Software Design & Architecture
(Week-1)

Usama Musharaf

MS-CS (Software Engineering)

LECTURER *(Department of Computer Science)*

FAST-NUCES Peshawar

Course Content

- Software Design Concepts, Design principles,
- Object-Oriented Design with UML,
- System design and software architecture, Object design,
- Mapping design to code, User interface design, Persistent layer design, Web applications design, State machine diagrams and modeling,
- Agile software engineering,
- Design Patterns, Exploring inheritance, Interactive systems with MVC architecture,
- Software reuse. Architectural design issues,
- Software Architecture, Architectural Structures & Styles-, Architectural Patterns,
- Architectural & Design Qualities, Quality Tactics, Architecture documentation, Architectural Evaluation,
- Model driven development.

Recommended Books

- 1. Software Architecture And Design Illuminated By Kai Qian, Xiang Fu, Lixin Tao, Chong-wei Xu, Jones And Bartlett Publishers.
- 2. Software Engineering: A Practitioner's Approach, Roger S. Pressman, Bruce R. Maxim, 8th Ed, McGraw-Hill Education, 2015.
- 3. Object-Oriented Analysis, Design and Implementation, Brahma Dathan, Sarnath Ramnath, 2nd Ed, Universities Press, India, 2014.
- 4. Software Modeling and Design: UML, Use Cases, Patterns, and Software Architectures, Hassan Gomaa, Cambridge University Press, 2011.
- 5. Head First Design Patterns, Eric Freeman, Elisabeth Freeman, Kathy Sierra and Bert Bates, O'Reilly Media, Inc. 2004.

Objective of this course

- To be introduced to principles of good design, and techniques for the evaluation of software design quality.
- To cover the principal architectural issues associated with the design and construction of software systems including architectural design and documentation, component models and technologies, and frameworks.
- To introduce the students to a number of design patterns and their applications.

Agenda of Week # 1

- Introduction to software design
- Importance of software design
- Design vs Architecture
- Design Process Activities
- Levels of Design

What is Design?

- Design is the first step in the development phase for any engineered product or system.

- Design is about HOW the system will perform its functions.

Software Design

- A software design is a meaningful engineering representation of some software product that is to be built.

- “The process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization”.

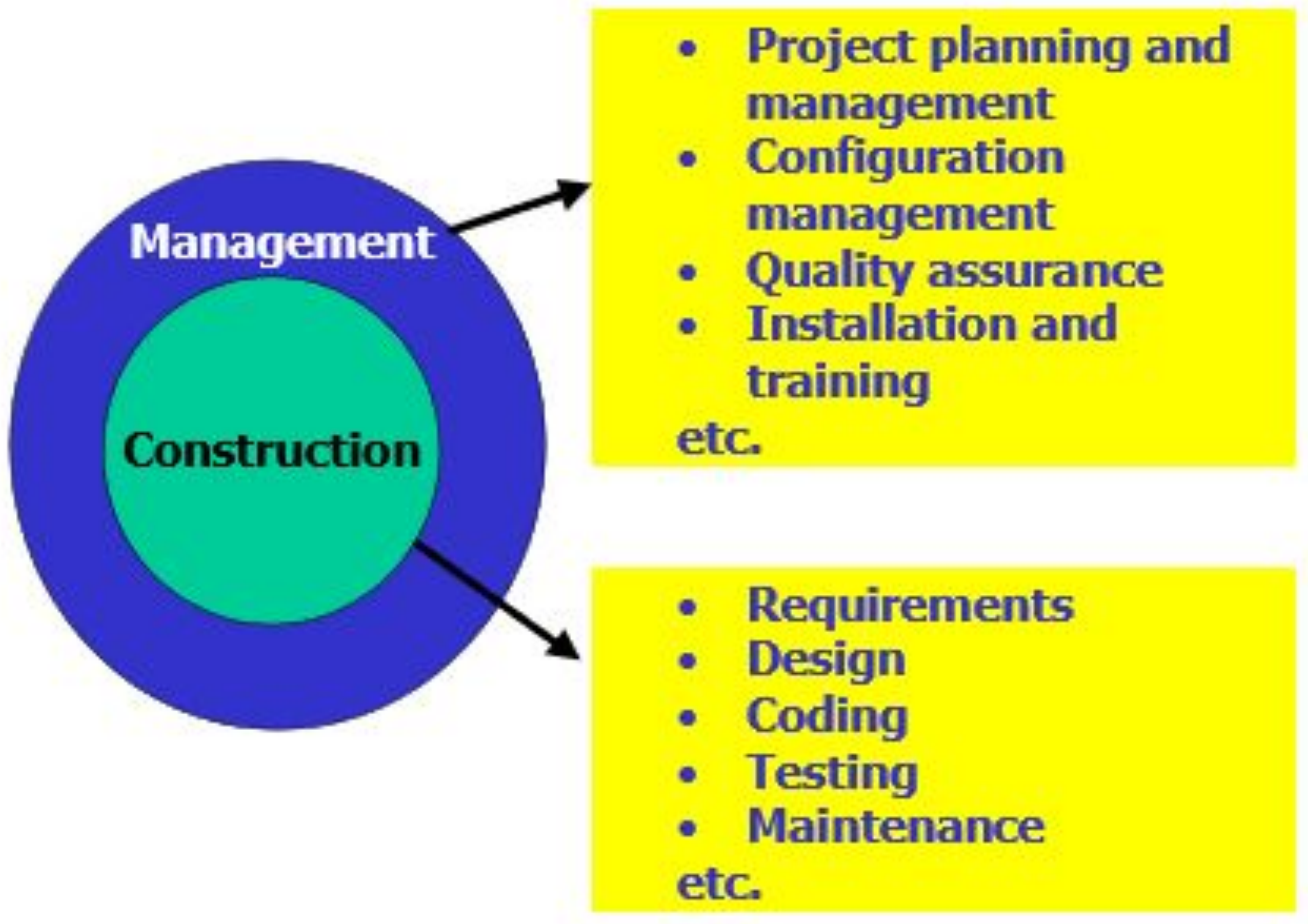
Software Design - Simplified

Requirements specification was about the WHAT the system will do

Design is about the HOW the system will perform its functions

- provides the overall decomposition of the system.
- allows to split the work among a team of developers.
- also lays down the groundwork for achieving non-functional requirements (performance, maintainability, reusability, etc.)

Software Development



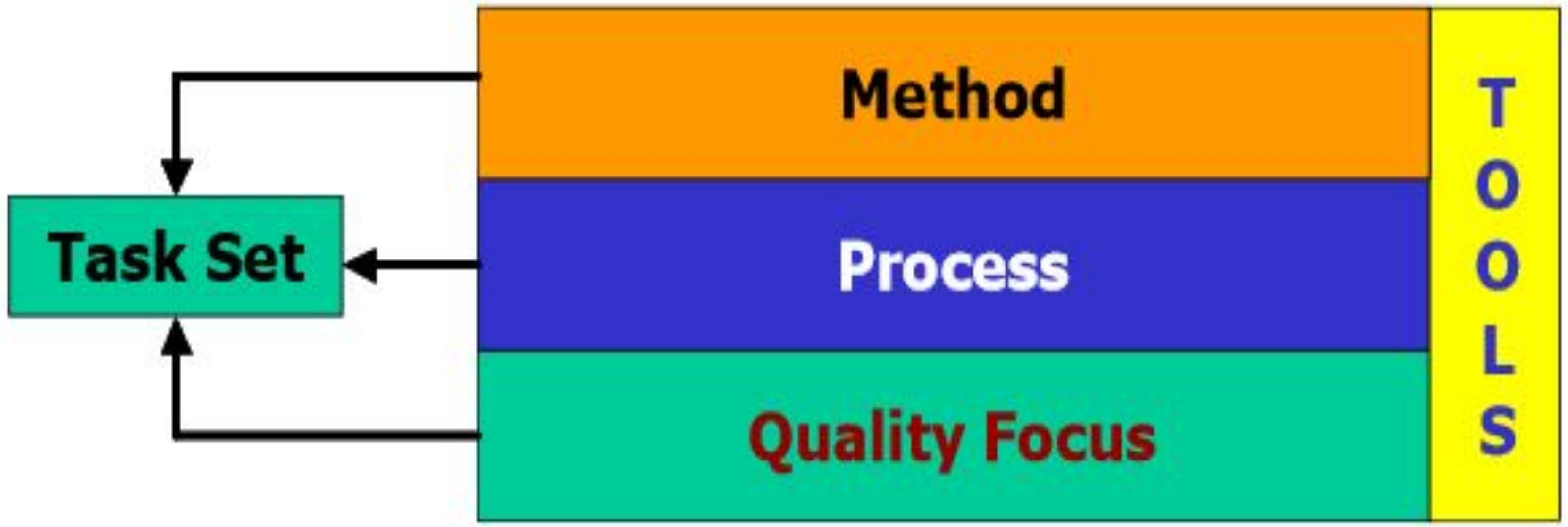
- Project planning and management
- Configuration management
- Quality assurance
- Installation and training

etc.

- Requirements
- Design
- Coding
- Testing
- Maintenance

etc.

Software Development



Software Engineering Phases

- Definition: What?
- Development: How?
- Maintenance: Managing change
- Umbrella Activities: Throughout lifecycle

Definition

REQUIREMENTS DEFINITION AND ANALYSIS

Developer must understand

- Application domain
- Required functionality
- Required performance
- User interface

Definition (cont.)

- Project planning
  - Allocate resources
  - Estimate costs
  - Define work tasks
  - Define schedule

- System analysis
  - Allocate system resources to
    - Hardware
    - Software
    - Users

Development

SOFTWARE DESIGN

- User interface design
- High-level design
  - Define modular components
  - Define major data structures
- Detailed design/Low level Design
  - Define algorithms and procedural detail

Development (cont.)

- Coding
  - Develop code for each module
  - Unit testing
- Integration
  - Combine modules
  - System testing

Maintenance

- Correction - Fix software defects
- Adaptation - Accommodate changes
  - New hardware
  - New company policies
- Enhancement - Add functionality

Why Design is important?

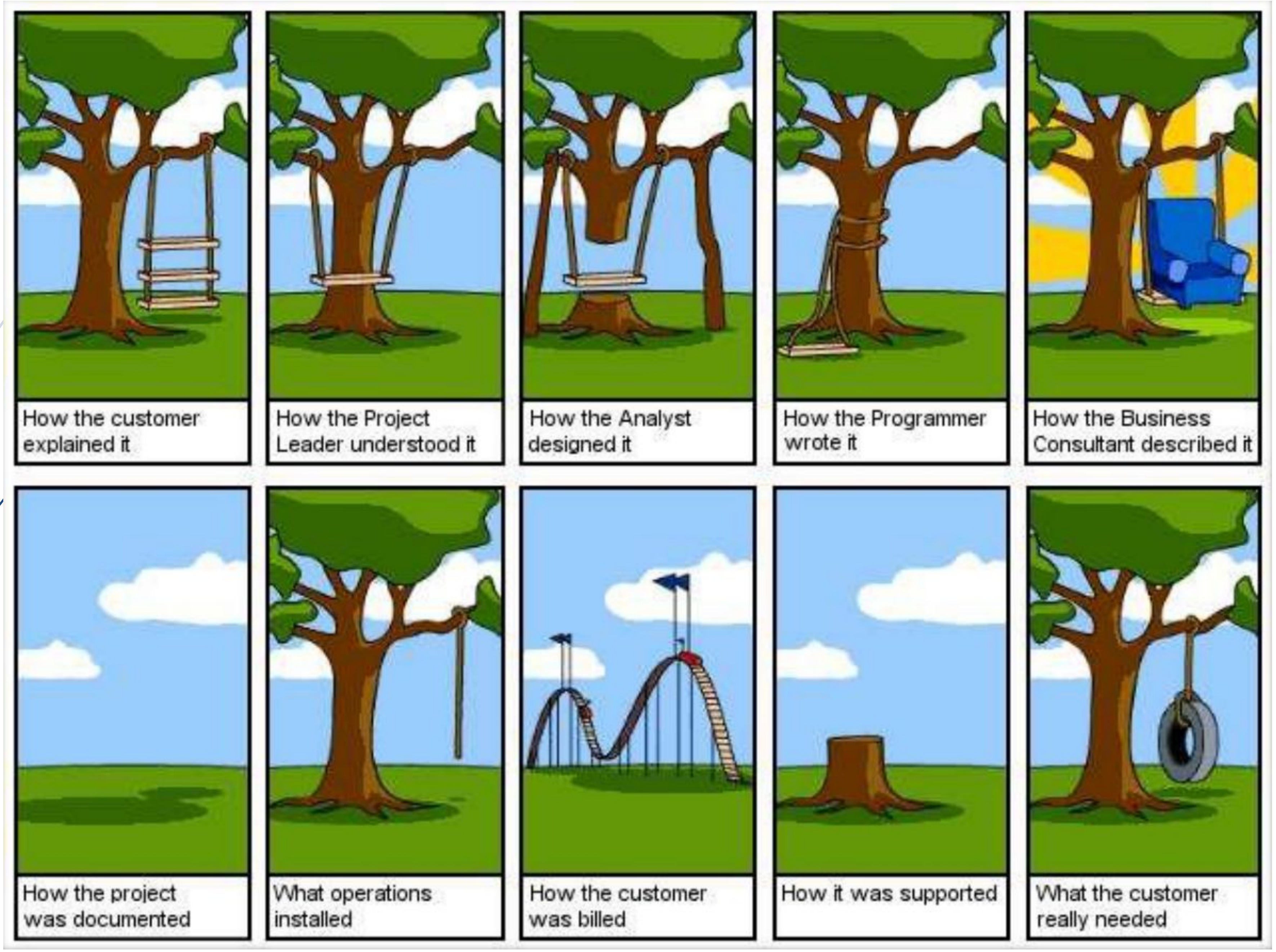
As one of the most complex man-made artifacts “computer software” is very difficult to design.

Without a proper design, we risk building an unstable system

- one that will fail when small changes are made.
- one that may be difficult to test.

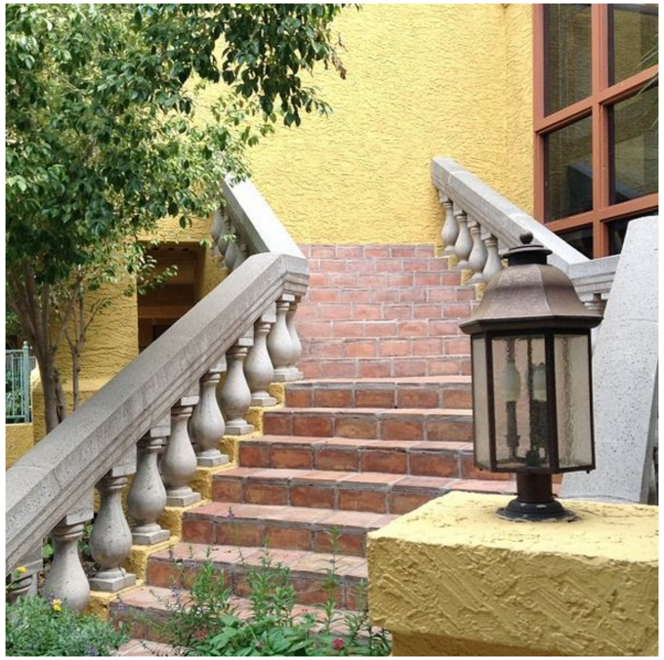
Problems in software development

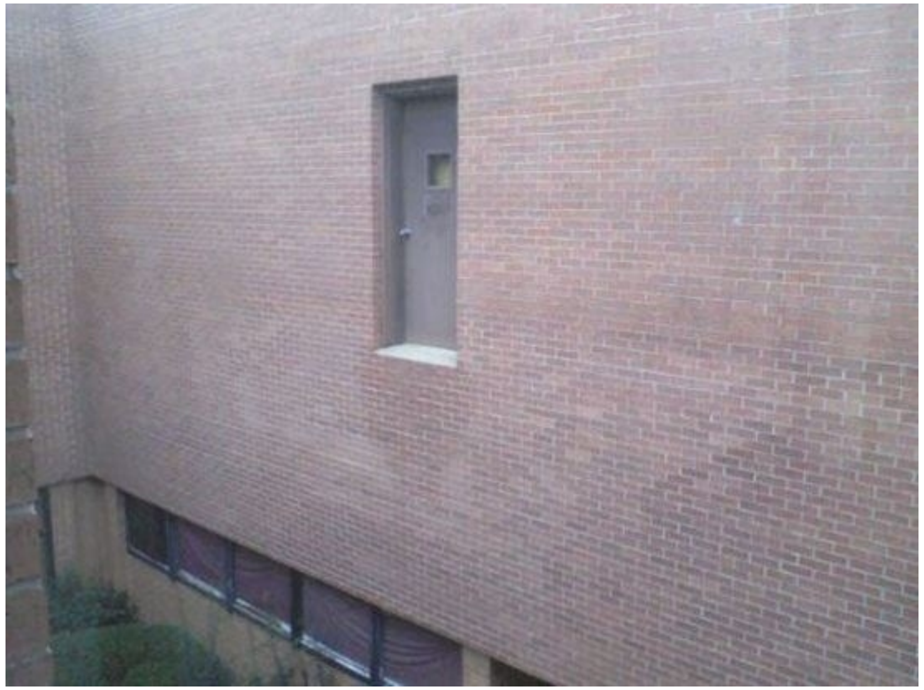
- Common issues
  - The final software does not fulfill the needs of the customer
  - Hard to extend and improve: if you want to add a functionality later its mission impossible
  - Bad documentation
  - Bad quality: frequent errors, hard to use,
  - More time and costs than expected


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

- A staircase that leads right into a wall!

- A door that would drop you 10 feet down!





What do you think is wrong in these real life scenarios?

The requirements are correct!

- A staircase next to the outer wall
- A door on the first floor
- The bridge

The design is flawed!

- The execution based on the design results in disaster.

A clever person solves a problem.

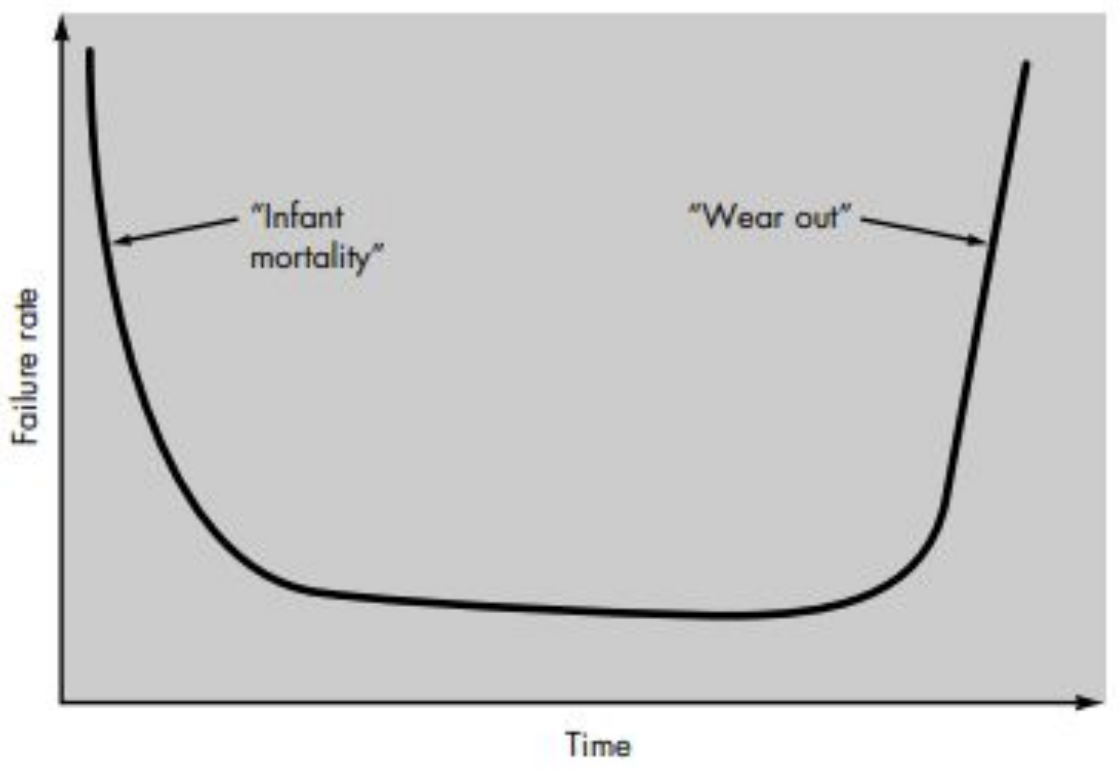
A wise person avoids it.

- Albert Einstein

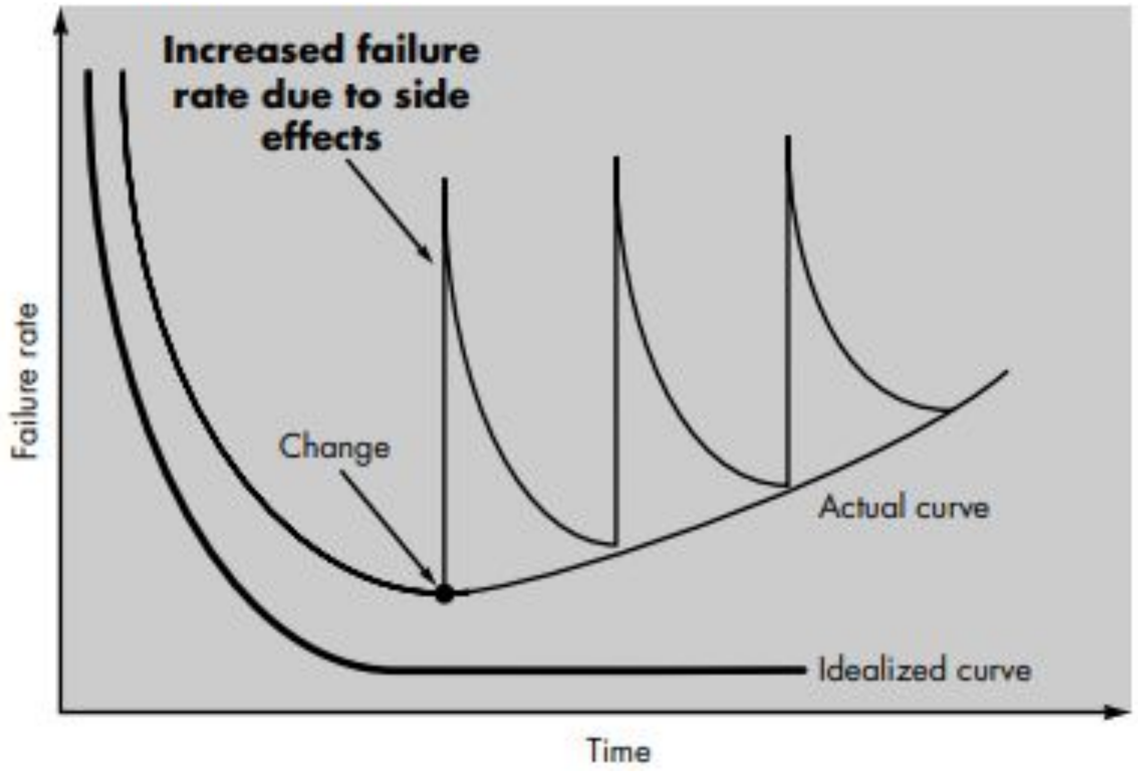
- For a useful software, it has to be ‘engineered’; which involves giving specific attention to every phase of software development.

Hardware vs software

Failure curve for hardware



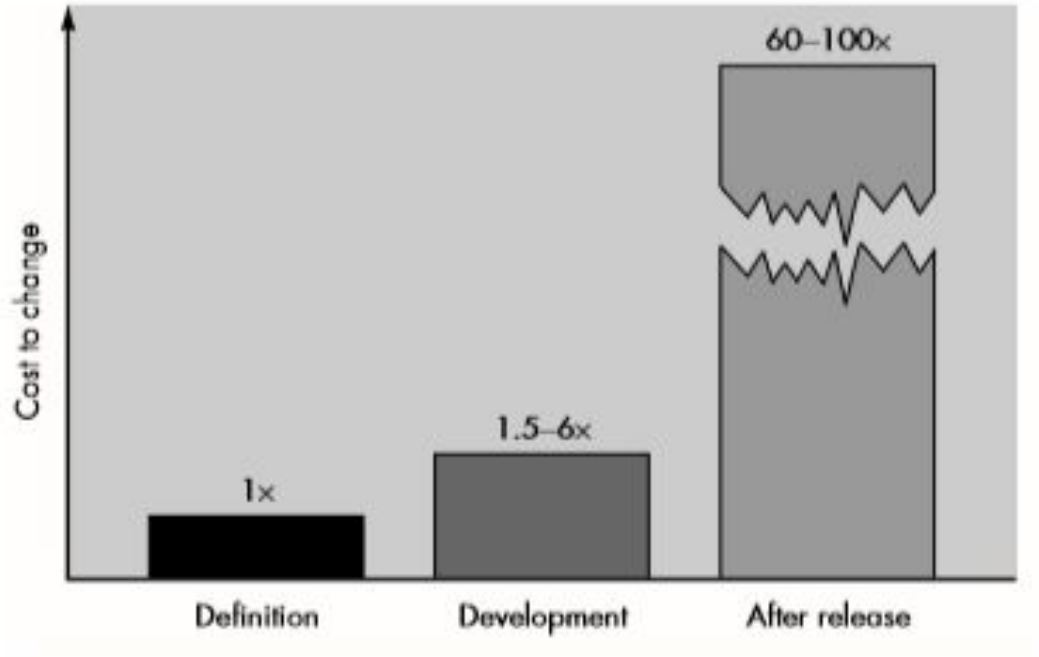
Idealized and actual failure curves for software



Why is software development so difficult?

Changing requirements

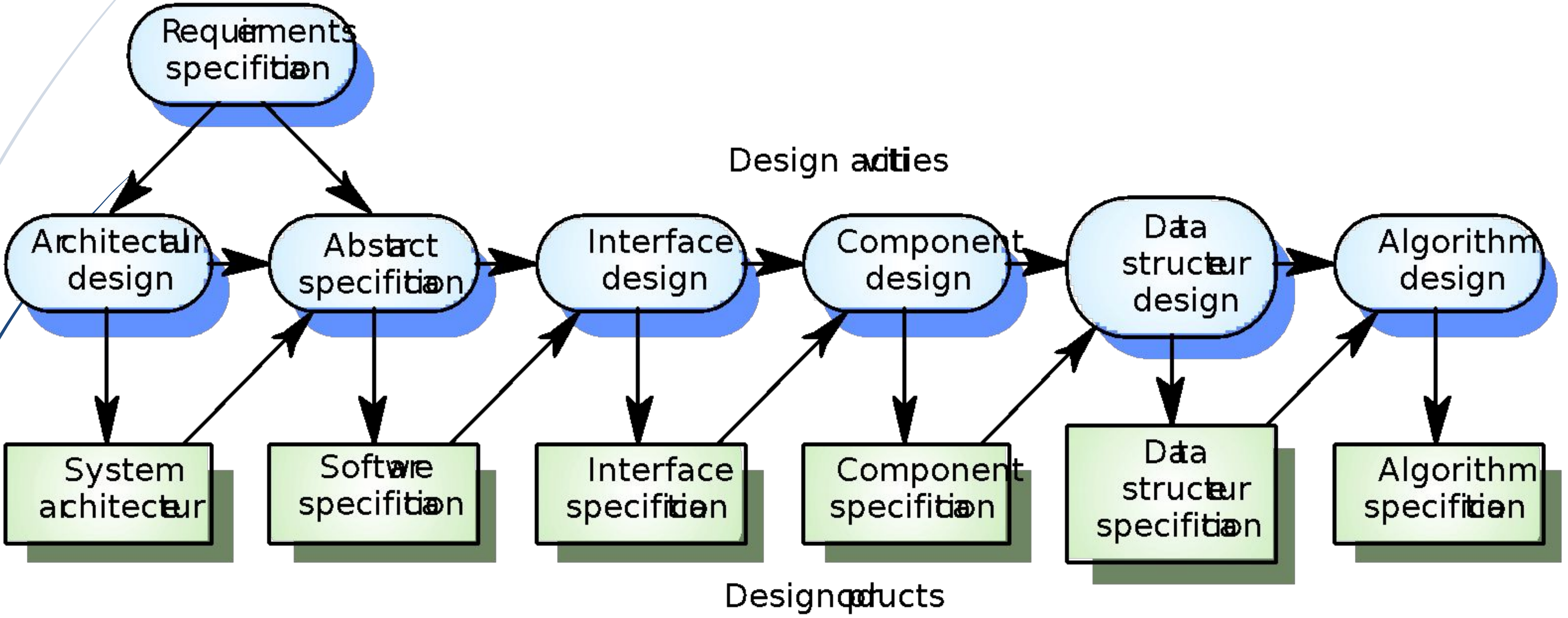
- 5 x cost during development
- up to 100 x cost during maintenance
  - Hardware/software configuration
  - Security requirements
  - Real time requirements
  - Reliability requirements



Design Process Activities

- Architectural design
  - Modules, inter-relationships etc
- Abstract specification
  - Services of each sub-system, constraints etc
- Interface design
  - Interface to other sub-system or outside environment
- Component design
  - Services allocated to components and their interfaces designed
- Data structure design
- Algorithm design

The Software Design Process



Levels of Software Design

Architectural design (high-level design)

- architecture - the overall structure, main modules and their connections
- addresses the main non-functional requirements (e.g., reliability, performance)
- hard to change

Detailed design (low-level design)

- the inner structure of the main modules
- detailed enough to be implemented in the programming language

$$\begin{tikzcd}
\text{Architecture} \arrow[d] \\
\text{Design} \arrow[d] \\
\text{Implementation}
\end{tikzcd}$$

Design vs. Architecture

- Architecture is concerned with the selection of architectural elements, their interaction, and the constraints on those elements and their interactions

- Design is concerned with the modularization and detailed interfaces of the design elements, their algorithms and procedures, and the data types needed to support the architecture and to satisfy the requirements.

- Architecture…is specifically not about…details of implementations (e.g., algorithms and data structures.)

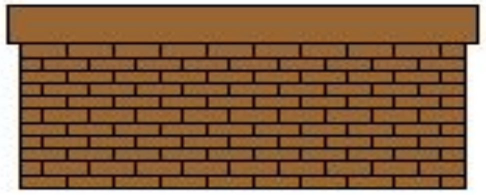
Software Development

- Lists
- Arrays
- Class
- Object
- Procedures
- Functions
- Algorithms
- Etc.

```
//********************************************************************
import InventoryItem;
import java.util.StringTokenizer;
import java.io.*;

public class Inventory
{
   //-----------------------------------------------------------------
   //  Reads data about a store inventory from an input file,
   //  creating an array of InventoryItem objects, then prints them.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      final int MAX = 100;
      InventoryItem[] items = new InventoryItem[MAX];
      StringTokenizer tokenizer;
      String line, name, file="inventory.dat";
      int units, count = 0;
      float price;

      try
for (int scan = 0; scan < count; scan++)
         System.out.println (items[scan]);
      }
      catch (FileNotFoundException exception)
      {
         System.out.println ("The file " + file + " was not found.");
      }
      catch (IOException exception)
      {
         System.out.println (exception);
      }
   }
}
```

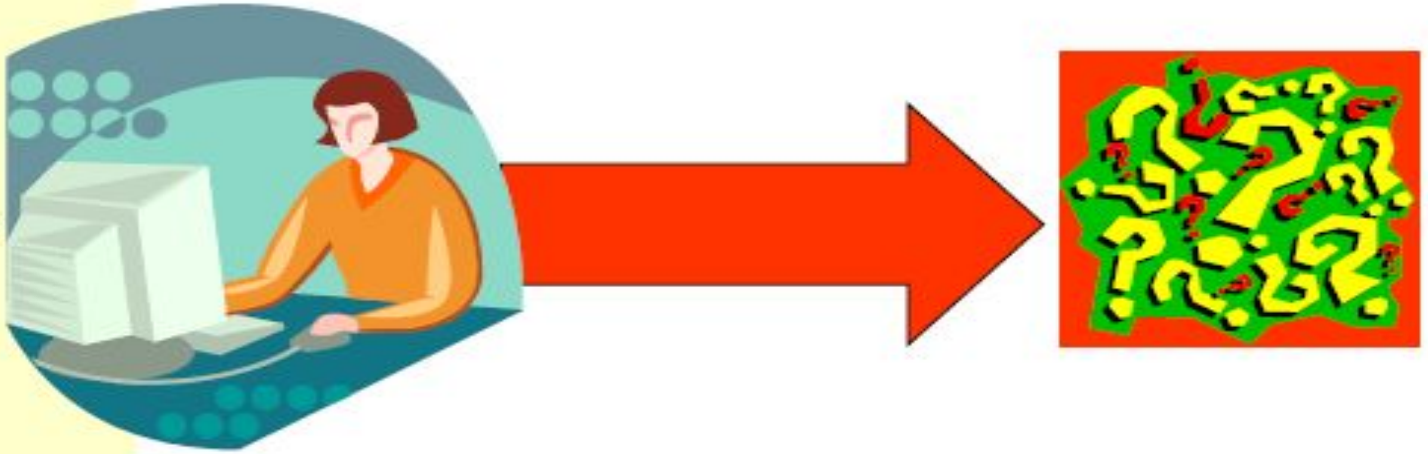


Large-scale,complex software systems…

- Large (Distributed) System
- Many people working on the same problem
- Overly Complex
- Millions of Code…
- Should be deliverd on time and within budget!

Coding only will not do…

- Lists
- Arrays
- Class
- Object
- Procedures
- Functions
- Algorithms
- Etc.



More programmers...?

Software Architecture

The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships between them.

HAVE A GOOD DAY !