Software Technology 01

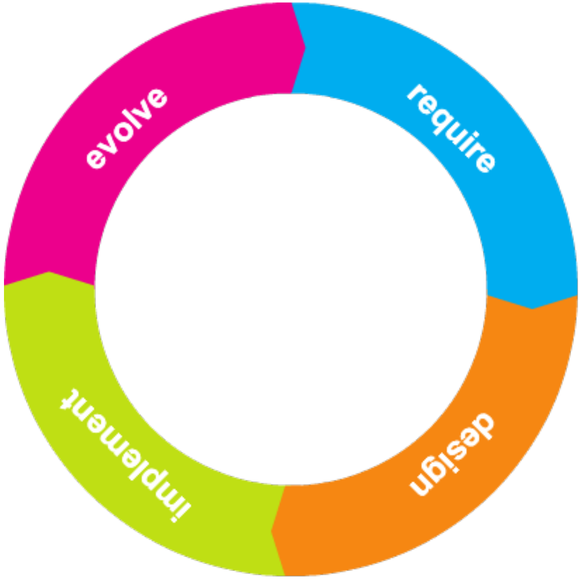
Course Introduction

What is Software Technology?

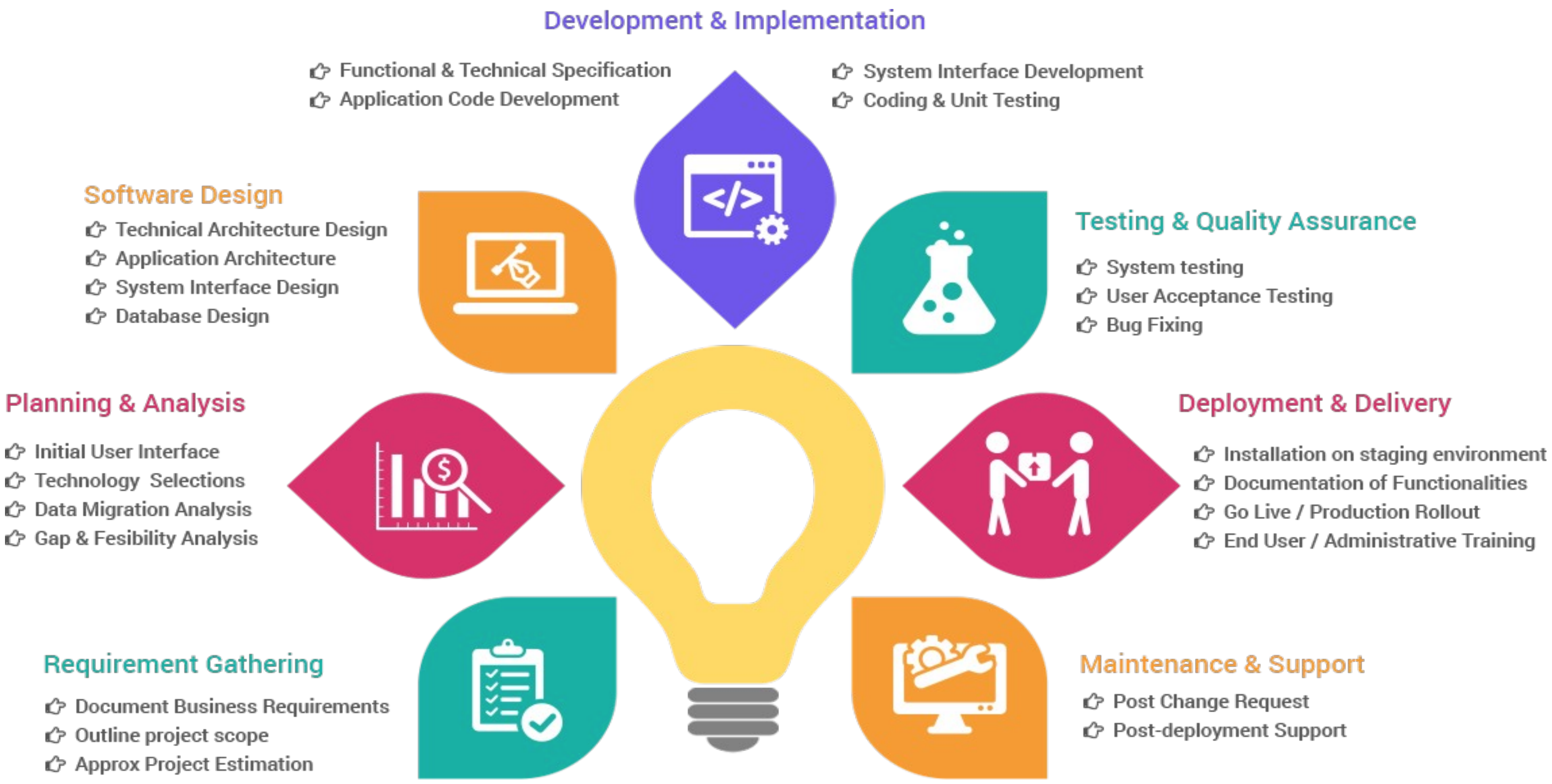
- Course? Faculty? Program? → too broad
- SW Engineering? SW Development?
- „**Software development** is the process of computer *programming*, *documenting*, *testing*, and *bug fixing* involved in creating and *maintaining* applications and frameworks resulting in a **software product**.”

- Ultimate goal: Software Product
- But how?

Software Development



Software Development Detailed



Requirement Gathering

- Document Business Requirements
- Outline project scope
- Approx Project Estimation

Planning & Analysis

- Initial User Interface
- Technology Selections
- Data Migration Analysis
- Gap & Fesibility Analysis

Software Design

- Technical Architecture Design
- Application Architecture
- System Interface Design
- Database Design

Development & Implementation

- Functional & Technical Specification
- Application Code Development
- System Interface Development
- Coding & Unit Testing

Testing & Quality Assurance

- System testing
- User Acceptance Testing
- Bug Fixing

Deployment & Delivery

- Installation on staging environment
- Documentation of Functionalities
- Go Live / Production Rollout
- End User / Administrative Training

Maintenance & Support

- Post Change Request
- Post-deployment Support

Contradictions

- Do we **optimize** for
  - Release date?
  - Unique product which is hard to copy?
  - Quality?
  - User Experience? (UX)
  - Developer Experience?
  - Easy Maintenance?
  - Easy Deployment?
  - Easy Usage and Compatibility?
  - Vendor Lock-In? :)
  - Marketable product?
    (what to write on the box, number of features)
  - Platform independence?
  - Performance?
  - Cost?
  - Least amount of useless work?

Software Development can be ~~Good~~ **Fit for the job**

SW Development History

- Software Development
  - Processes
  - Methodologies
  - Philosophies →
  - & Trends

- Acceptance test–driven development
- After the Software Wars
- Agile Manifesto
- Agile software development
- Pull-based agile coaching
- Behavior-driven development
- Best practice
- The Cathedral and the Bazaar
- Comment programming
- Composition filters
- Cowboy coding
- Design-driven development
- Domain-driven design

- Extreme programming
- Formal methods
- Homesteading the Noosphere
- Integration competency center
- Iterative and incremental development
- Kanban (development)
- KISS principle
- Lean integration
- Lean software development
- Lightweight methodology
- The Magic Cauldron (essay)
- Mayo-Smith pyramid
- Micro-innovation
- Minimalism (computing)
- Open/closed principle
- Planning poker

Why this course?

- Bridge the gap between university and work experience
- Learn and try software technology in a “real world setup”
- MSc vs. Bsc:
  - Innovation (academic, technical)
  - Organization & leadership skills

Output

- Roles we are shooting at:
  - Manager
  - Architect
  - Lead Developer
  - Knowledgeable Specialist
  - ...anyone with Product Mindset

Structure of Course

- Assignment: The Project
- Lecture
  - Semester 1st part: The Process
    - Software Development Models, Methodologies and Tools
  - Semester 2nd part: The Architecture
    - Architectural Improvements, Design Patterns

Technical details: See in Canvas

- Contacts (Team-level communication)
- Requirements
- Lectures, timing
- Practice, consultation: defined by the mentor

The Project Game

- Suggested (now trending) fields
  - Make the World a Better Place!
  - Motivation
  - Innovation
  - Technical Expertise
- Future (aka Take it Seriously!)
  - Sketched Idea & Business Model
  - Sustainable Progress, Maintainable Product
  - The Team (carefully assess and select!)
  - Startup, Investors, Success

How to qualify

- Goal: MVP = Minimum Viable Product
- Teams of 5 (or 4, if needed)
- Successful project execution
  - Documented (concisely) –
    - Software Design (UX) – towards user
    - Product Architecture – towards maintainer
    - Execution Planning and Chosen Methodology
    - Development Process and Delivery Chain
  - Working and Maintainable –
    - Software Product
    - Test and Build System
    - Automated Release

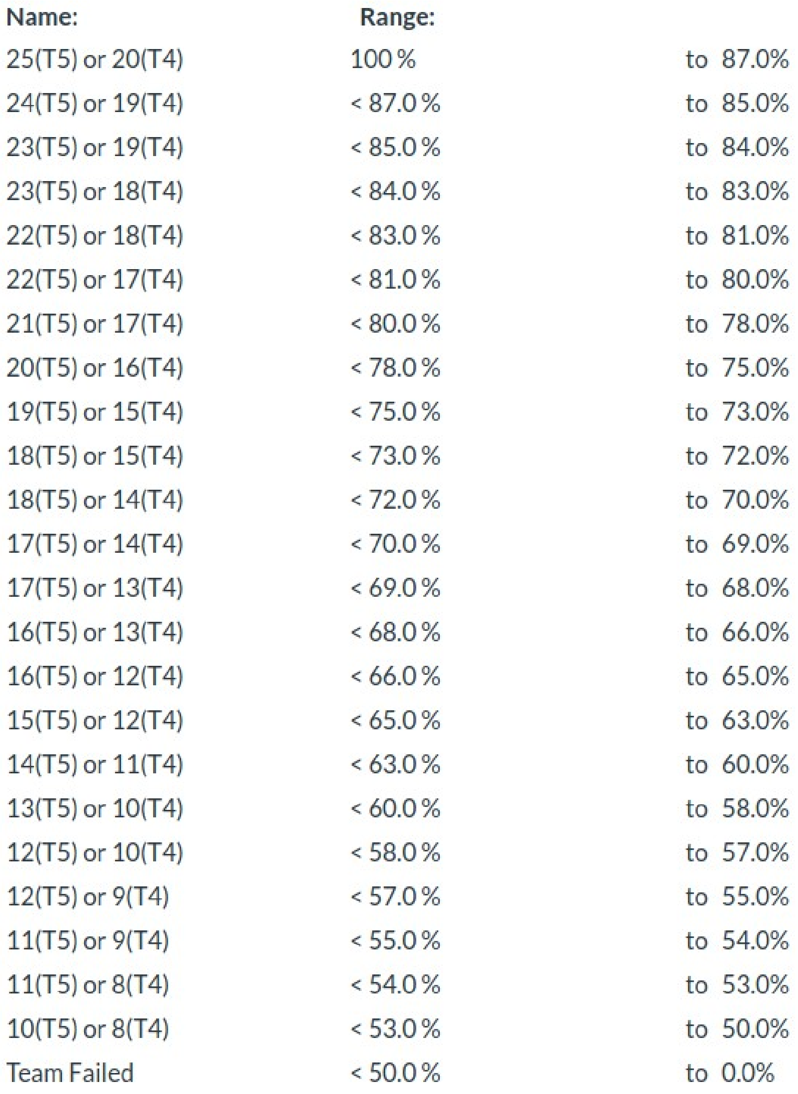
Requirements

- Single mark for all work during whole semester
- Evaluation of Teams
  - Balanced teamwork effort
- Everything is **Team Responsibility**
- Everything should be put into **VCS** (version control) (we will check individual commits)
- Participate on Weekly Consultation

Final Star Distribution

- Equal Contribution
- Fair Distribution
- Min. 1 star
- Min 50%

+ Final Test Quiz



Name:	Range:	
25(T5) or 20(T4)	100 %	to 87.0%
24(T5) or 19(T4)	< 87.0 %	to 85.0%
23(T5) or 19(T4)	< 85.0 %	to 84.0%
23(T5) or 18(T4)	< 84.0 %	to 83.0%
22(T5) or 18(T4)	< 83.0 %	to 81.0%
22(T5) or 17(T4)	< 81.0 %	to 80.0%
21(T5) or 17(T4)	< 80.0 %	to 78.0%
20(T5) or 16(T4)	< 78.0 %	to 75.0%
19(T5) or 15(T4)	< 75.0 %	to 73.0%
18(T5) or 15(T4)	< 73.0 %	to 72.0%
18(T5) or 14(T4)	< 72.0 %	to 70.0%
17(T5) or 14(T4)	< 70.0 %	to 69.0%
17(T5) or 13(T4)	< 69.0 %	to 68.0%
16(T5) or 13(T4)	< 68.0 %	to 66.0%
16(T5) or 12(T4)	< 66.0 %	to 65.0%
15(T5) or 12(T4)	< 65.0 %	to 63.0%
14(T5) or 11(T4)	< 63.0 %	to 60.0%
13(T5) or 10(T4)	< 60.0 %	to 58.0%
12(T5) or 10(T4)	< 58.0 %	to 57.0%
12(T5) or 9(T4)	< 57.0 %	to 55.0%
11(T5) or 9(T4)	< 55.0 %	to 54.0%
11(T5) or 8(T4)	< 54.0 %	to 53.0%
10(T5) or 8(T4)	< 53.0 %	to 50.0%
Team Failed	< 50.0 %	to 0.0%

Deadlines – see Canvas

WARNING

DEADLINES ARE MUCH
CLOSER THAN THEY APPEAR

Team Facilitation

1) Startup Idea Pool by **next week!!! (w2)**
2) Mentors select best ideas (top 20%)
3) Mentors are assigned to ideas
4) Idea owners put together Team
    1) Set Canvas Profile Picture!!!
    2) Use Discussions (Forum) to find members or teams!
5) Consultation Starts with Mentor, Idea Elaborated
6) Pitch

Be Open!

- Share Your Idea, Get Feedback
- Make Profile Picture and CV on your technical skills for Idea Owners!
- What Makes a Good Team
  - Members Selected for Idea by – **GOOD**
    - Motivation
    - Technical Skills
  - Idea Selected for Members by – **EXTREMELY DANGEROUS!!!**
    - Friendship
    - Speaking the same language
    - Drinking Buddies
- Work Openly https://opensource.guide/