

SOFT620020.02

Advanced Software Engineering

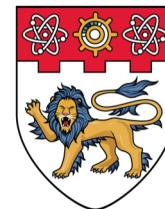
Bihuan Chen, Pre-Tenure Assoc. Prof.

bhchen@fudan.edu.cn

<https://chenbihuan.github.io>

About Me

- BSc and PhD from Fudan
- Postdoc from NTU
- Pre-Tenure Assoc. Prof. at Fudan



NANYANG
TECHNOLOGICAL
UNIVERSITY
SINGAPORE

- No teaching experience 🤦

- Software Engineering
- Program Analysis
- Software Testing
- Software Security



Course Structure

- In-Class Teaching and **Discussion** (Every Monday 11-13)
 - No After-Class Assignments
-
- Discussion (30%): a range of topics
 - Survey or New Ideas Presentation (30%): 13 + 2 minutes
 - Survey or New Ideas Paper (40%): Jan. 11, 2019

Course Communication

- All materials are available at the course website
<https://chenbihuan.github.io/course/ase/>
- Email: bhchen@fudan.edu.cn
- Office: Room 403, Software Building, Zhangjiang Campus
(make an appointment via email first)
- WeChat group for course announcement

QR Code of WeChat Group



Valid until 9/17 and will update upon joining
group

A Quick Journey through Software Engineering

A Warm-Up Questionnaire



- Did you take the course about software engineering?
- Did you have programming experience of \geq two years?

- Did you participate in the development of a project?
- Did you write and manage tests for your projects?
- Did you have a debugging nightmare for your projects?
- Did you use any automatic tools to help you test or debug?

- Did you submit bug reports for open-source projects?
- Can you prove that your program is correct?

Discussion 1 – Software and SE



- What is software?
 - What are the attributes of good software?
 - What is software engineering?
-
- Software: computer **programs** and associated **documents**
 - Good software: deliver the required **functionalities** and **performance** to users, be **maintainable**, **dependable** and **usable**
 - Software engineering: an engineering **discipline** that is concerned with **all** aspects of software production

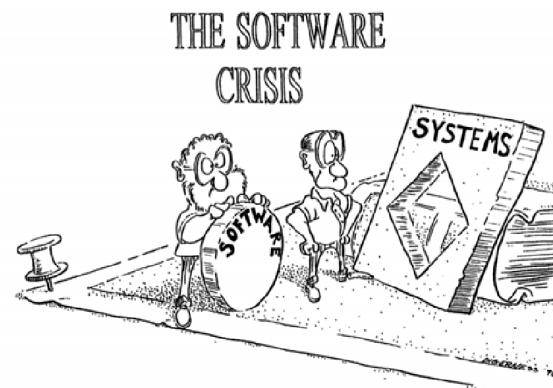
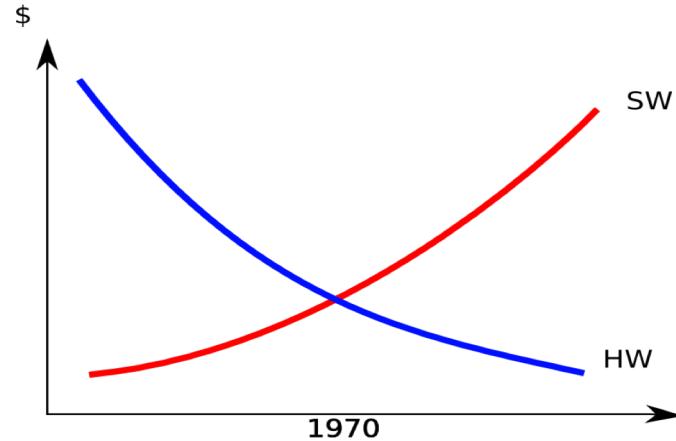
Discussion 2 – Challenges and Costs



- What are the key challenges facing software engineering?
- What are the main costs of software engineering?
- Challenges: coping with increasing **diversity**, demands for **reduced delivery times** and developing **trustworthy** software
- Costs: 60% of costs are **development costs**, 40% are **testing costs**, and **evolution costs** often exceed development costs

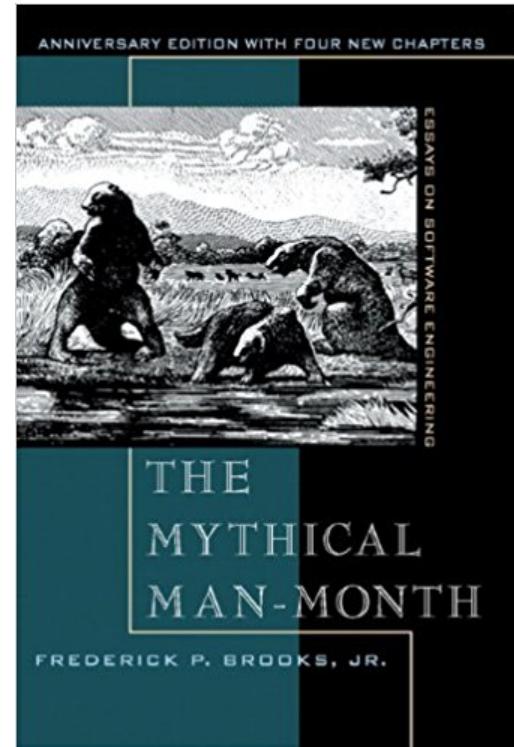
Software Crisis

- Projects ran over-budget
- Projects ran over-time
- Software was very inefficient
- Software was of low quality
- Software often did not meet requirements
- Projects were unmanageable
- Code was difficult to maintain
- Software was never delivered
- ...



IBM's System/360 Operating System

- Development period: 1963 – 1966
- Manpower: 5,000 man-year
- Code: one million lines of code
- Costs: hundreds of millions of dollars
- Delivery was delayed
- Cost was several orders of magnitude higher
- More memory was needed as planned
- The first release did not work well
- The system worked well after several releases
- Each release fixed thousands of bugs



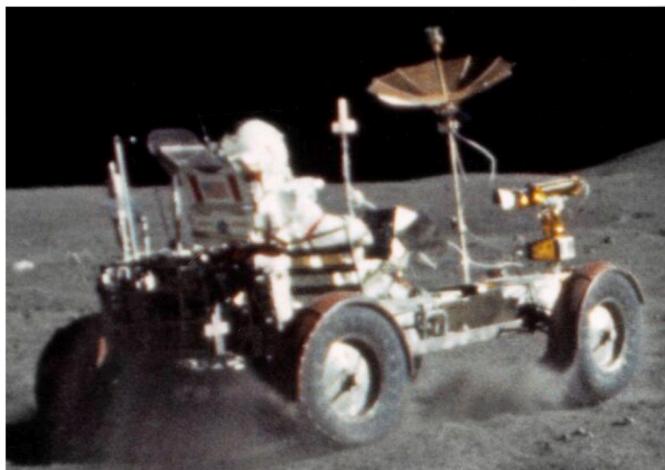
[Frederick Brooks](#)
[“Adding manpower to a late software project makes it later”](#)

The Idea of Software Engineering

<https://www.youtube.com/watch?v=ZbVOF0Uk5IU>

- Margaret Hamilton, the lead Apollo flight software designer

"When I first came up with the term, no one had heard of it before, at least in our world. It was an ongoing joke for a long time. They liked to kid me about my radical ideas."



The Apollo 11 on the moon

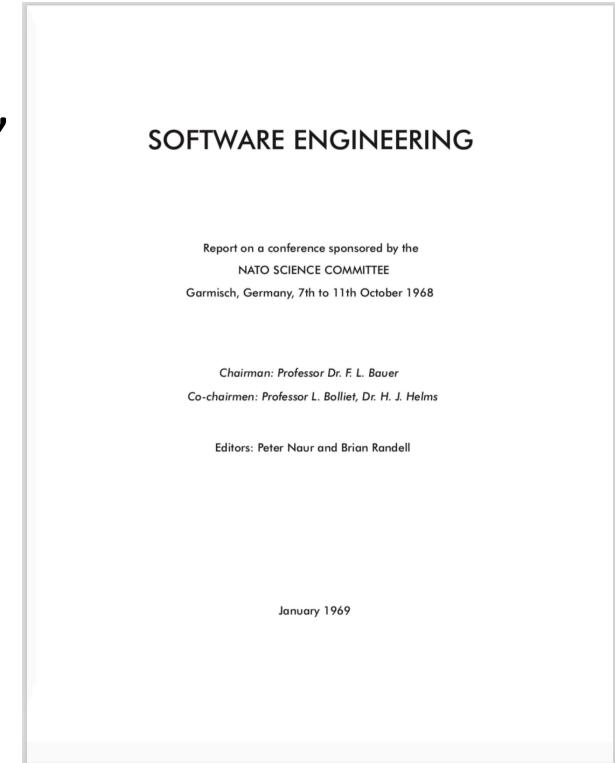


Standing next to listings of the software

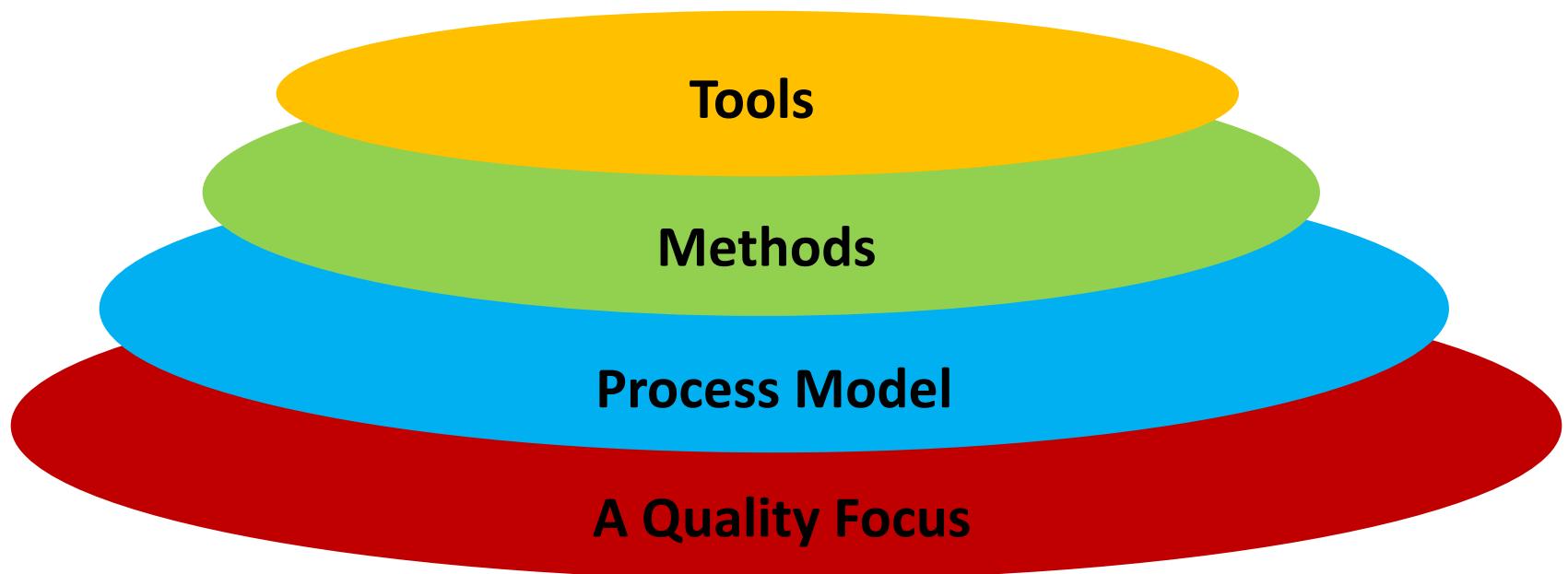
Her concepts (e.g., asynchronous software, priority scheduling and software reuse) became the foundation for ultra-reliable software design

The Birth of Software Engineering

- **Fritz Bauer**, at NATO in 1968
- Suggest the term “Software Engineering” as a way to conceive both the problem and the solution of software crisis
- *“The establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines”*



SE: A Layered Technology



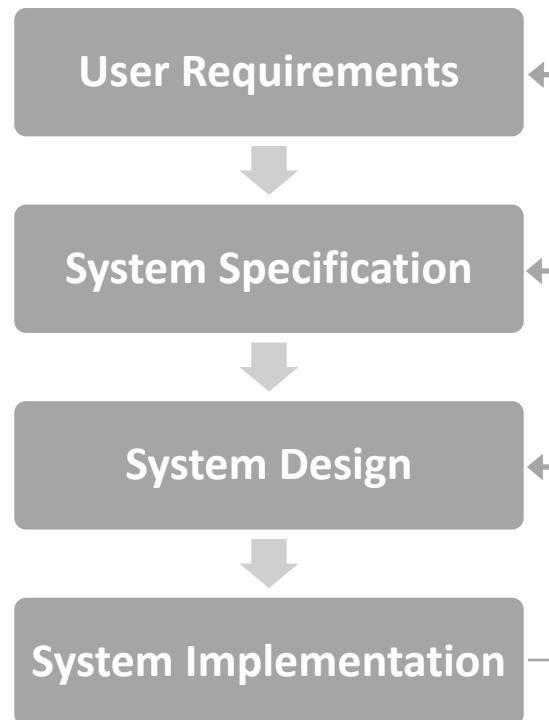
Activities in Software Engineering

Do we get the right requirements?

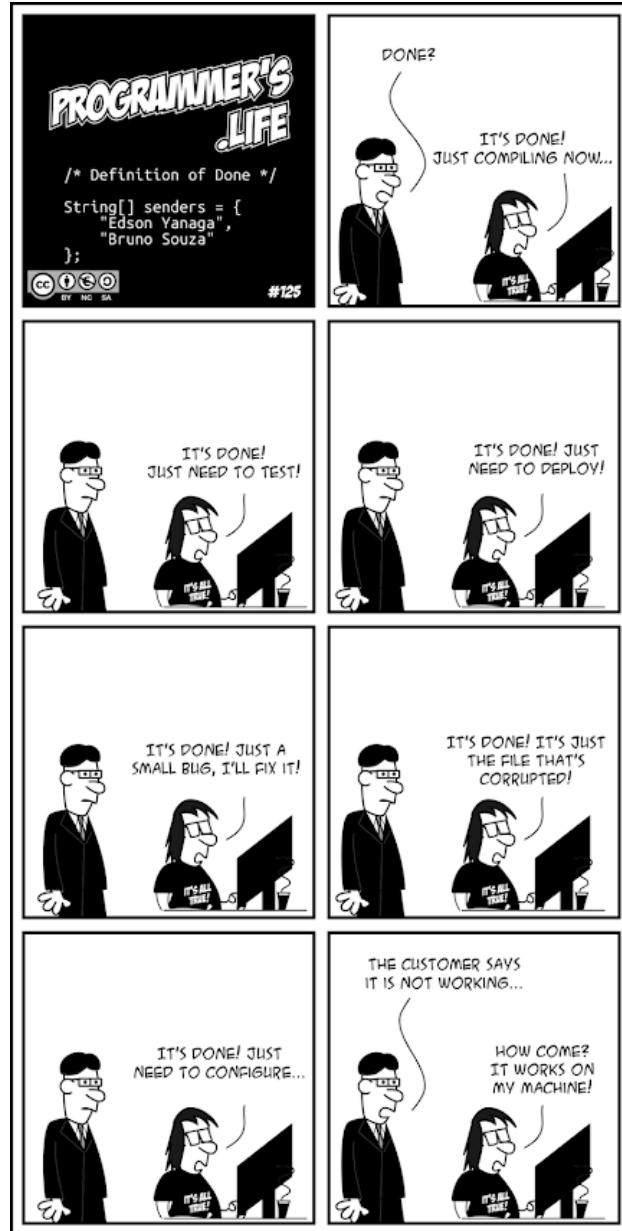
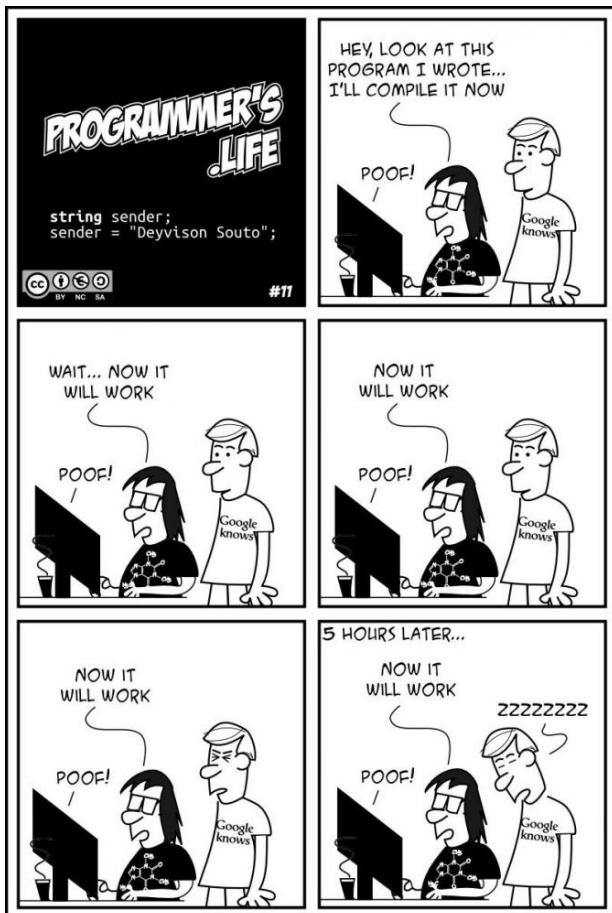
Is the specification equivalent to requirements?

Does the design satisfy the specification?

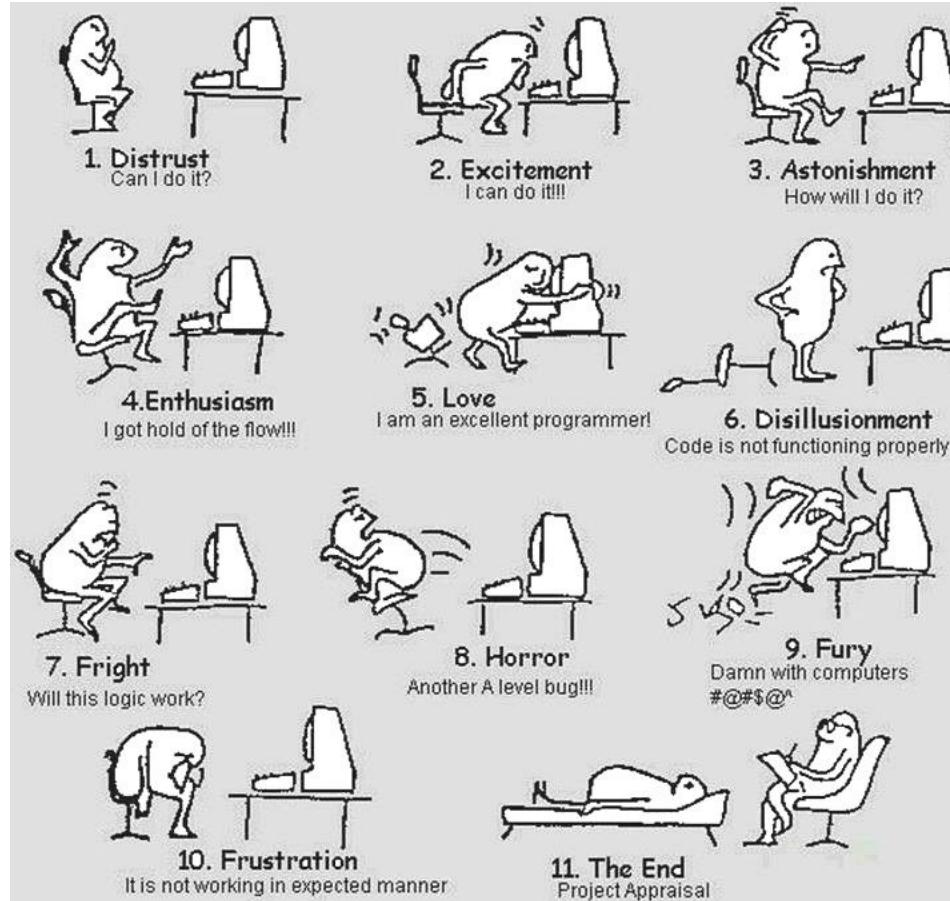
Is the design correctly implemented?



A Programmer's Life

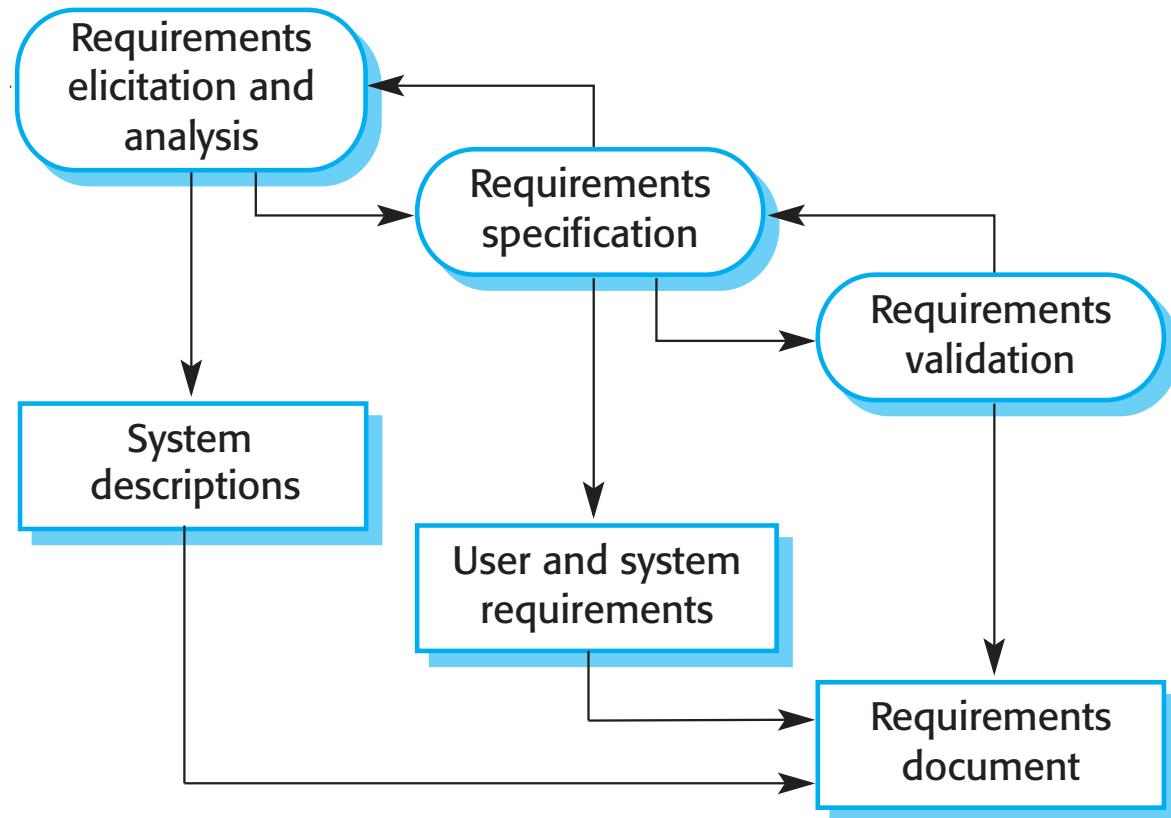


A Programmer's Life (cont.)



Requirements and Specification

- Establish the required services and the constraints on the system's operation and development

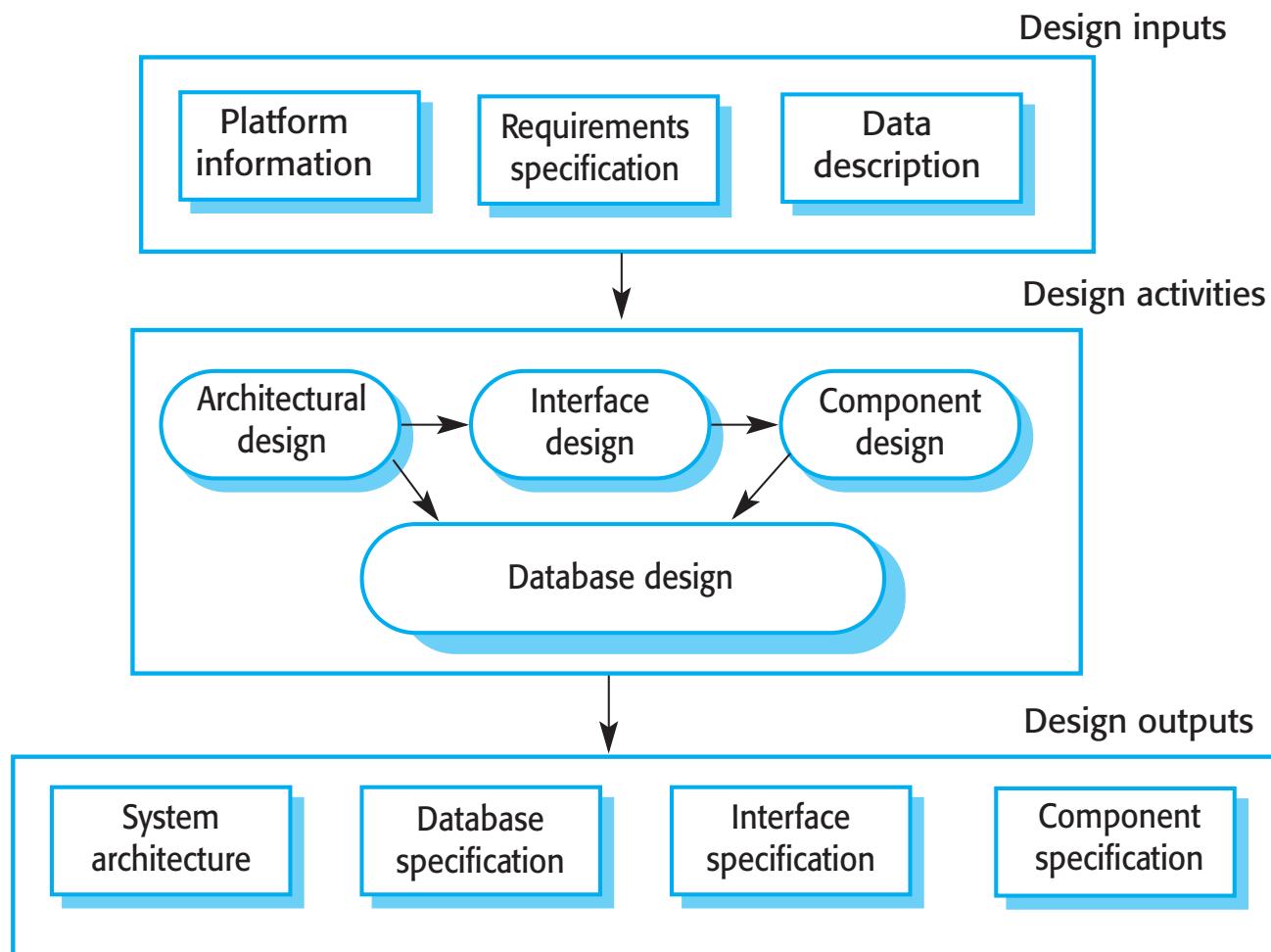


Requirements and Specification (cont.)

- Example: sort an array of integers within certain time
 - Red: functional requirement
 - Green: quality requirement
- Formal specification languages
 - The Z language, VDM, the B language, etc.
 - CSP, CCS, etc.

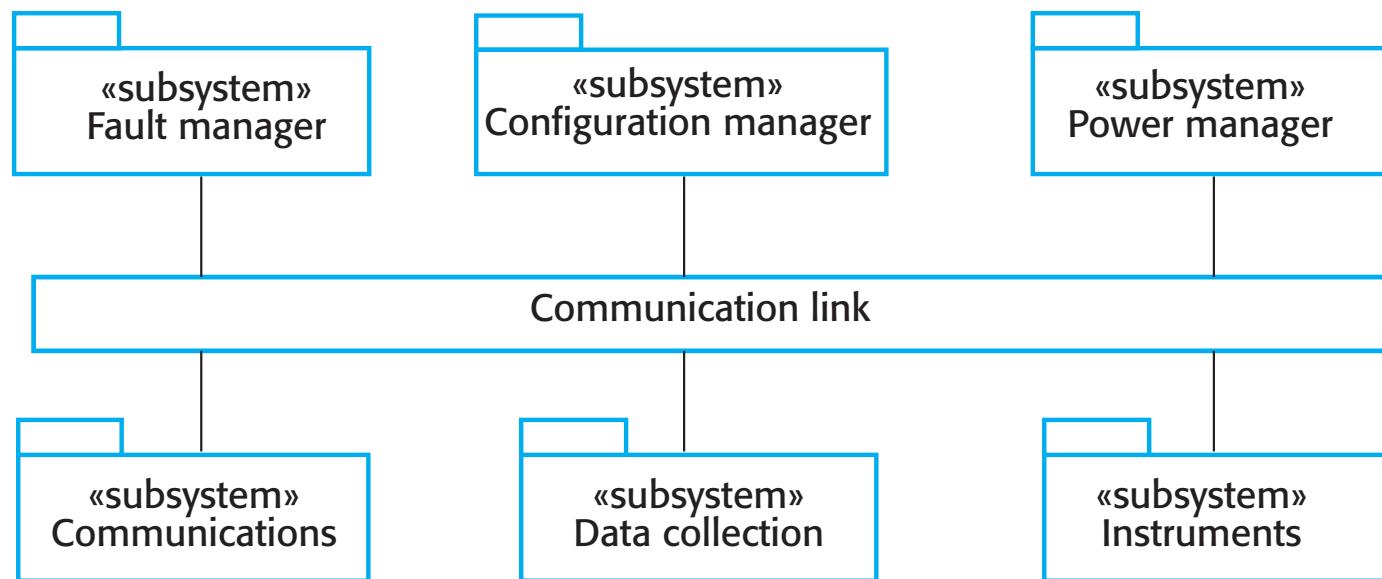
System Design

- **Design a software structure that realizes the specification**



System Design (cont.)

- Example: high-level architecture of a weather station system

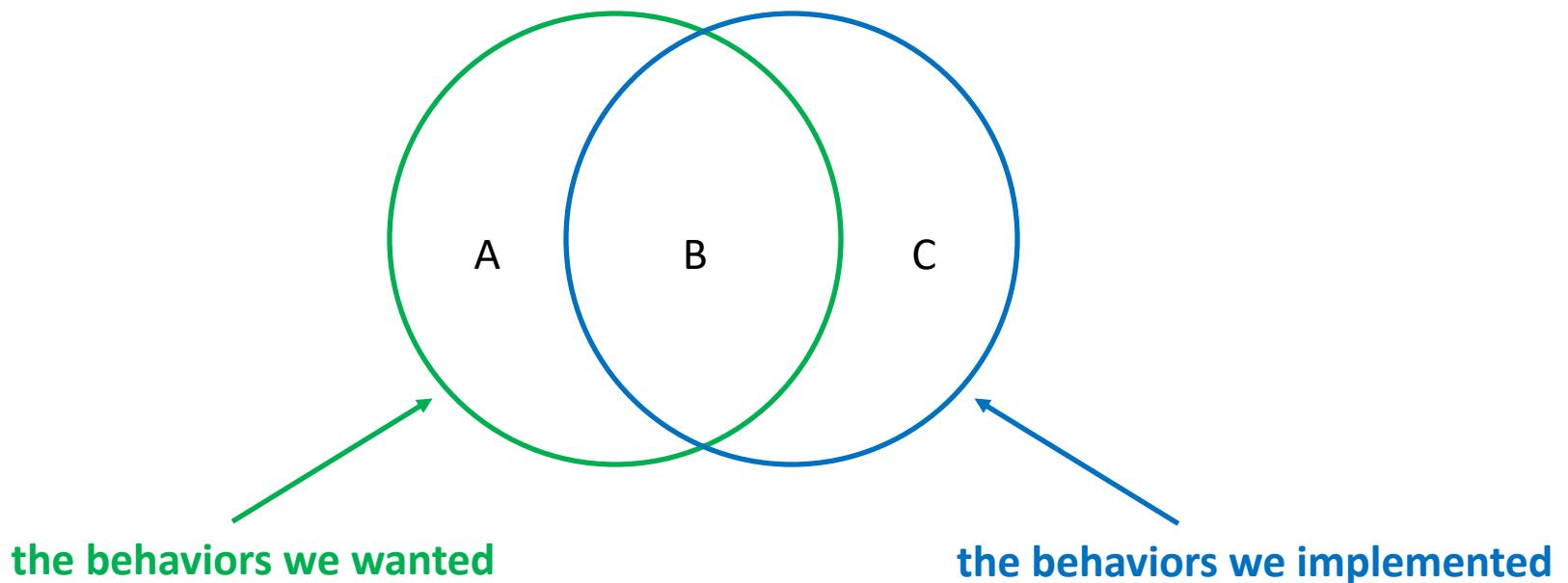


System Implementation

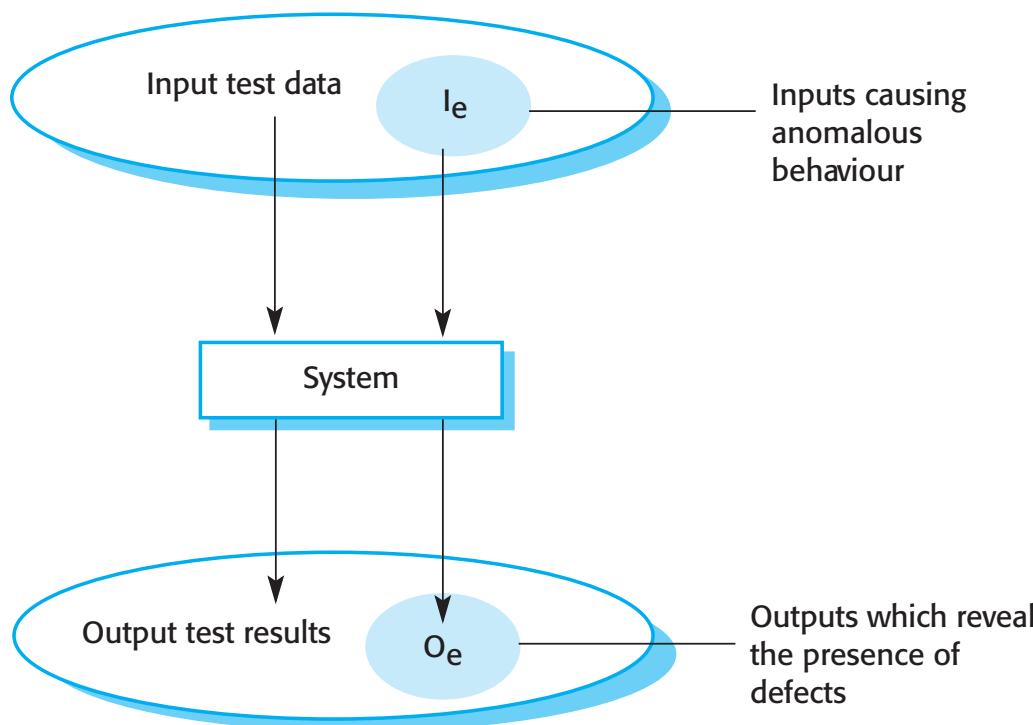
- Translate the software structure into an executable program
- The focus here is not on programming but on other implementation issues
 - Reuse
 - Configuration management
 - Host-target development
 - ...
- The activities of implementation and testing are often closely related and may be interleaved

Testing

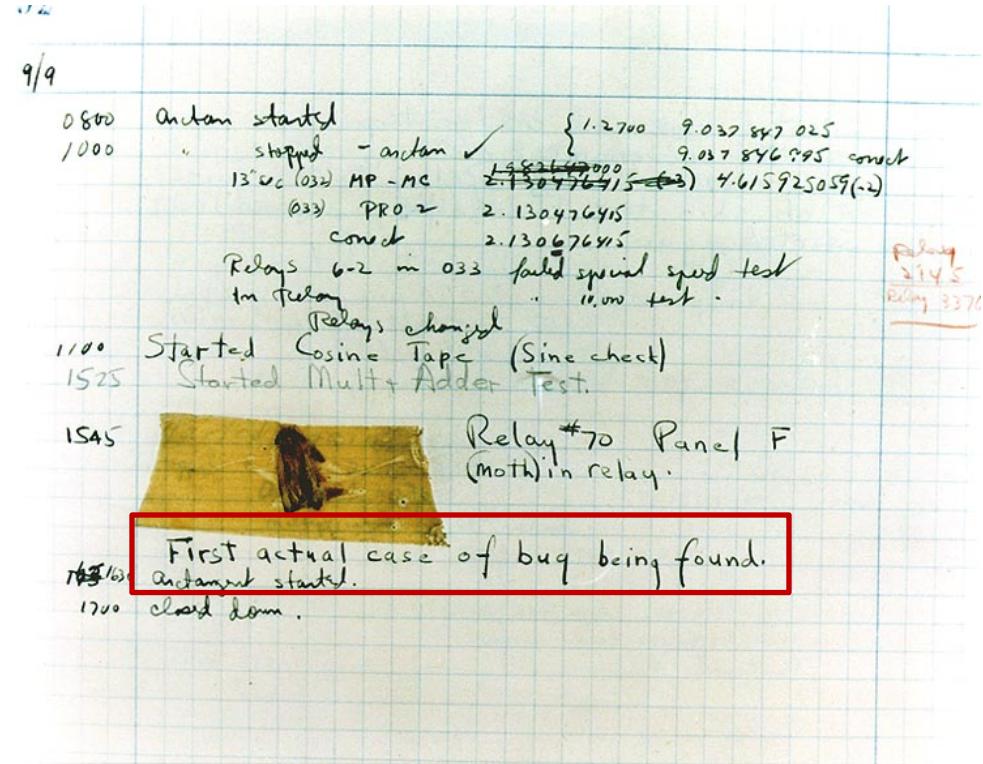
- Testing is the most commonly used verification and validation activity



Testing (cont.)



The Origins of Bug



Grace M. Hopper's associates discovered a **moth** that was stuck in a relay; and the moth impeded the operation of the relay

Discussion 3 – Bug



- unexpected program behaviours manifested as crashes, errors, loss of data, denial of service, poor performance, high energy consumption, etc.

What kinds of bugs did you encounter?

The **red** and **green** library contains a class with same name but different function, and I needed the class in the **red** library



Computer A

Computer B

Root cause: use **File.listFiles** to load the two libraries, which has no guarantee of any specific order.

Discussion 4 – Debugging

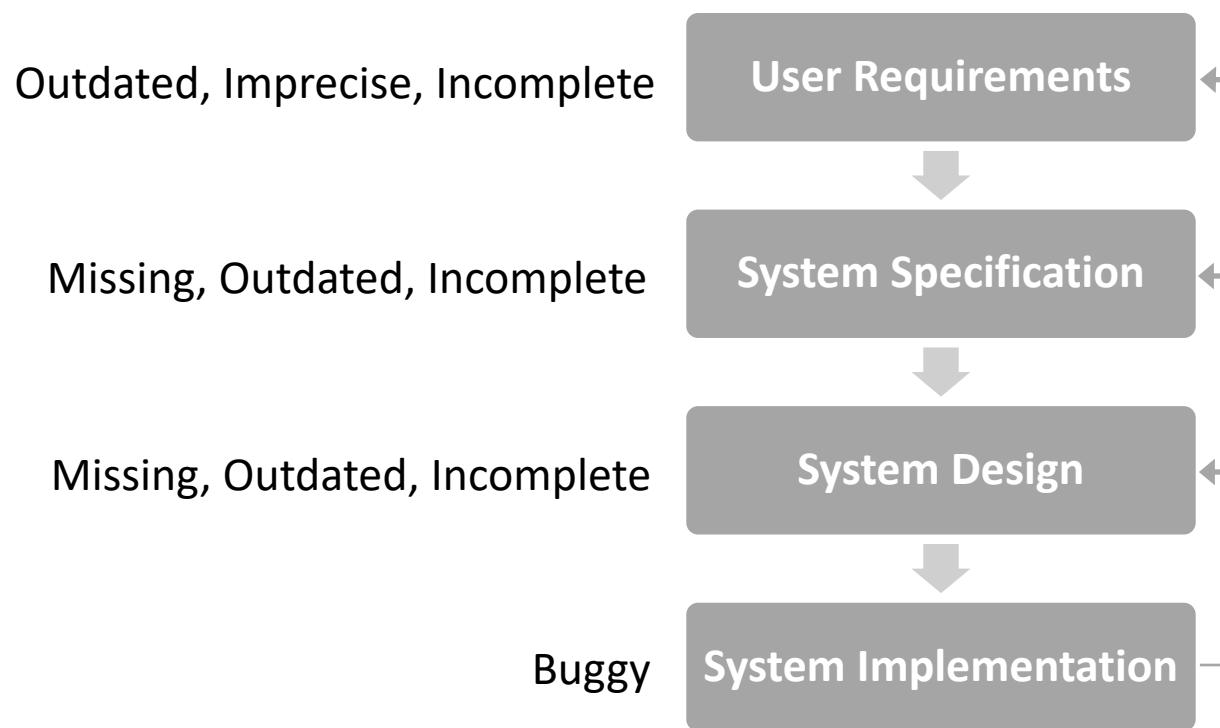


How do you debug the following buggy sort?

```
class BuggySort {  
    public static void sort (int[] input) {  
        int n = input.length;  
        for (int i = 1; i < n - 1; i++) {  
            for (int j = i; j < n - 1; j++) {  
                if (input[j] > input[j+1]) {  
                    int tmp = input[j];  
                    input[j] = input[j+1];  
                    input[j+1] = tmp;  
                }  
            }  
        }  
    }  
}
```

```
class CorrectSort {  
    public static void sort (int[] input) {  
        int n = input.length;  
        for (int i = 0; i < n - 1; i++) {  
            for (int j = 0; j < n - i - 1; j++) {  
                if (input[j] > input[j+1]) {  
                    int tmp = input[j];  
                    input[j] = input[j+1];  
                    input[j+1] = tmp;  
                }  
            }  
        }  
    }  
}
```

Software Engineering





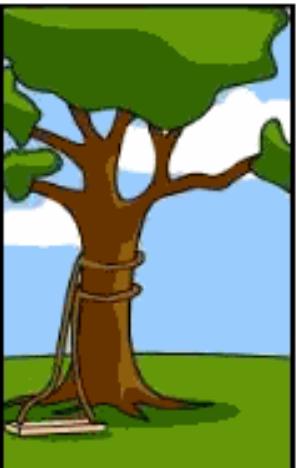
How the customer
explained it



How the Project
Leader understood it



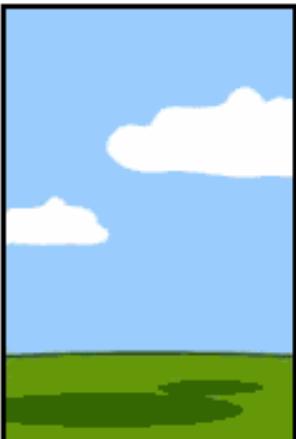
How the Analyst
designed it



How the Programmer
wrote it



How the Business
Consultant described it



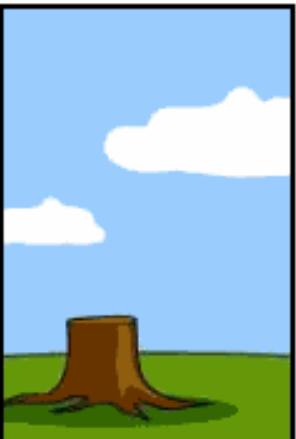
How the project
was documented



What operations
installed



How the customer
was billed



How it was supported



What the customer
really needed

Course Planning

Course Outline

Date	Topic	Date	Topic
Sep. 10	Introduction	Nov. 05	Mobile Testing
Sep. 17	Testing Overview	Nov. 12	Bug Prediction
Sep. 24	Holiday	Nov. 19	Bug Localization
Oct. 01	Holiday	Nov. 26	Delta Debugging
Oct. 08	Automatic Testing	Dec. 03	Automatic Repair
Oct. 15	Security Testing	Dec. 10	Symbolic Execution
Oct. 22	Performance Analysis	Dec. 17	Presentation
Oct. 29	Compiler Testing	Dec. 24	Presentation

For each topic, I will introduce two or three approaches proposed in the literature, and organize multiple discussions.

Presentation and Paper

Survey

- At Least 3 Papers
- Problem Statements
- **Critical Approach Analysis**
- **Pros and Cons**
- Possible Future Directions

New Ideas

- Problem Statement
- **Motivating Example**
- **Detailed Approach**
- Evaluation (Optional)
- Related Work

New Ideas Papers are favored over survey papers, and have a high chance of getting high marks.

Q&A?

Discussion 5 – AI Developers



Do you believe that AI can replace human developers?

