bartlett Learning

Project

- A large Project will be the glue in this course, at it will last the whole semester, beginning already the first week.
- 3-4 Students in each Team
- You will be given Week Assignments and work through the Project week by week the whole semester

Team Project Milestones



Overall Requirements (Project Description)

"Weekly" Work:

Iterations/Sprints:

Requirements

Design

Implentation

Testing

Deployment

Coding, Programming

Week Assignment

Week Assignment

Week Assignment

Week Assignment

Week Assignment

Meetings, Reviews, etc.

Alpha

Review

Beta

Review

RC

Review

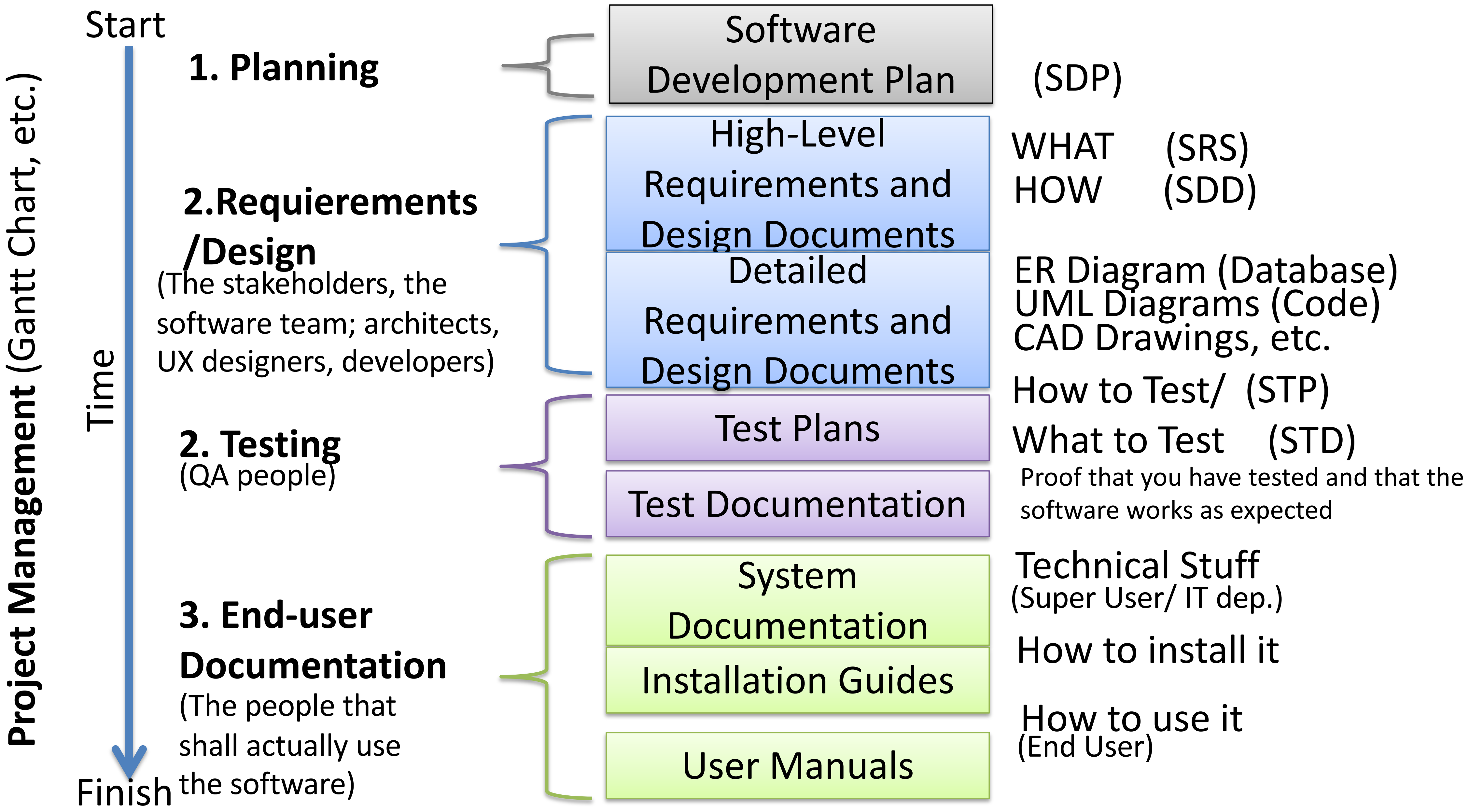
RTM

Final Delivery

System Delivery with Documentation

Presentation and Demonstration

Typical Software Documentation



Project Management (Gantt Chart, etc.)

Time

Start

1. Planning

Software Development Plan (SDP)

2.Requierements /Design
(The stakeholders, the software team; architects, UX designers, developers)

High-Level Requirements and Design Documents

WHAT (SRS)
HOW (SDD)

Detailed Requirements and Design Documents

ER Diagram (Database)
UML Diagrams (Code)
CAD Drawings, etc.

2. Testing
(QA people)

Test Plans

How to Test/ (STP)
What to Test (STD)

Test Documentation

Proof that you have tested and that the software works as expected

3. End-user Documentation
(The people that shall actually use the software)

System Documentation

Technical Stuff
(Super User/ IT dep.)

Installation Guides

How to install it

User Manuals

How to use it
(End User)

Finish

Level of complexity in your solution?

The 80 – 20 Rule:

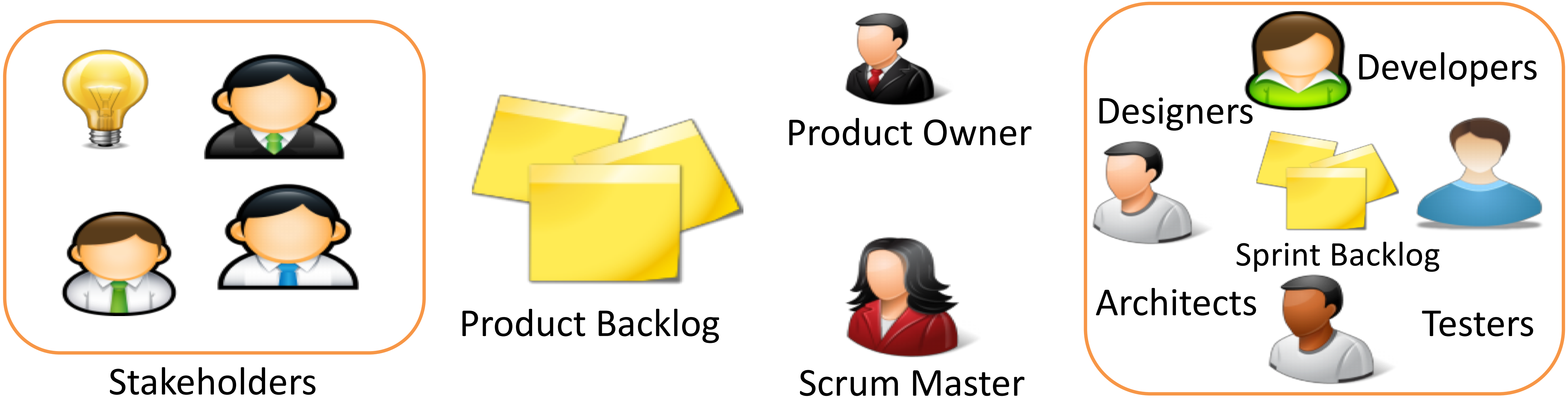
- It takes 20% of the time to finish 80% of your application -> Prototype (80% finished)
- 80% of the users only use 20% of the features

Conclusion:

- Someone always tends to make things more complicated than necessary!
- The main goal in this Project and Course is to make a functional Prototype! – Not a fully working professional Product ready for sale
- Estimated Hours: 270 hours

Scrum

Scrum Members:



Stakeholders

Product Backlog

Product Owner

Scrum Master

Designers

Developers

Sprint Backlog

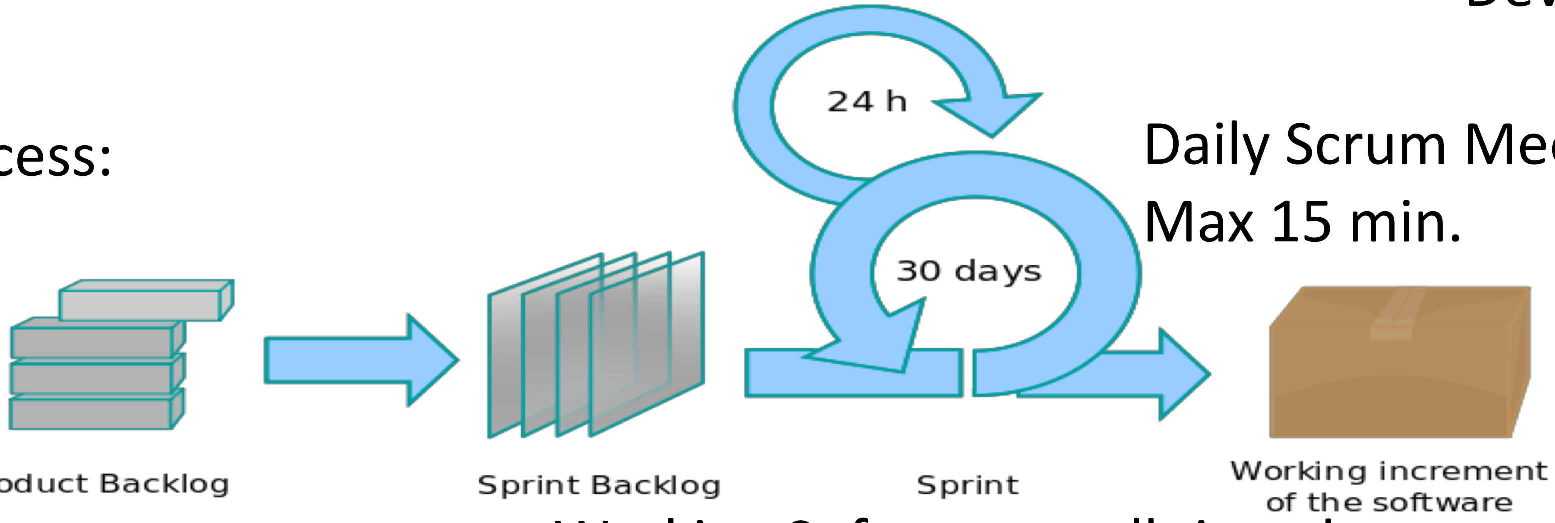
Architects

Testers

Development Team

3-9 persons

Scrum Process:



Daily Scrum Meetings

Max 15 min.

Sprint Review

A Framework for Software Development - Working Software at all times!

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

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

Web: http://www.halvorsen.blog