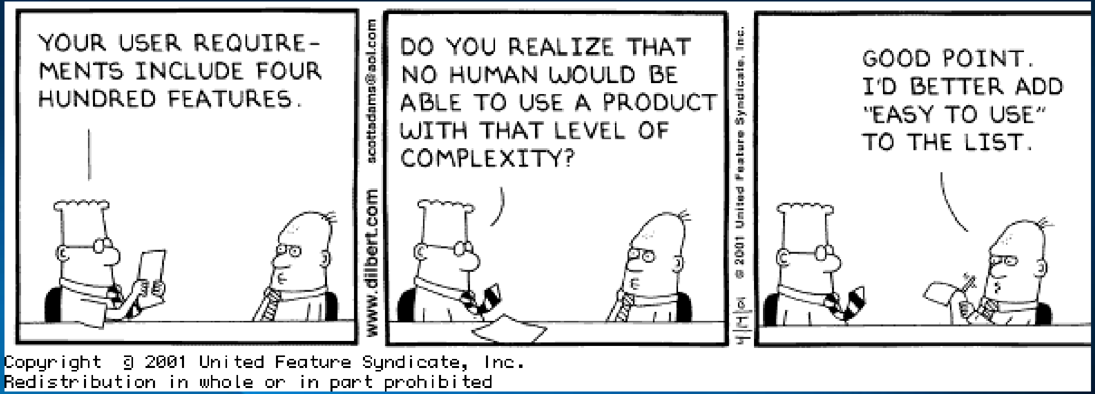
TEACHING

D0020E PROJECT IN COMPUTER SCIENCE 2023/2024 LECTURE 5.1: REQUIREMENTS

Ulf Bodin

LTU 07.11.2023


YOUR USER REQUIREMENTS INCLUDE FOUR HUNDRED FEATURES.
DO YOU REALIZE THAT NO HUMAN WOULD BE ABLE TO USE A PRODUCT WITH THAT LEVEL OF COMPLEXITY?
GOOD POINT. I'D BETTER ADD "EASY TO USE" TO THE LIST.
www.dilbert.com scottadams@aol.com
© 2001 United Feature Syndicate, Inc.
Copyright © 2001 United Feature Syndicate, Inc.
Redistribution in whole or in part prohibited

LULEÅ UNIVERSITY OF TECHNOLOGY

Why requirements?

- We need to give the user/customer what she/he wants!
- *An unhappy customer or investor will most likely never return!*
- *Happiness = richness ?* ☺

LULEÅ UNIVERSITY OF TECHNOLOGY

Scenario

A supermarket wants to create a mobile app for their store to help customers find the groceries and products in their store, and to help them find good deals.

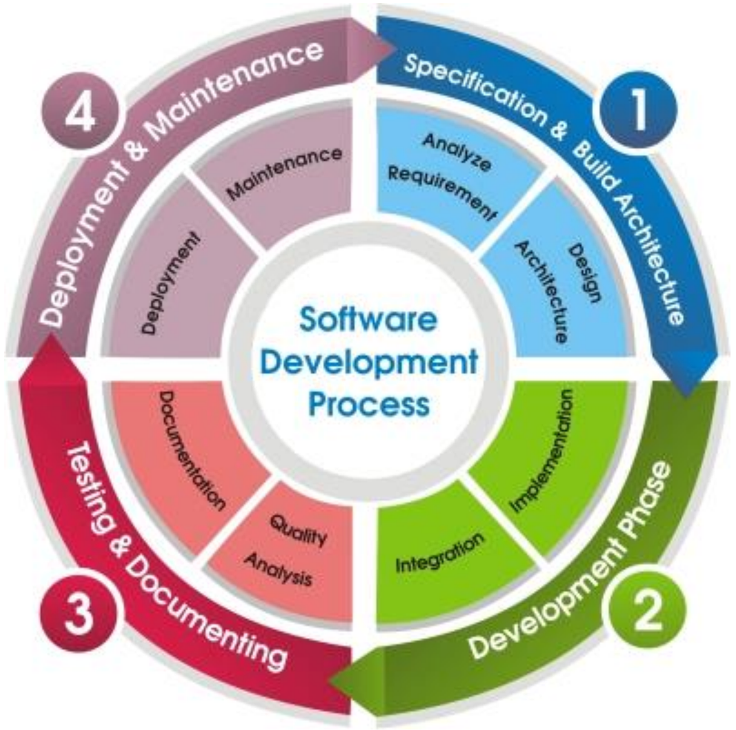
You have been asked to develop the software.

- How do you start?
- How complex will it be?
- How long will it take to develop?

A systematic, disciplined, and quantifiable approach helps a lot!

LULEÅ UNIVERSITY OF TECHNOLOGY

The software development lifecycle



LULEÅ UNIVERSITY OF TECHNOLOGY

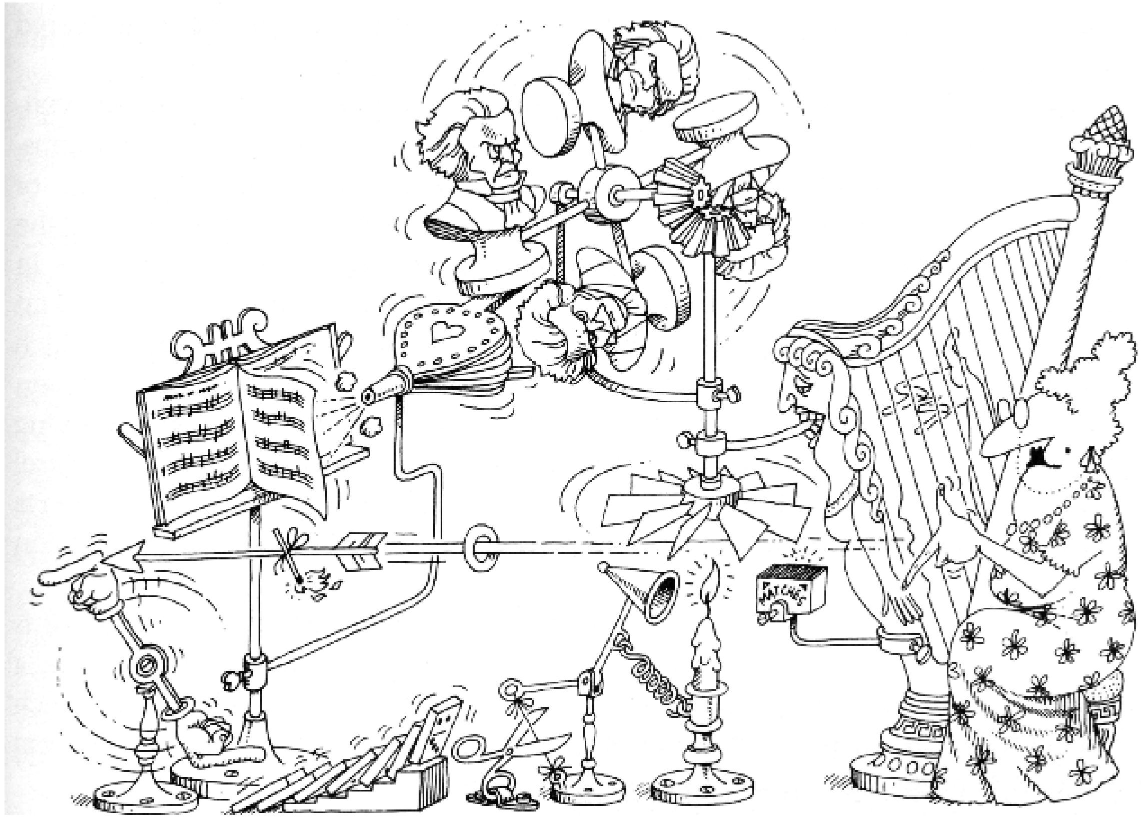
The grand goal

”A successful software organization is one that consistently deploys quality software that meets the **needs of the users**.

An organization that can develop such software in a **timely** and **predictable** fashion, with an efficient and effective use of **resources**, both human and material, is one that has a **sustainable business**.”

On time, within budget and meet requirements!

Design of Software Systems



"The more complex the system, the more open it is to breakdown."

LULEÅ UNIVERSITY OF TECHNOLOGY

The Software Crisis

- Edsger Dijkstra (Turing award winner*) said in1972:

  “*The major cause of the software crisis is that the machines have become several orders of magnitude more powerful!*

  To put it quite bluntly: as long as there were no machines, programming was no problem at all; when we had a few weak computers, programming became a mild problem, and now we have gigantic computers, programming has become an equally gigantic problem.”

What happened?

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

* ACM Turing award, the "Nobel Prize of Computing” (https://en.wikipedia.org/wiki/Turing_Award)

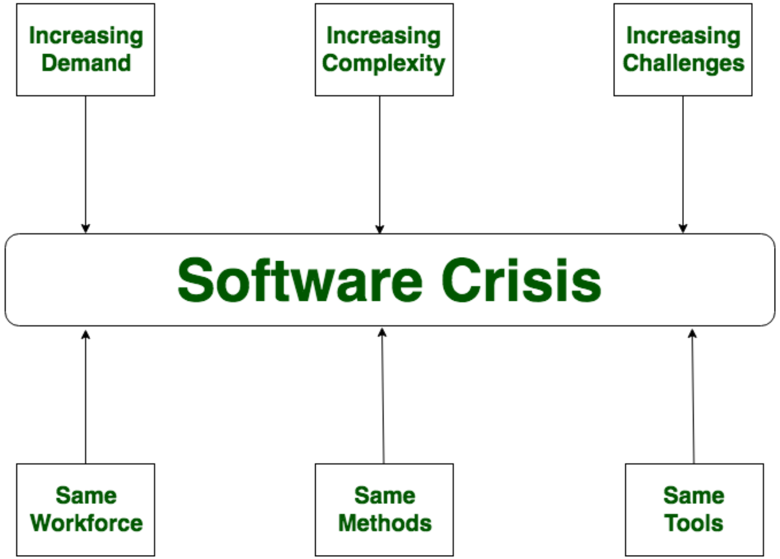
LULEÅ UNIVERSITY OF TECHNOLOGY

The Software Crisis

- Now (Geekforgeeks, Last Updated : 07 April 2022**):

  "There is no single solution to the crisis."

  "One possible solution to a software crisis is ***Software Engineering*** *because it is* ***a systematic, disciplined, and quantifiable approach****."*



** Geekforgeeks, Software Engineering | Software Crisis (geeksforgeeks.org/software-engineering-software-crisis/)

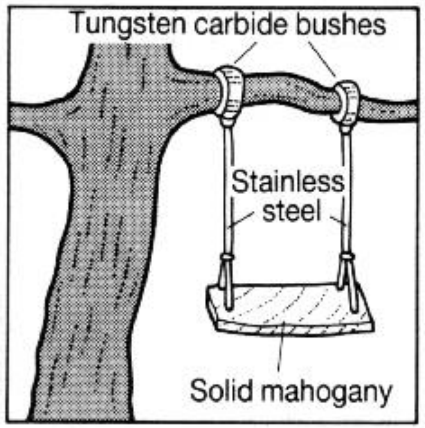
Flexibility leads to Complexity

- Software offers the ultimate flexibility.

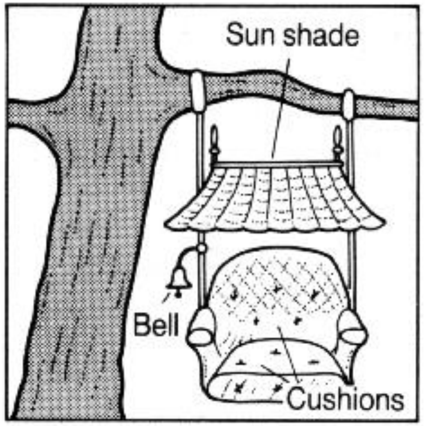
- A developer can express almost any kind of abstraction.

- While the construction industry has uniform building codes and standards, few such standards exists in the software industry.
  - Uniform Modeling Language, UML
  - Design Patterns
  - Software Libraries

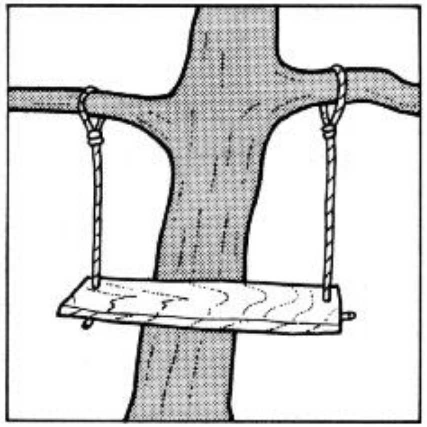
To define a System



What Product Marketing specified

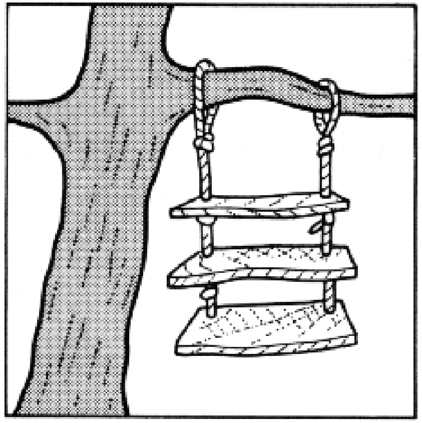


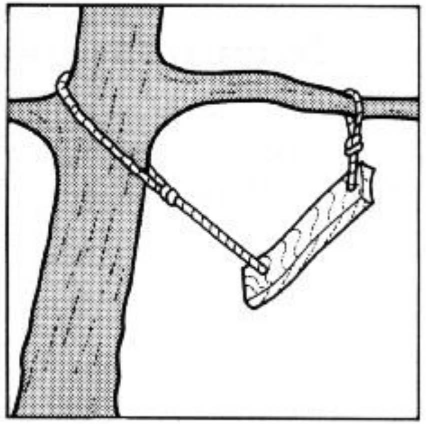
What the salesman promised

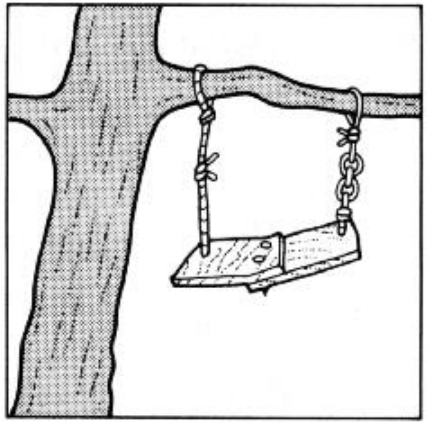


Design group's initial design

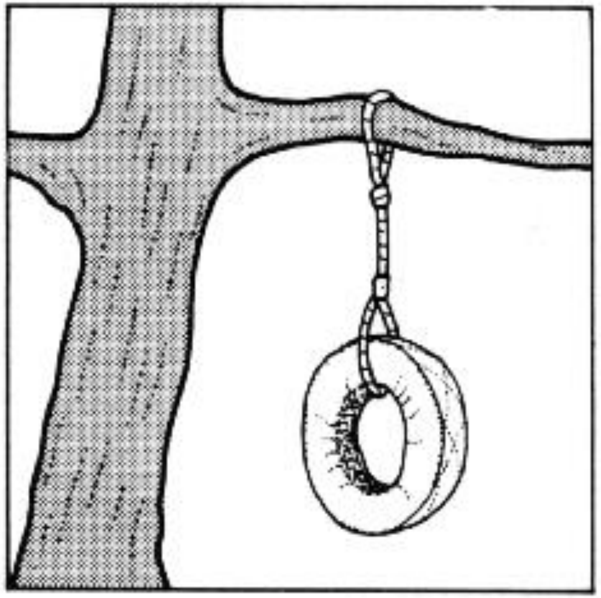
To define a System


Corp. Product Architecture's modified design


Pre-release version


General release version

To define a System

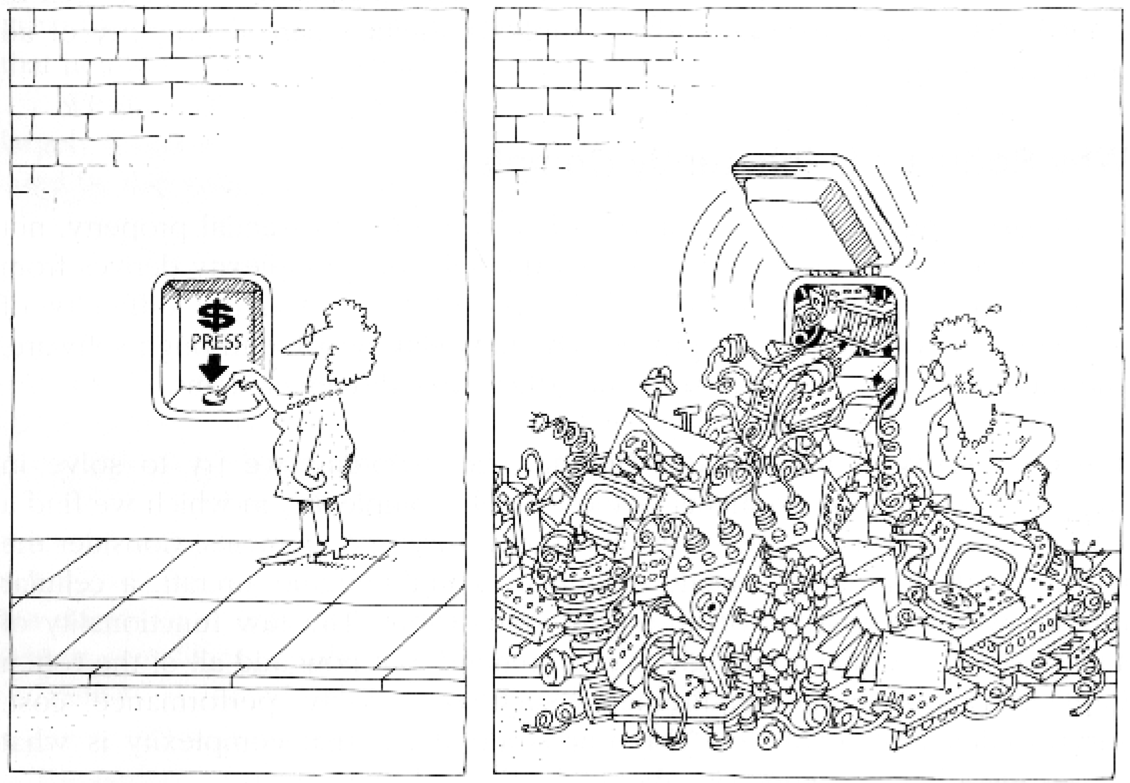

What the customer actually wanted

LULEÅ UNIVERSITY OF TECHNOLOGY

Complexity in specification

- A myriad of **competing**, perhaps **contradictory**, **requirements**:
  - Useability
  - Performance
  - Cost
  - Reliability, etc.
- There is a difference between the users of a system and its developers.
  - Problem space vs. Solution space
- The common way to express requirements is huge volumes of text, which are:
  - difficult to comprehend,
  - open for varying interpretations, and
  - contains design rather than essential requirements.
- **The requirements change during the development.**

Our task: Engineer the illusion of simplicity!

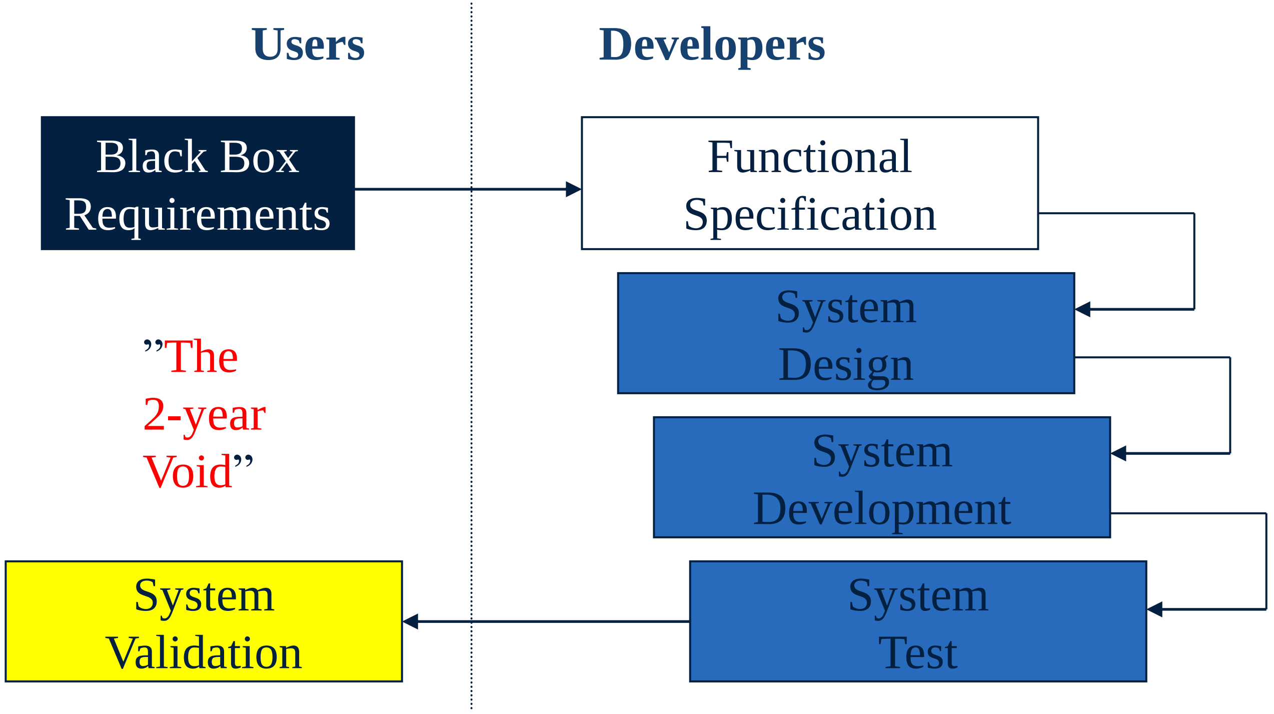


LULEÅ UNIVERSITY OF TECHNOLOGY

Reducing complexity

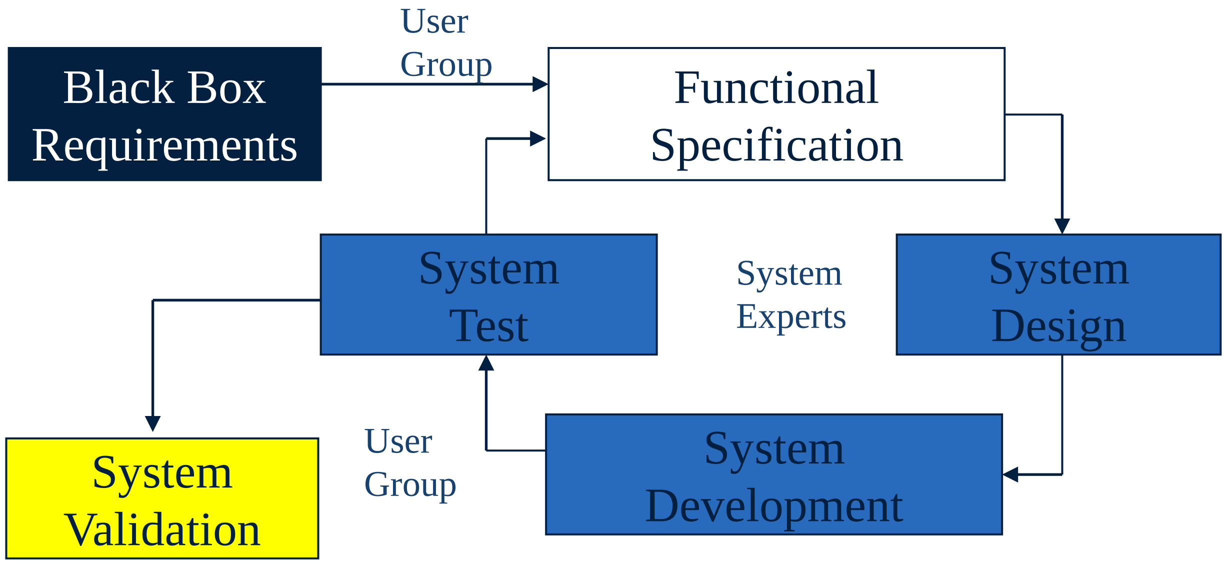
- No one person can understand a big modern system completely, since it often contains millions of lines of code, and thousands of separate modules.
  - Thus, we need to **abstract complexity** using **Object Orientation (OO)** and **UML**!
- We strive to develop **less code**, and instead inventing clever mechanisms to give us the illusion of simplicity.
  - **Reuse** of frameworks of existing designs and code.
  - Thus, we need to identify and reuse **design patterns**!
- More developers means more complex communication and coordination.
  - Thus, use as small a team of engineers as possible!
  - Also, have clear **roles** in your development team.
  - We need a **clearly defined** and **reproducible process**!

The Waterfall development process



LULEÅ UNIVERSITY OF TECHNOLOGY

An evolutionary development process

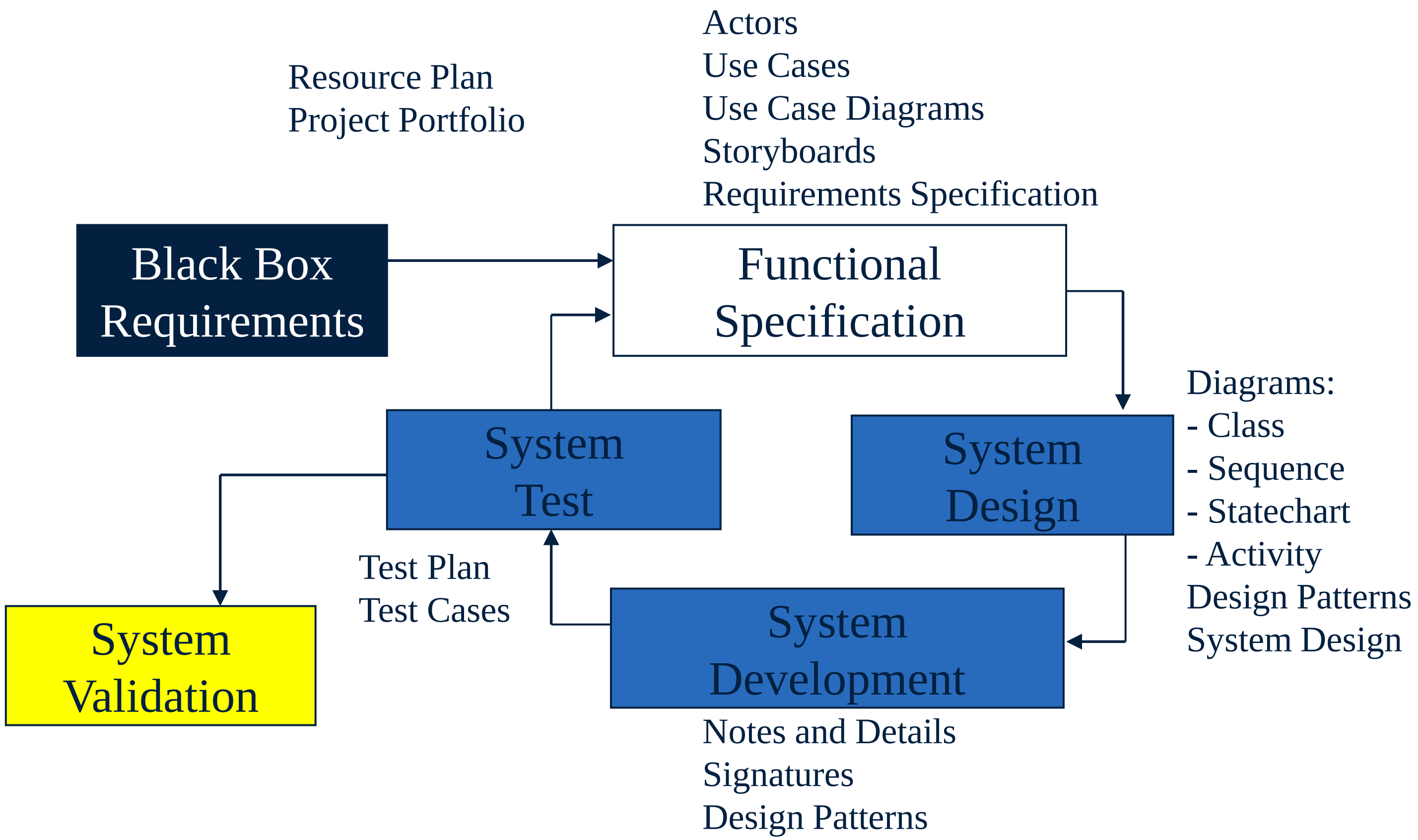


No Void!

Users and System Experts are involved during the whole process!

LULEÅ UNIVERSITY OF TECHNOLOGY

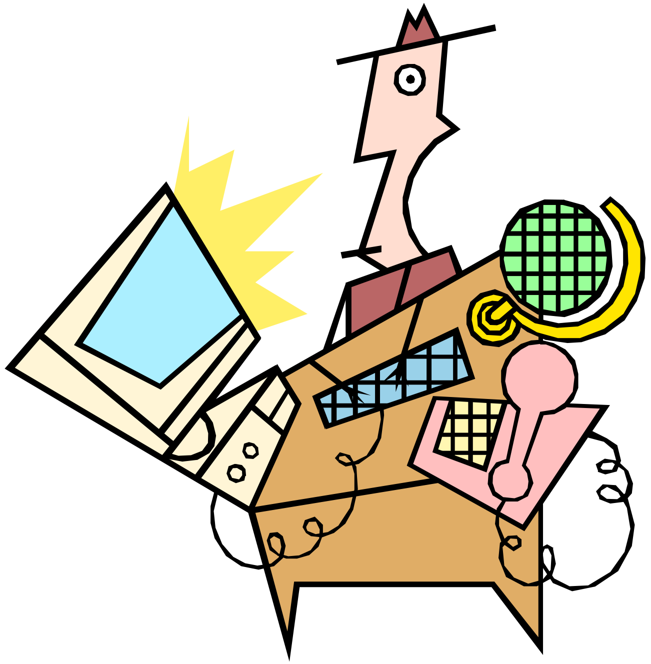
Artifacts



LULEÅ UNIVERSITY OF TECHNOLOGY

Important aspects in developing software

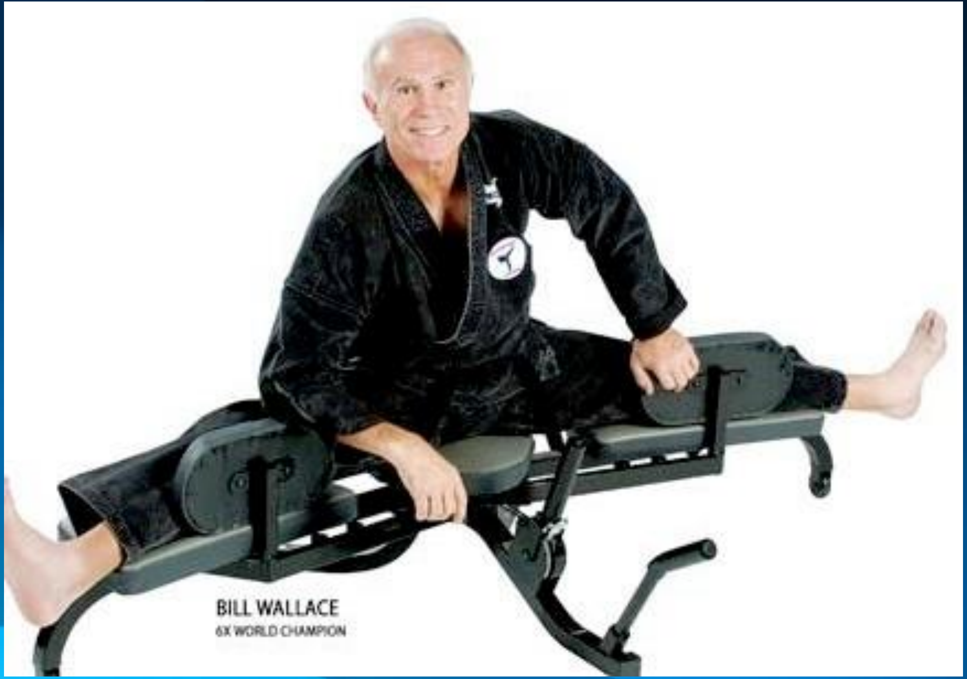
- Software Engineering
  - The Software Process
    - Group dynamics
    - Plans and documentation
  - Requirement specification
    - Use case driven design
    - Metrics
  - Software Design
    - Modeling in UML



LULEÅ UNIVERSITY OF TECHNOLOGY

BREAK – LEG STRETCHER

(swe: Rast – bensträckare)



LULEÅ UNIVERSITY OF TECHNOLOGY

LULEÅ
UNIVERSITY
OF TECHNOLOGY