



Master HPC/QC  
HPC-04 Software Engineering

# Overview

Prof. Dr. Christoph Schober, 3/20/2024


# Lectures


- 1 | About Me: Short introduction and background
- 2 | Introduction To Software Engineering:
- 3 | Requirements Engineering:
- 4 | Operating Systems: Introduction to operating systems for computers
- 5 | Introduction To Linux: An in-depth introduction to Linux as operating system
- 6 | Version Control Systems: Introduction to version control systems for source code
- 7 | Project Management: Introduction to software project management
- 8 | Agile development with Jira: How Jira can be used to do agile projects
- 9 | Agile Estimation: Why and how project tasks are estimated in agile software development
- 10 | System Modeling: System Modeling using UML
- 11 | Software Testing: Unit-, integration- and end2end testing in software
- 12 | Design Patterns: How to 'Not Reinvent The Wheel'
- 13 | Containerization: Introduction to containerization using Docker
- 14 | DevOps and CICD: Using automated processes and tools in software engineering

# Why Software Engineering for HPC/QC?

Both HPC and QC are very dynamic fields often close to cutting-edge research and complex methodology (as opposed to, let's say, business application development) and done by a variety of different professions (engineers, physicists, mathematicians, chemists, computer scientists, ...)

- Large-scale HPC (e.g., supercomputer simulations) done by scientists, often as part of a PhD
- HPC in engineering and manufacturing (e.g., finite element simulations) done as part of product research and development
- Quantum computing due to the lack of standardized tools, methodology and hardware

- How to design a simulation code for best (parallel) performance?
-  How to maintain a complex code developed by multiple contributors?
- How to handle large legacy codes?

 Good understanding of software engineering (and best practices) help to get better results faster with less pain!

# Competences

 You should have acquired the following competencies:

- Work with Linux locally and remote
- Use Git for version control
- Use Gitlab as collaborative platform
- Approach new projects systematically using established methodology
- Know about different ways to test code
- Apply software design patterns when applicable
- Understand and apply containerization with Docker and Docker Compose
- Work with Gitlab CI/CD to create automated DevOps pipelines

# Lecture Structure

- Lectures with small in-class exercises and material for self-study
- Exercise content re-useable for HPC-05 Programming Lab!
- Initial lectures offered as hybrid lectures (in-person + video call)
- Later lectures only in person in Deggendorf
- Examination type: **Written Exam 90 minutes** (at the end of the term)

# Outline

Week	Content
1	Introduction, Linux Basics
2	<b>Holiday</b> , Linux
3	Version Control
4	Requirements Engineering
5	Project Management
6	Gitlab
7	<b>Holiday</b>
8	System Modeling
9	Testing

Week	Content
10	<b>Holiday</b>
11	Writing Testable Code
12	Containerization
12	Containerization
14	DevOps and CI/CD
15	DevOps and CI/CD
16	Open Topics
17	Exam Review

