CS 3300-A Introduction to Software Engineering

# Lecture 01: Introduction & Overview

Nimisha Roy ▸ nroy9@gatech.edu



These slides are adapted from Alessandro Orso and Mahdi Roozbahani

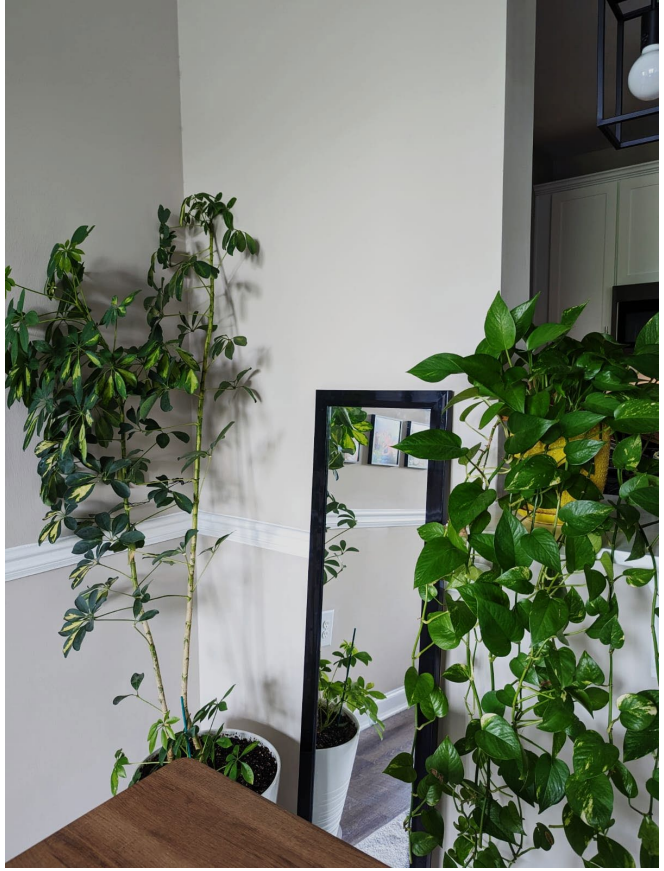
# Contents

- Course Overview
  - Introduction and Big Picture
- Course Logistics
  - Schedule,Class Organization, Projects, Team Formation, Policies

# Your Instruction Team

**Instructor: Dr. Nimisha Roy**

- 3rd year teaching this course
- Lecturer in the College of Computing (SCI/OMSA/OMSCS)
- PhD in Computational Science & Engineering from Georgia Tech



Research Interests: Computing Education, Intelligent infrastructure, Software Development

Hobbies: singing (jamming), interior designing, admiring my plants

# TA Team

Schezeen Fazulbhoy
Head TA

Jamal Faqeeri
TA

# Course Overview

# What is Software Engineering?
# What do software engineers do?

- Software Engineering – The discipline allows systematic application of methods to build and manage high-quality programs/softwares.

- Software engineers are engineering or computer science professionals who combine engineering principles and programming.
  - End to end software life cycle planning
  - Maintaining Disciplined Product control
  - Use modern programming practices
  - Perform Continuous Validation and Improvement of the process

# Industry Roles

- **Front-End Engineer:** Creating user interfaces and focus on the visual elements, fixing bugs and user experience (UX).
- **Back-End Engineer:** Focus on the core logic and performance of the application while taking scalability and integration into account.
- **Full Stack Engineer**: Handle both front-end and back-end, which means that they create a full application on their own.
- **QA Engineer:** Writes the software that tests for quality. QA ensures the product or processes run as expected.
- **DevOps Engineer:** Build and maintain back-end software and distributed systems, such as database servers and application infrastructure.
- **Security Engineer:** Use their ethical hacking skills to test the security of software systems for vulnerabilities that need to be fixed.

# What is Software Engineering?
# Why is it Important?

What is this?

[ ] 4th of July fireworks

[ ] Flare gun in action

[✓] Explosion of Ariane 5 rocket due to Software errors

https://www.bugsnag.com/blog/bug-day-ariane-5-disaster

# What is Software Engineering? Why is it Important?

Why is it so difficult to build good software??

Topic of this course and why Software Engineering is an important course in Computer Science

# What are attributes of a good software?

- Functionality
  - The software should deliver the required functionality and performance to the user
- Usability
  - Should be easy to use. Not unnecessarily complex
- Maintainability
  - Software must be evolvable to meet changing needs
- Dependability
  - Software must be trustworthy (reliability, security, and safety)
- Efficiency
  - Time & Cost effective

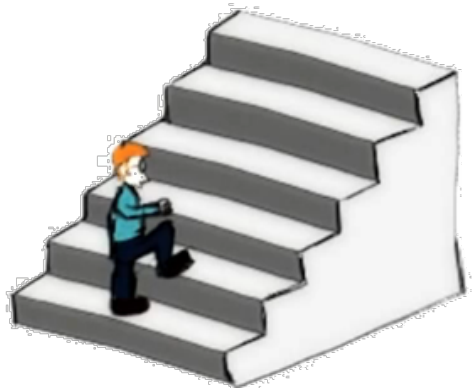
# Some questions

- What is the largest software system on which you have worked?
- How many LOCs/day were you producing?
- How many LOCs/day professional software engineers produce?

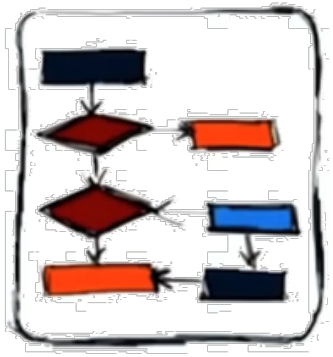
  < 25?  25-50?  50-100?  100-1000?  > 1000?

- But what are they doing with the rest of their time?
- How do large systems get built?
- What process should be followed?
  - No one size fits all
  - We will see several

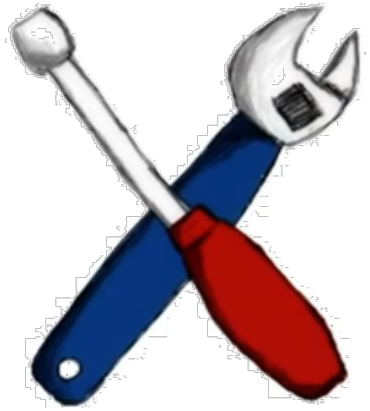
# Software Engineering



Methodologies



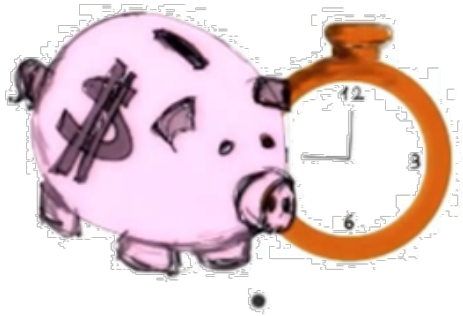
Techniques



Tools

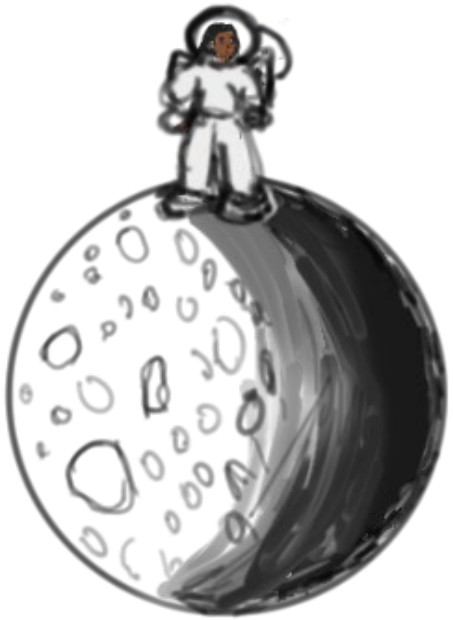

To build software of high quality

That works



And fits into Budget

# The 60s



Man on Moon

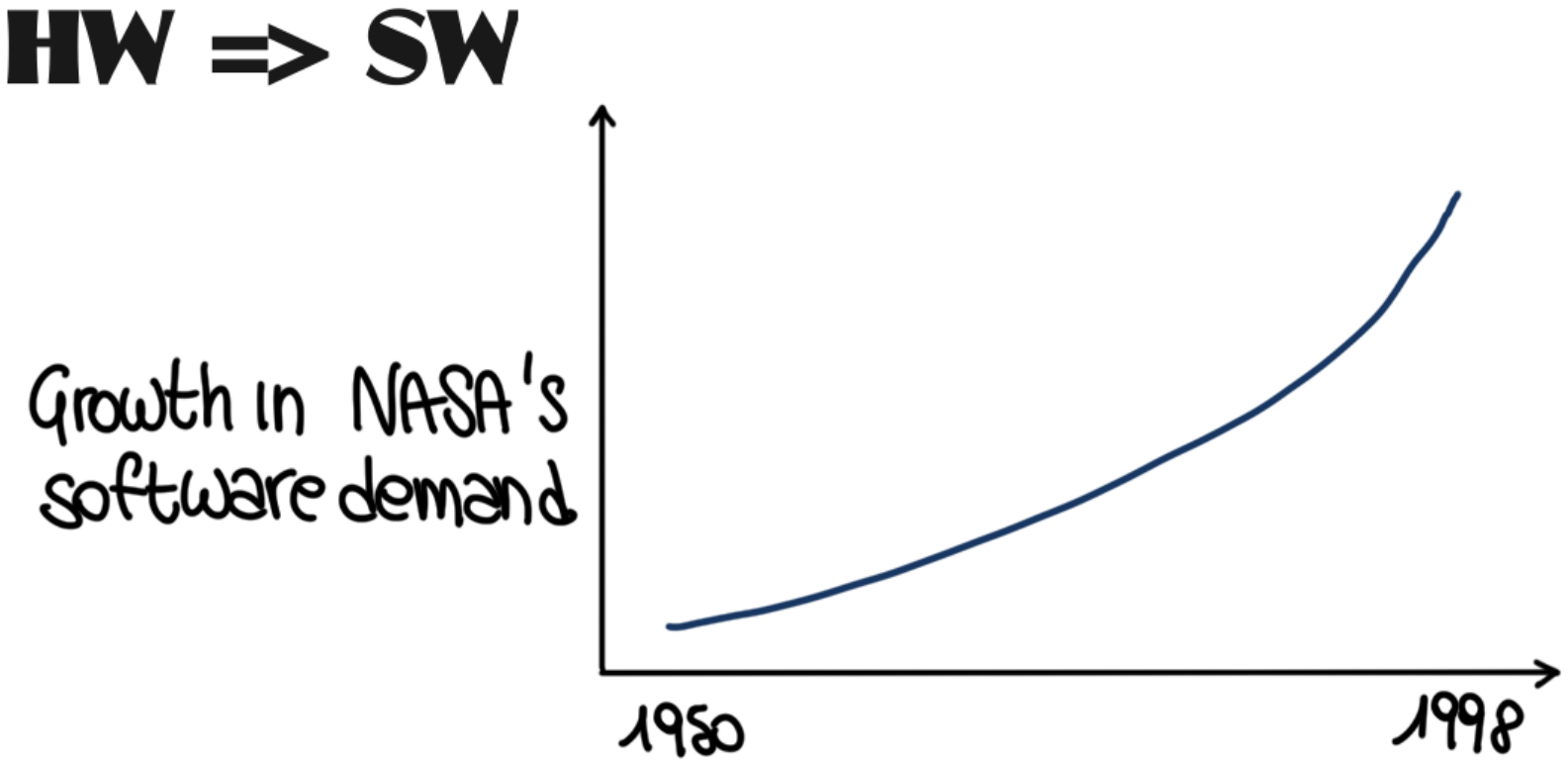
Polaroid

# History


https://en.wikipedia.org/wiki/Software_crisis

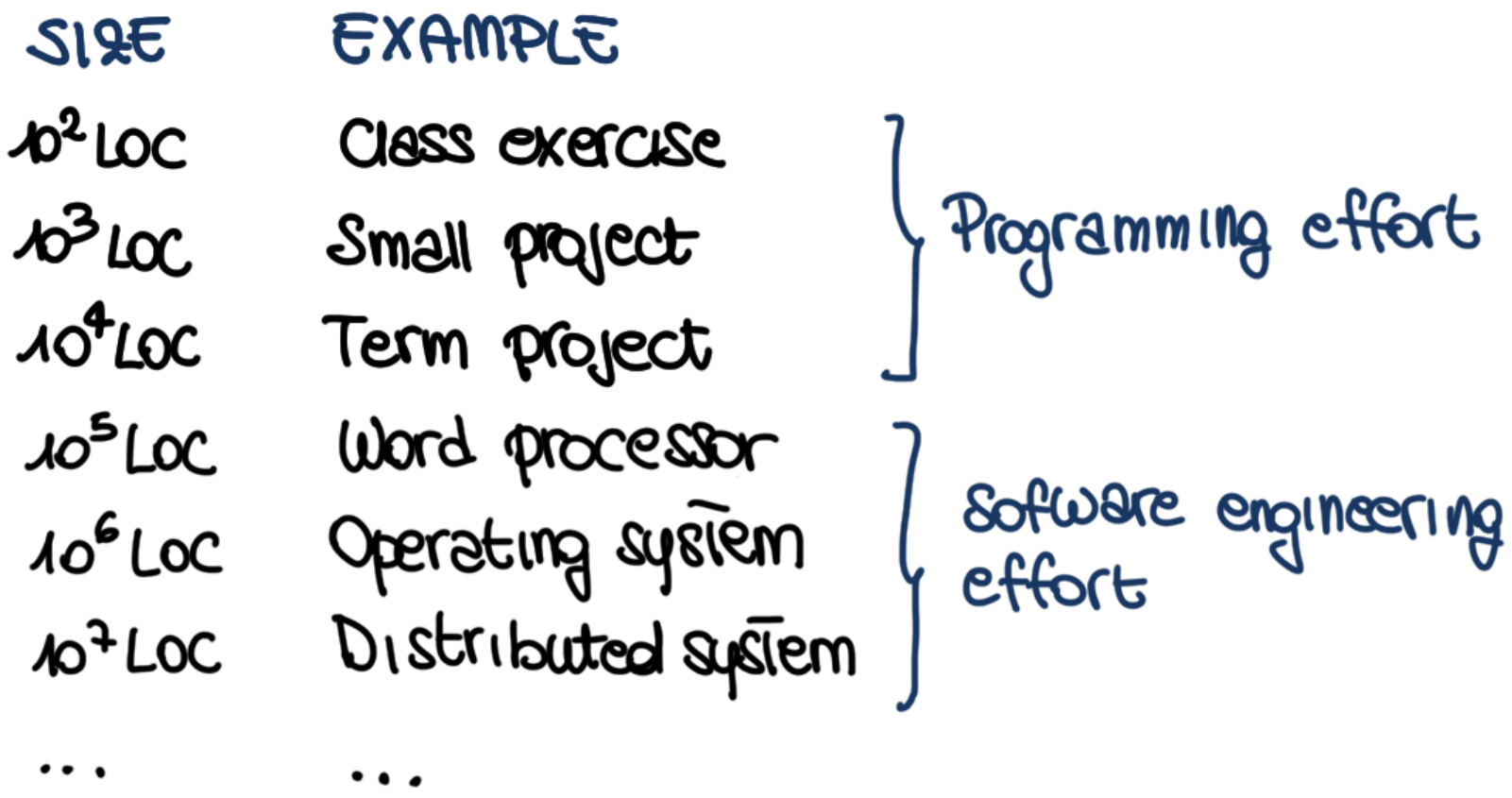
# History

## Reason 1: Rising Demand for Software



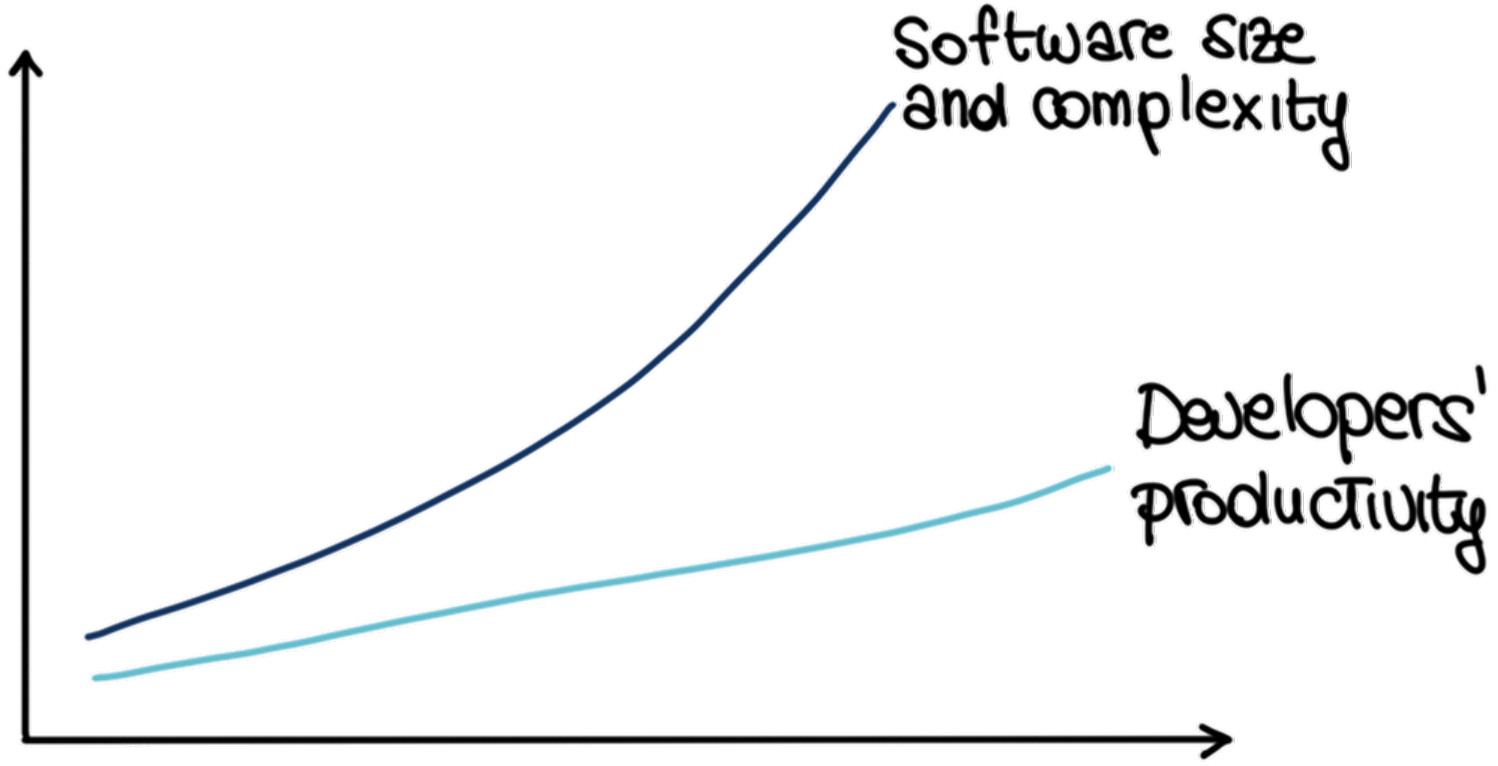
# History

## Reason 2: Increasing product complexity



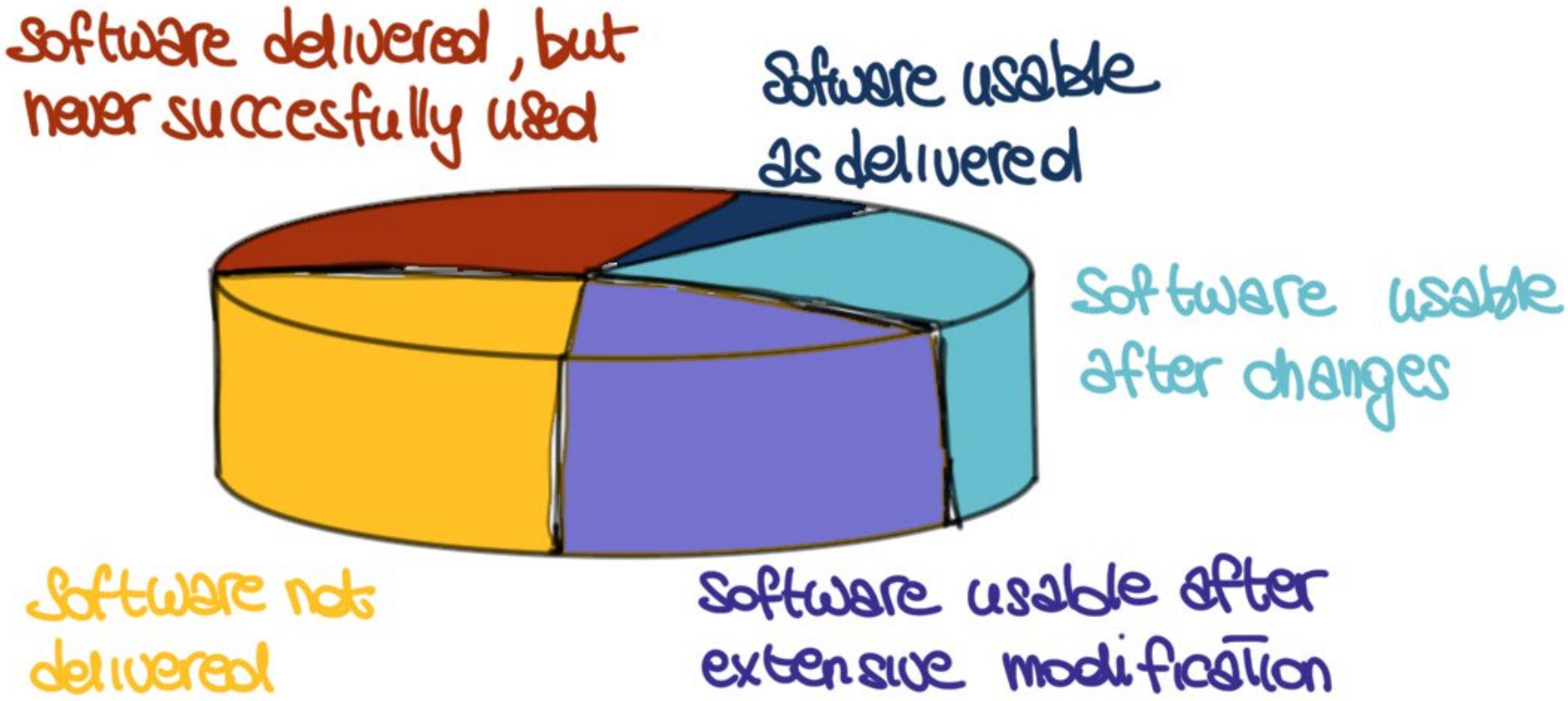
# History

## Reason 3: Developers Productivity Growth



# History

## Study of 9 Development Contracts (Davis, 1990)

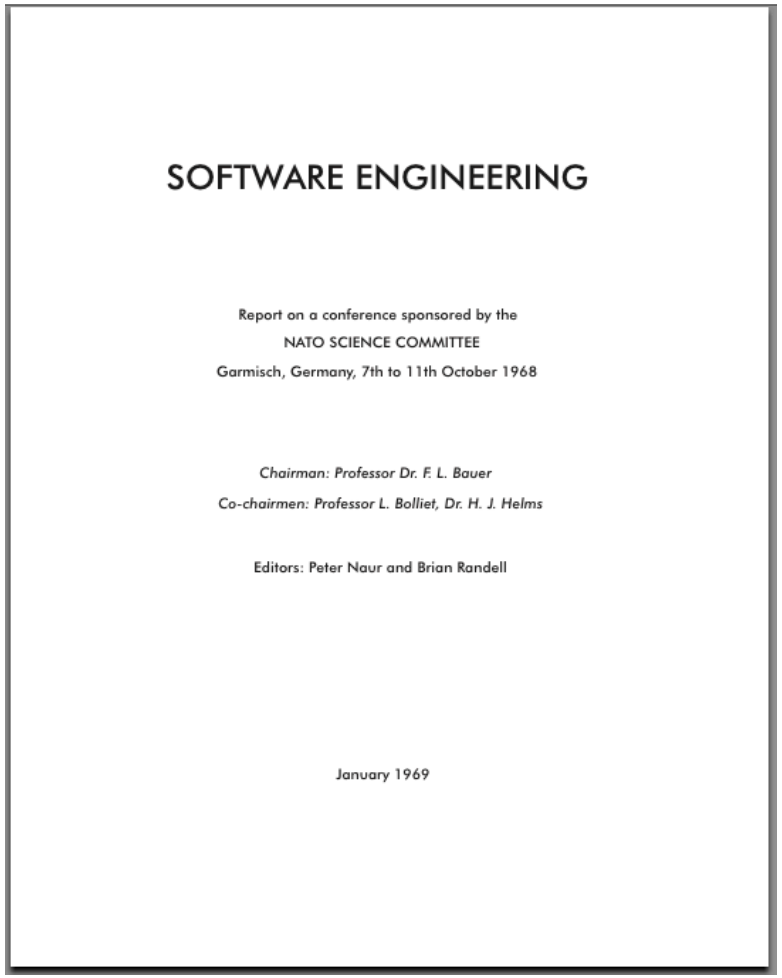


# History

## Birth of Software Engineering

# History

## Birth of Software Engineering



SOFTWARE ENGINEERING

Report on a conference sponsored by the
NATO SCIENCE COMMITTEE
Garmisch, Germany, 7th to 11th October 1968

*Chairman: Professor Dr. F. L. Bauer*
*Co-chairmen: Professor L. Bolliet, Dr. H. J. Helms*

Editors: Peter Naur and Brian Randell

January 1969

Link to proceedings:
http://homepages.cs.ncl.ac.uk/brian.randell/NATO/

# What are common causes of SW failures?

- No standard procedures for development
- Inadequate understanding of requirements
- Sheer complexity of software (e.g., concurrency, distribution)
- Size of project (too large for a single manager)
- Difficult to match technical knowledge of staff with project needs
- Poor design/implementation/testing methodology
- Requirements change during project
- Poor documentation
- Force fitting software components to applications
- Changing/reusing code without understanding it
- Poor management: lack of communication, poor cost/schedule estimates
- Unrealistic expectations
- Lack of measurement
- Lack of teamwork
- Performance differences among staff
- …

# Software importance today

- More and more systems are software controlled
- The economies of ALL developed nations are dependent on software
- Expenditure on software represents a significant fraction of GNP in all developed countries

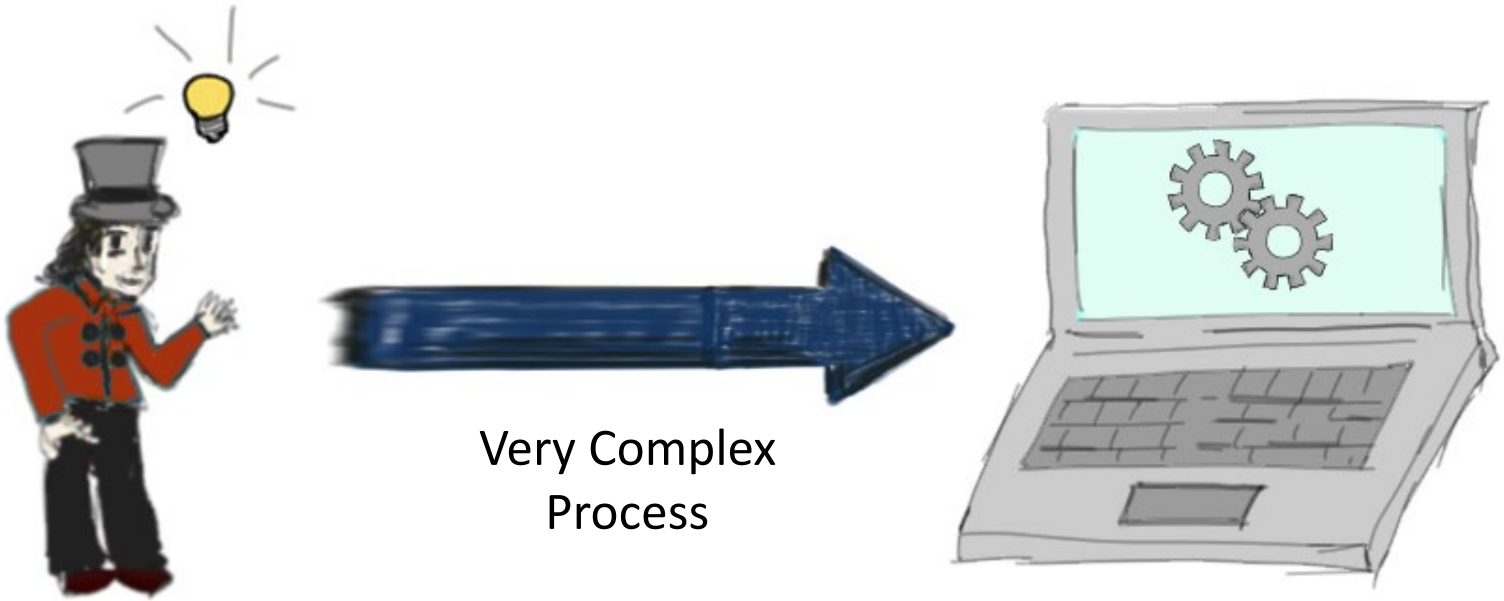
# What are the key challenges facing SE?

- How can we build high-quality systems?
- How can we do it in a reasonable time?
- How can we do it at a reasonable cost?

# Software Process, Phases, Tools of the Trade

Abstract Idea

System Implementation



Very Complex Process

Software Process

- Systematic
- Formal

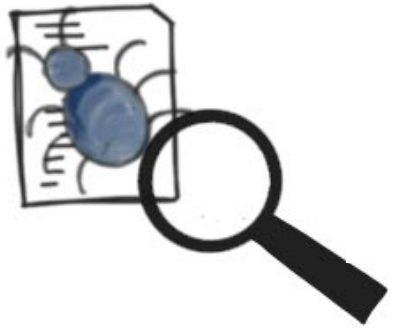
# Software Process, Phases, Tools of the Trade

## Software Phases

Requirements Engineering

Design

Implementation



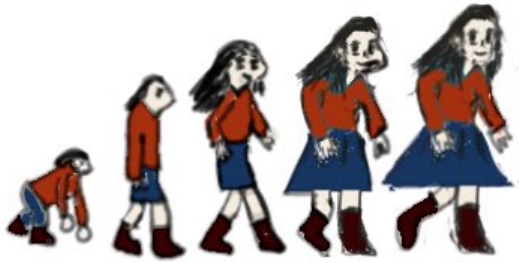
Verification & Validation

Maintenance

# Software Process, Phases, Tools of the Trade

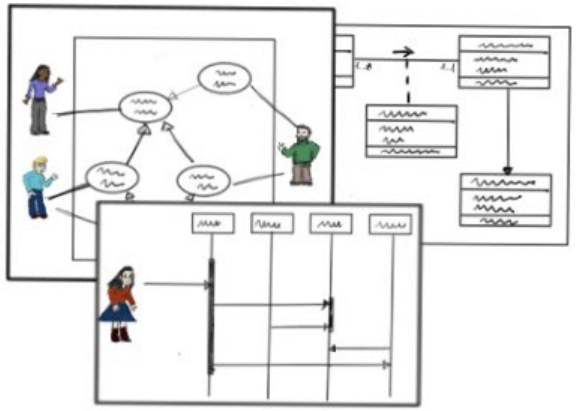
## Software Process

Waterfall



Evolutionary Prototyping

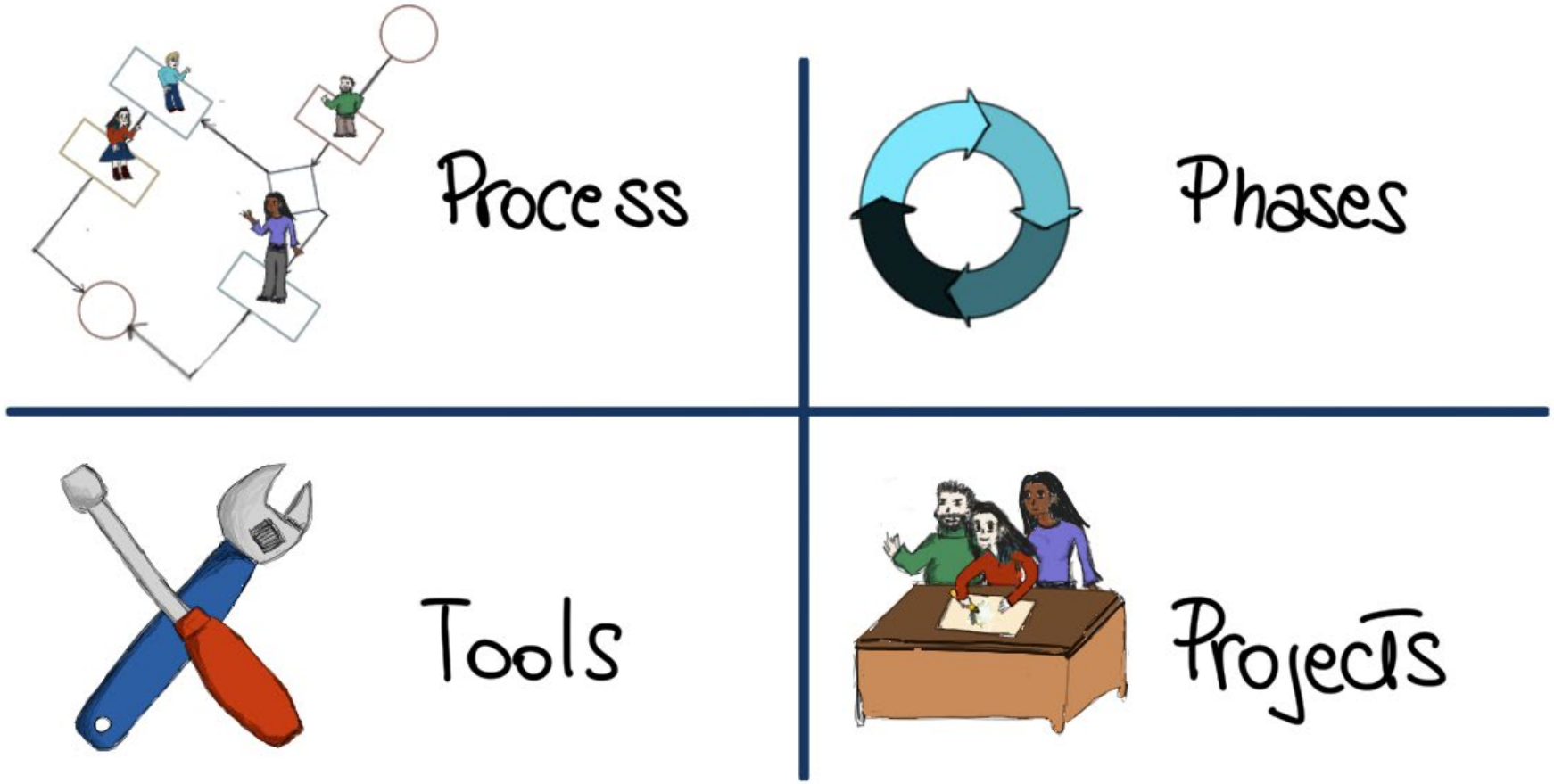
RUP/USP



Agile

# Course Logistics

# Course Overview



# Class organization

- Class Website and Canvas
- Ed Discussion
- In-class Lectures
- Attendance is required
- Tools of the Trade Demonstrations – Bring Laptops to class.
- Invited lectures
- Team-work

Refer to:

## Canvas

For anything (syllabus, lectures, helpful links, logistics etc.) related to this class

## Ed Discussion

For any communication with the instruction team. No Emails please!

Introduce yourselves on Ed ☺

# Grading Distribution

- Assignments (Individual) 20%
- Project 1 (group) 30%
- Project 2 (group) 35%
- Team Management (Individual) 5%
- CATME Peer Review (Individual) 8%
- Quizzes (Individual) 8%
- Survey Submissions (Individual) 2%

# Tools of the Trade Lectures

Demonstration in Lectures – Bring your Laptops

- Version Control Systems
- IDEs
- Front-End Development
- Back-End Development
- Google Cloud Platform

# Information about projects

- Two projects
- **Project 1:** WEB-APP deployed on Google Cloud Platform. All teams will do the same project
- **Project 2:** You will choose the project. It can be web- or mobile-app.
- Team based
  - Different teams
  - Grades will be adjusted based on peer review
- Tools/environments
  - Project 1 – Springboot, Maven, GCP
  - Project 2- GCP but tech stack upto you.

# Team Formation

- Team Formation Survey:
  - To be completed by ***everyone***
  - **Released today; Due: 8/29 at 11:59 PM**
- Team Preferences:
  - Groups of 2+ students: Sign up using Canvas Groups
    - Use the ***existing*** available group
    - ***Do NOT create your own group***
  - No preferences:
    - Will be matched via CATME

**Teams will be assigned by 8/31.**

**1st group assignment out 8/31**

# Summary

# Summary

- SE important/critical discipline
  - Concerned with cost-effective software development (all aspects!)
  - Based on a systematic approach that uses appropriate tools and techniques, follows a process, and operates under specific development constraints

- Goal of SE is to deliver high-quality products that provide the expected functionality, meet projected time estimates, and have a reasonable cost