SOFTWARE ENGINEERING
(Week-1)

USAMA MUSHARAF

MS-CS (Software Engineering)

LECTURER (Department of Computer Science)

FAST-NUCES PESHAWAR

COURSE CONTENT

- Introduction to Computer-based System Engineering;
- Project Management; Software Specification; Requirements Engineering, System Modeling; Requirements Specifications; Software Prototyping;
- Software Design: Architectural Design, Object-Oriented Design, Function-Oriented Design, User Interface Design;
- Quality Assurance; Processes & Configuration Management;
- Introduction to advanced issues:
- Reusability, Patterns;
- Assignments and projects on various stages and deliverables of SDLC.

RECOMMENDED BOOKS

Text Books

- Software Engineering, Sommerville I., 10th Edition, Pearson Inc., 2014
- Software Engineering, A Practitioner’s Approach, Pressman R. S.& Maxim B. R., 8th Edition, McGraw-Hill, 2015.

OBJECTIVE OF THIS COURSE

- To familiarize students to the fundamental concepts, techniques, processes, methods and tools of Software Engineering,

- To help students to develop basic skills that will enable them to construct software of high quality software that is reliable, and that is reasonably easy to understand, modify and maintain.

- To foster an understanding of why these skills are important.

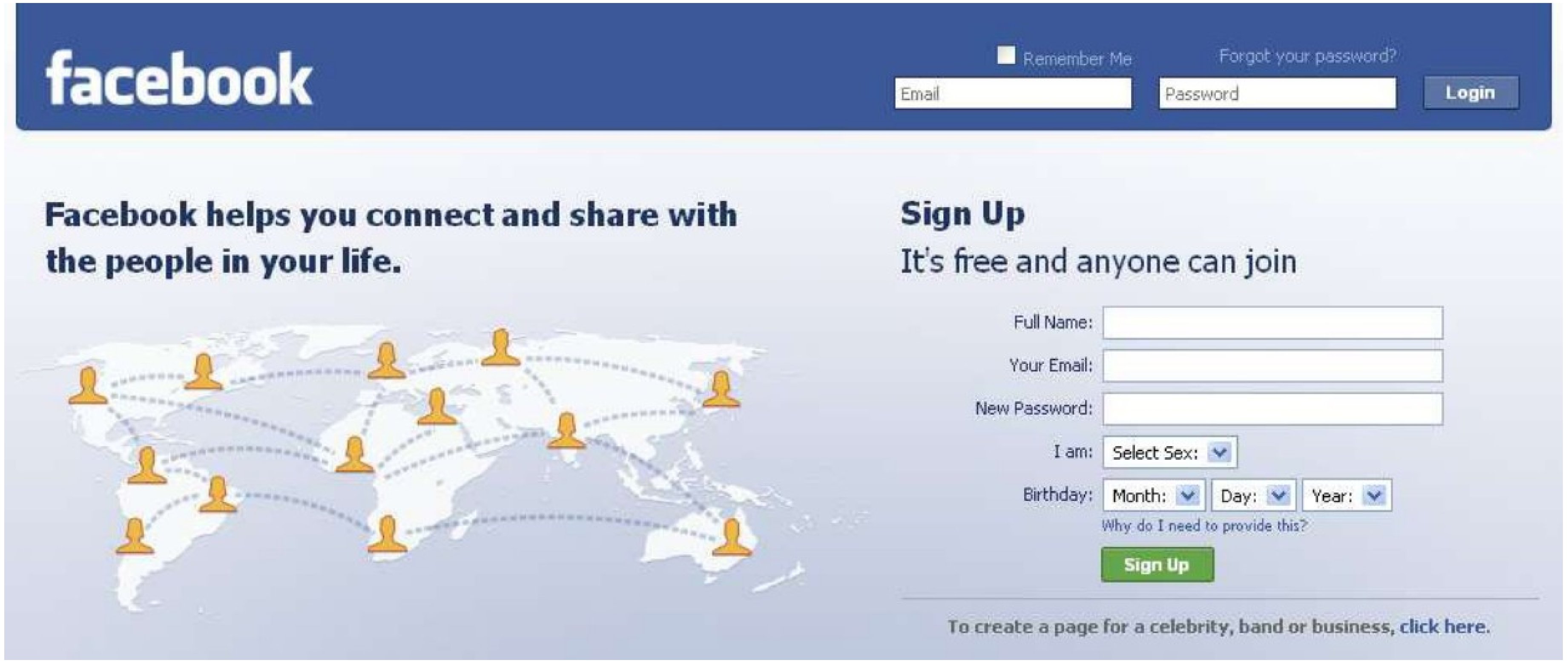
AGENDA OF WEEK # 1

1. Introduction to Software Engineering
2. Importance of Software Engineering
3. Phases of Software Engineering
   - Definition
   - Development
   - Maintenance
4. Related Activities in Software Engineering
5. Problems in Software Development
6. Software Myths

Software can have huge impact in any aspect of our society

WHERE CAN WE FIND SOFTWARE?

SOME POPULAR ONES…



Facebook helps you connect and share with the people in your life.

Sign Up

It's free and anyone can join

Full Name:

Your Email:

New Password:

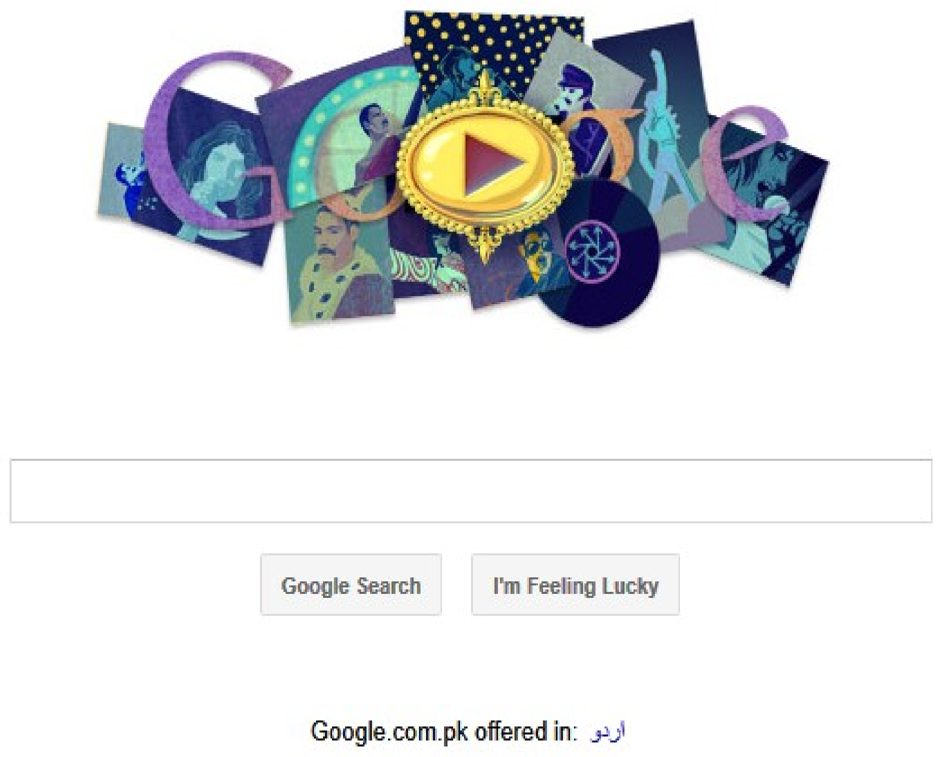
I am: Select Sex:

Birthday: Month: Day: Year:

Why do I need to provide this?

To create a page for a celebrity, band or business, click here.

SOME POPULAR ONES…



AND EVEN IN…

CONCLUSION

Software is almost everywhere!!!

SOFTWARE APPLICATIONS

- ✓ Personal Computer Software
- ✓ Business Software
- ✓ System Software
- ✓ Real Time Software
- ✓ Engineering & Scientific Software
- ✓ Embedded Software
- ✓ Web Based Software
- ✓ Artificial Intelligence Software

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

A clever person solves a problem.

A wise person avoids it.

- Albert Einstein

SOLUTION

SOFTWARE ENGINEERING

SE HISTORY

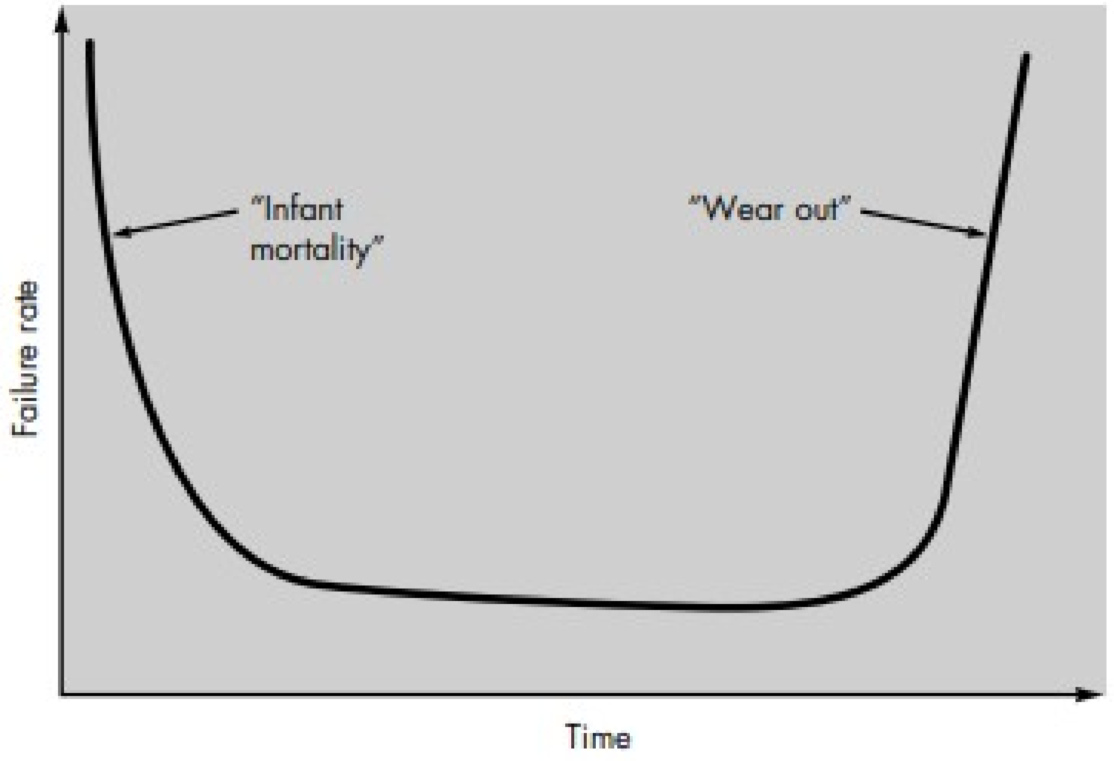
- SE introduced first in 1968 – conference about “software crisis” when the introduction of third generation computer hardware led more complex software systems then before.

- Early approaches based on informal methodologies leading to

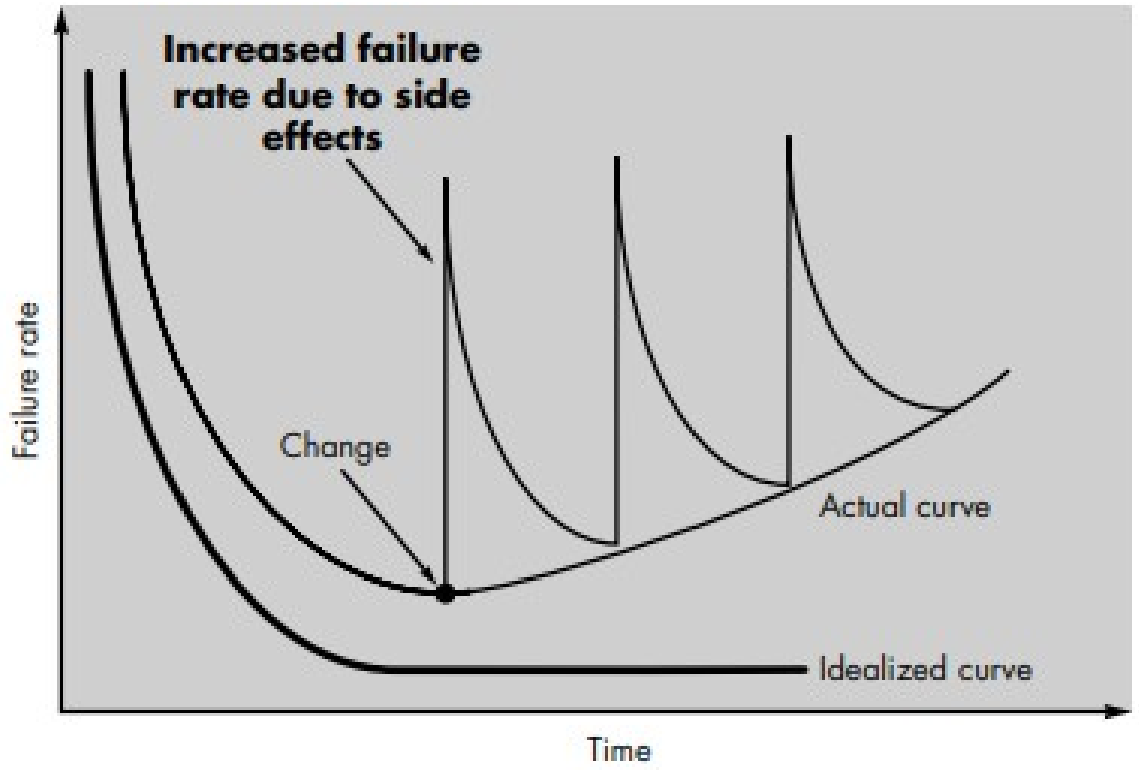
- Need for new methods and techniques to manage the production of complex software.

HARDWARE VS SOFTWARE

Failure curve for hardware



Idealized and actual failure curves for software



WHAT IS ENGINEERING?

“The process of productive use of scientific knowledge is called engineering.”

WHAT IS SOFTWARE ENGINEERING?

- Systematic approach for developing software

- Methods and techniques to develop and maintain quality software to solve problems.

- Study of the *principles* and *methodologies* for developing and maintaining software systems.

WHAT IS SOFTWARE ENGINEERING?

- *Practical* application of scientific knowledge in the design and construction of computer programs and the associated *documentation* required to develop, operate, and maintain them.

- Deals with establishment of *sound engineering principles and methods* in order to *economically* obtain software that is reliable and works on real machines.

WHAT IS SOFTWARE ENGINEERING?

According to the IEEE

Software is:

"Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system".

THE ROLE OF SOFTWARE ENGINEERING

A bridge from customer needs to programming implementation

Customer
Software Engineering
Programmer

<u>First law of software engineering</u>

Software engineer is willing to learn the problem domain
(problem cannot be solved without understanding it first)

WHAT IS THE DIFFERENCE BETWEEN SOFTWARE ENGINEERING AND COMPUTER SCIENCE?

Computer Science

Software Engineering

is concerned with

- ➢ theory
- ➢ fundamentals

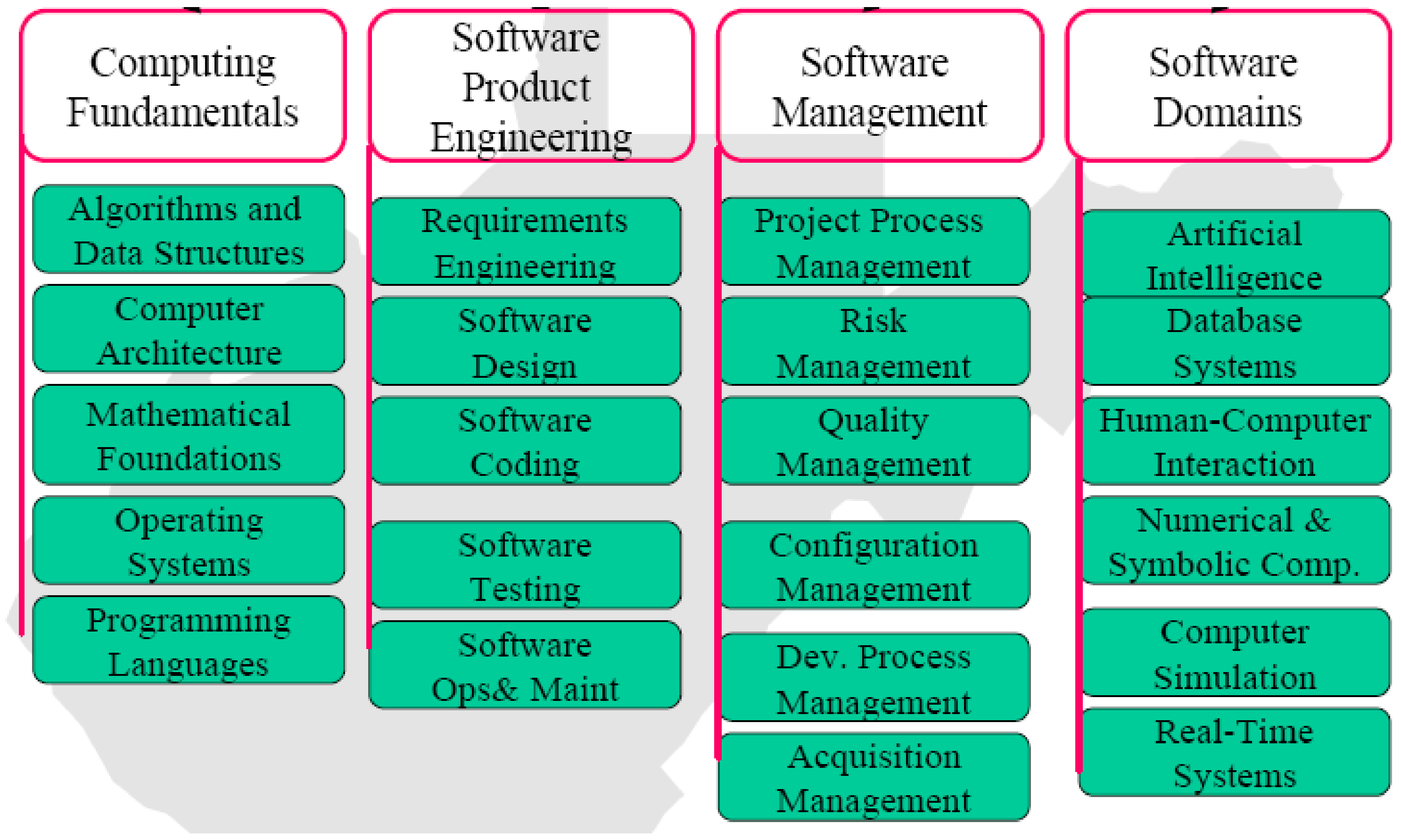
Algorithms, data structures, complexity theory, numerical methods

- ➢ the practicalities of developing
- ➢ delivering useful software

SE deals with practical problems in complex software products

Computer science theories are currently insufficient to act as a complete underpinning for software engineering, BUT it is a foundation for practical aspects of software engineering.

SOFTWARE ENGINEERING BODY OF KNOWLEDGE



WHAT ARE THE ATTRIBUTES OF GOOD SOFTWARE?

The software should deliver the required functionality and performance to the user and should be maintainable, dependable and usable.

- **Maintainability**
  - Software must evolve to meet changing needs
- **Dependability**
  - Software must be trustworthy
- **Efficiency**
  - Software should not make wasteful use of system resources
- **Usability**
  - Software must be usable by the users for which it was designed

and much more....

WELL ENGINEERED SOFTWARE?

It is reliable

It has good user-interface

It has acceptable performance

It is of good quality

It is cost-effective

WHAT ARE THE KEY CHALLENGES FACING SOFTWARE ENGINEERING?

Software engineering in the 21st century faces three key challenges:

- **Legacy systems**
  - Old, valuable systems must be maintained and updated.
- **Heterogeneity**
  - Systems are distributed and include a mix of hardware and software.
- **Delivery**
  - There is increasing pressure for faster delivery of software.

QUESTIONS ADDRESSED BY SOFTWARE ENGINEERING

- How do we ensure the quality of the software that we produce?
- How do we meet growing demand and still maintain budget control?
- How do we avoid disastrous time delays?

WHY APPLY SOFTWARE ENGINEERING TO SYSTEMS?

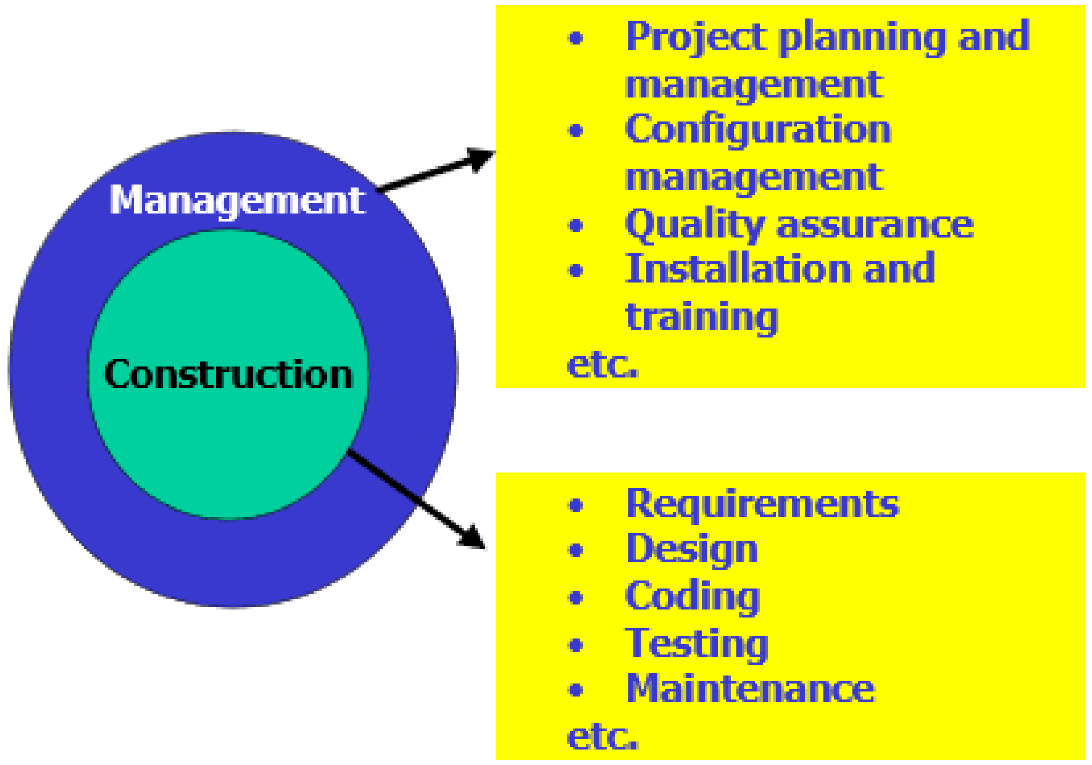
- Provide an understandable process for system development.
- Develop systems and software that are maintainable and easily changed.
- Develop robust software systems.

SOME IMPORTANT SOFTWARE ENGINEERING RELATED ACTIVITIES

- Project Management
- Requirement Engineering
- Software Design
- Coding
- Testing

- Software Quality Assurance
- Software Configuration Management
- Software Integration

SOFTWARE DEVELOPMENT



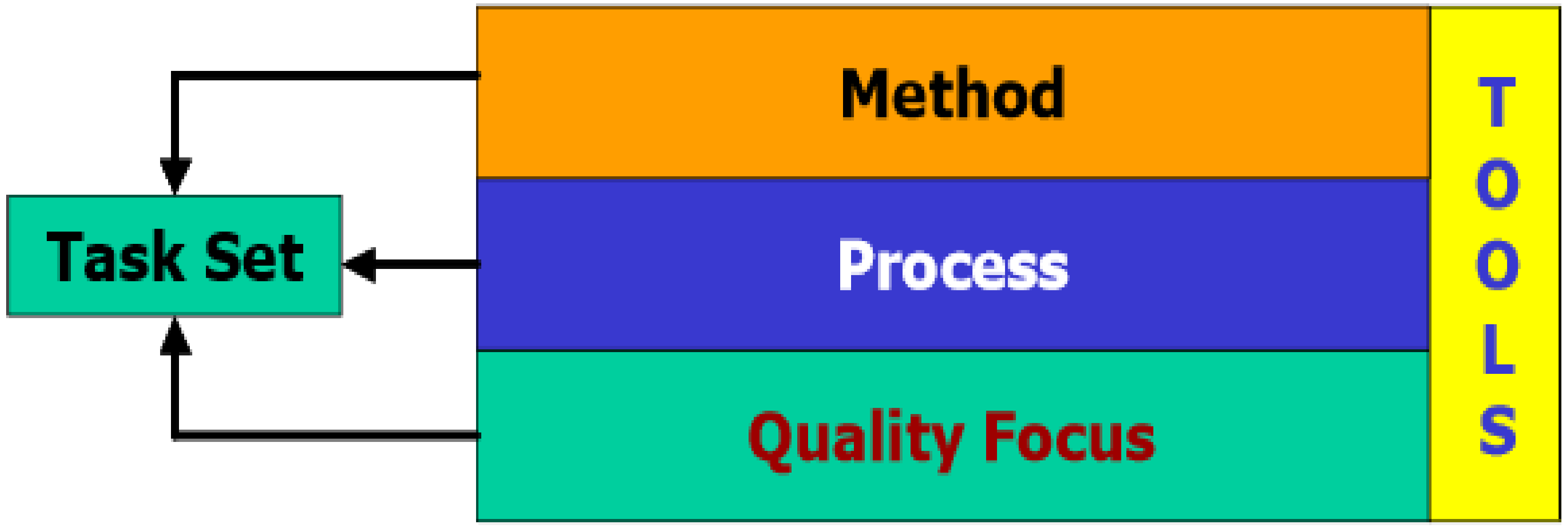
- Project planning and management
- Configuration management
- Quality assurance
- Installation and training

etc.

- Requirements
- Design
- Coding
- Testing
- Maintenance

etc.

SOFTWARE DEVELOPMENT



SOFTWARE DEVELOPMENT

The activities involved in software development can broadly be divided into two major categories

- Construction
- Management

SOFTWARE DEVELOPMENT

Construction

Construction activities are related to the development of software.

- ❖ Requirement Gathering
- ❖ Design Development
- ❖ Coding
- ❖ Testing

SOFTWARE DEVELOPMENT

Management

Management activities are kind of umbrella activities that are used to smoothly and successfully perform the construction activities

- Project Planning and Management
- Configuration Management
- Software Quality Assurance
- Installation and Training

SOFTWARE DEVELOPMENT

Questions that have to answer in Software Development

1. What is the problem to be solved?
2. What are the characteristics of the entity that is used to solve the problem?
3. How will the entity be realized?
4. How will the entity be constructed?
5. What approach will be used to uncover errors that were made in the design and construction of the entity?

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 IS SOFTWARE DEVELOPMENT SO DIFFICULT?

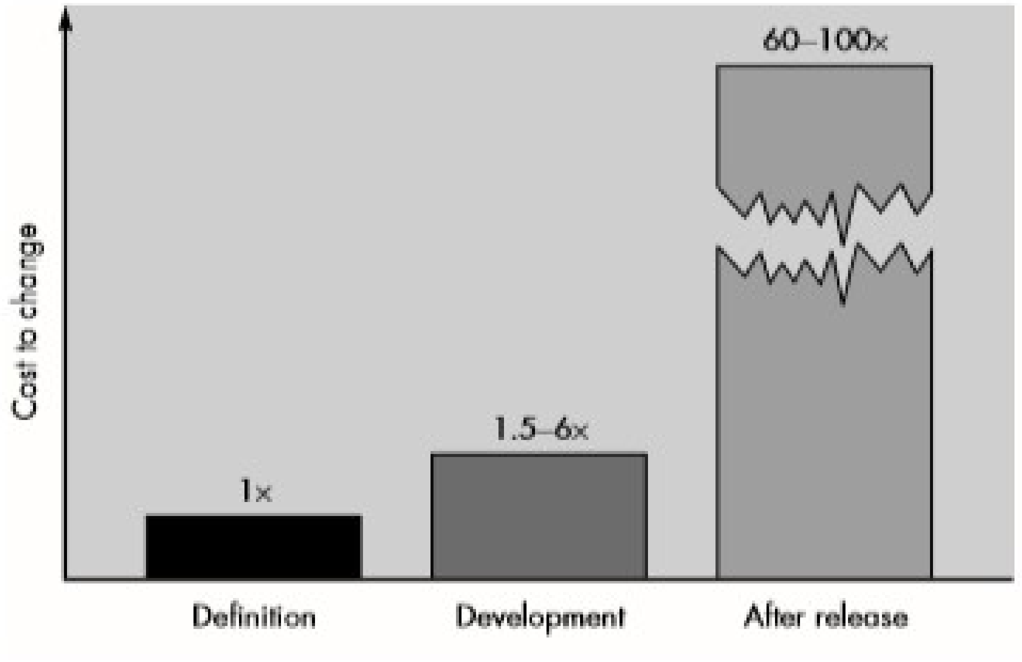
- Communication
  - Between customer and developer
- Poor problem definition is largest cause of failed software projects
  - Within development team
- More people = more communication
- New programmers need training

WHY IS SOFTWARE DEVELOPMENT SO DIFFICULT?

Changing requirements

- 5 x cost during development
- up to 100 x cost during mainter

- Hardware/software configuration
- Security requirements
- Real time requirements
- Reliability requirements

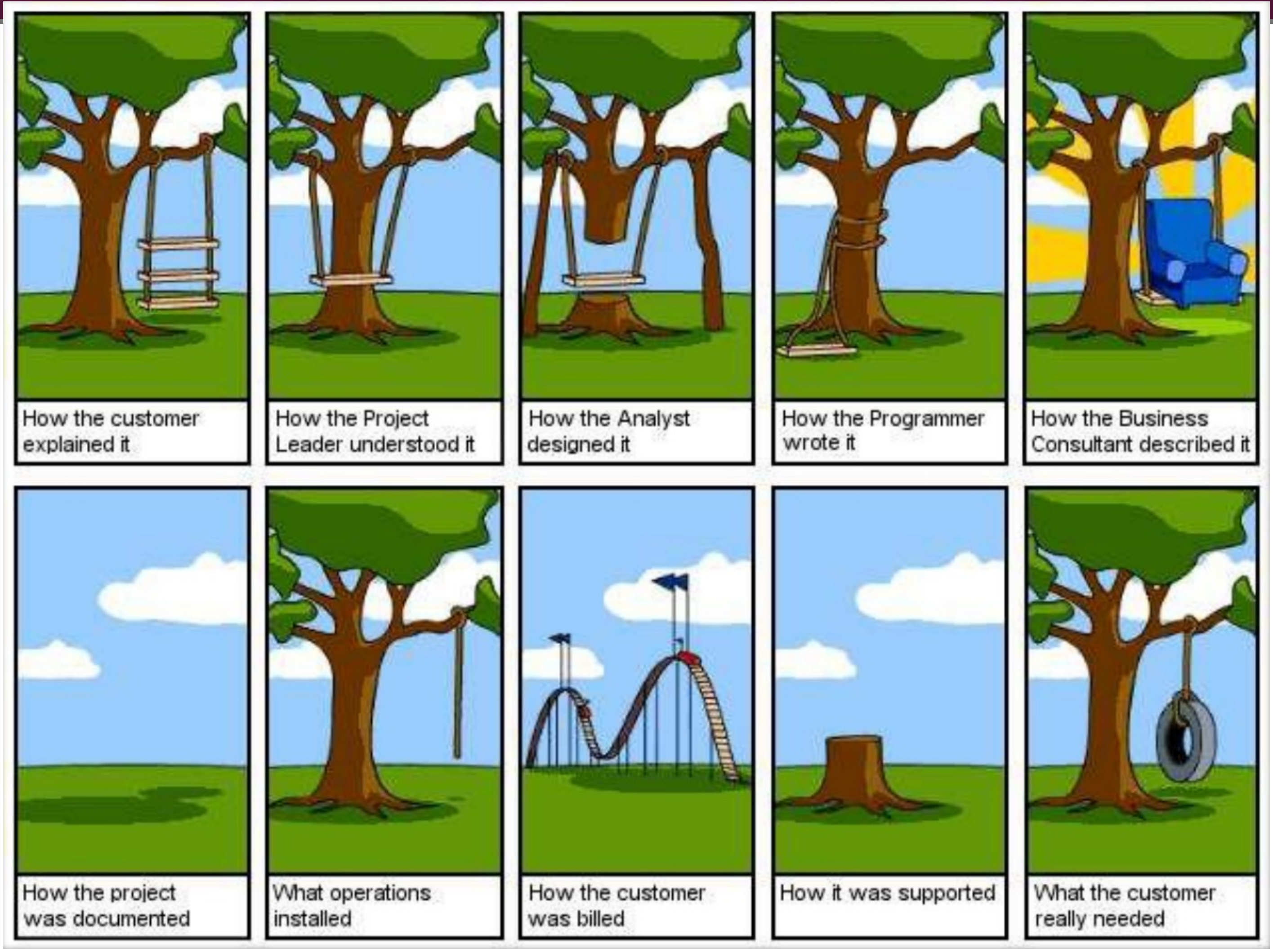


WHY IS SOFTWARE DEVELOPMENT DIFFICULT? (CONT.)

- Personnel characteristics
  - Ability
  - Prior experience
  - Communication skills
  - Team cooperation
  - Training

- Management issues
  - Cost estimation
  - Scheduling
  - Resource allocation
  - Quality assurance
  - Version control
  - Contracts

MAJOR PROBLEMS IN SOFTWARE DEVELOPMENTS


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

SOFTWARE MYTHS

- **Management myths**
  - *Add more programmers if behind the schedule.*
  - *My people have state-of-the-art software development tools, after all, we buy them the newest computers.*
  - *If I decide to outsource the software project to a third party, I can just relax and let that firm build it.*

SOFTWARE MYTHS

Customer myths

- *A general description of objectives enough to start coding.*
- *Project requirements continually change, but change can be easily accommodated because software is flexible.*

SOFTWARE MYTHS

Practitioner myths

- *Once we write the program and get it to work, our job is done.*
- *Until I get the program "running" I have no way of assessing its quality.*
- *The only deliverable work product for a successful project is the working program.*
- *Software engineering will make us create voluminous and unnecessary documentation and will invariably slow us down.*

HAVE A GOO DAY!