Introduction to Software Engineering

Mojtaba Shahin

Week #1: Lecture #1

Topics

- What is Software Engineering and why it is important
- Software process activities
- Ethical and professional issues that are important for software engineers

What is software?

Definition 1: Computer programs and **associated documentation**. Software products may be developed for a particular customer or may be developed for a general market.

Definition 2: Software is: (1) **instructions** (computer programs) that when executed provide desired features, function, and performance; (2) **data structures** that enable the programs to adequately manipulate information and (3) **documentation** that describes the operation and use of the programs.

Software products

- **Generic products**
  - Stand-alone systems that are developed by a development organization and sold to any customer who wishes to buy them.
  - Examples – apps for mobile devices, PC software such as graphics programs, project management tools; CAD software; software for specific markets such as appointments systems for dentists.
- **Customized products**
  - Software that is commissioned by a specific customer to meet their own needs.
  - Examples – embedded control systems, air traffic control software, traffic monitoring systems.

Product specification

- **Generic products**
  - The specification of what the software should do is owned by the software developer and decisions on software change are made by the developer.
- **Customized products**
  - The specification of what the software should do is owned by the customer for the software and they make decisions on software changes that are required.

What are the attributes of good software?

- Good software should deliver the required functionality and performance to the user and should be maintainable, dependable, efficient, and usable

Product characteristic	Description
Maintainability	Software should be written in such a way so that it can evolve to meet the changing needs of customers. This is a critical attribute because software change is an inevitable requirement of a changing business environment.
Dependability	Software dependability includes a range of characteristics including reliability, security and safety. Dependable software should not cause physical or economic damage in the event of system failure. Malicious users should not be able to access or damage the system.
Efficiency	Software should not make wasteful use of system resources such as memory and processor cycles. Efficiency therefore includes responsiveness, processing time, memory utilisation, etc.
Acceptability	Software must be acceptable to the type of users for which it is designed. This means that it must be understandable, usable and compatible with other systems that they use.

Software Engineering Definitions

- Software engineering is an **engineering discipline** that is concerned with **all aspects of software production** from the early stages of system specification through to maintaining the system after it has gone into use.
- Engineering discipline
  - –Using appropriate theories and methods to solve problems bearing in mind organizational and financial constraints.
- All aspects of software production
  - –Not just technical process of development. Also project management and the development of tools, methods etc. to support software production.

Other definitions

- Software engineering is the establishment and use of **sound engineering principles** in order to obtain **economically** software that is **reliable** and **works efficiently** on **real machines**.
- The IEEE definition:
  - Software Engineering:
    - (1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.
    - (2) The study of approaches as in (1).

Importance of software engineering

- More and more, individuals and society rely on advanced software systems. We need to be able to produce **reliable** and **trustworthy** systems **economically** and **quickly**.
- It is usually **cheaper**, in the long run, to use software engineering methods and techniques for software systems rather than just write the programs as if it was a personal programming project. For most types of system, the majority of costs are the costs of changing the software after it has gone into use.

Software process activities (more details on Week 2)

- **Software process:** The systematic approach that is used in software engineering is called a software process
  - **Software specification**, where customers and engineers define the software that is to be produced and the constraints on its operation.
  - **Software development**, where the software is designed and programmed.
  - **Software validation**, where the software is checked to ensure that it is what the customer requires.
  - **Software evolution**, where the software is modified to reflect changing customer and market requirements.

Software process activities (more details on Week 2)

- Different types of systems need different development processes
  - For example, **real-time software in an aircraft** has to be completely specified before development begins.
  - In **e-commerce systems**, the specification and the program are usually developed together.
- Consequently, these generic activities may be organized in different ways and described at different levels of detail, depending on the type of software being developed.

Software engineering diversity

- There are many different types of software systems and there is no universal set of software techniques that is applicable to all of these.
- The selection of software engineering methods and tools depends on these three factors:
  - the type of application being developed [the most important factor],
  - the requirements of the customer and
  - the background of the development team.
  - And more

Software engineering fundamentals

- Some fundamental principles apply to all types of software system, irrespective of the development techniques used:
  - Systems should be developed using a managed and understood development process. Of course, different processes are used for different types of software.
  - Dependability and performance are important for all types of system.
  - Understanding and managing the software specification and requirements (what the software should do) are important.
  - Where appropriate, you should reuse software that has already been developed rather than write new software.

Software engineering ethics

- Software engineering involves wider responsibilities than simply the application of technical skills.
- Software engineers must behave in an honest and ethically responsible way if they are to be respected as professionals.
- Ethical behaviour is more than simply upholding the law but involves following a set of principles that are morally correct.

Issues of professional responsibility

- **Confidentiality**
  - Engineers should normally respect the confidentiality of their employers or clients irrespective of whether or not a formal confidentiality agreement has been signed.
- **Competence**
  - Engineers should not misrepresent their level of competence. They should not knowingly accept work which is beyond their competence.

Issues of professional responsibility

- **Computer misuse**
  - Software engineers should not use their technical skills to misuse other people's computers. Computer misuse ranges from relatively trivial (game playing on an employer's machine, say) to extremely serious (dissemination of viruses).

Software Engineering / Computer Science / System Engineering

- **Computer Science vs. Software Engineering**
  - Computer science focuses on theory and fundamentals
  - Software engineering is concerned with the practical problems of developing and delivering useful software.
- **System Engineering vs. Software Engineering**
  - System engineering is concerned with all aspects of computer-based systems development including hardware, software and process engineering.
  - Software engineering is part of this more general process.

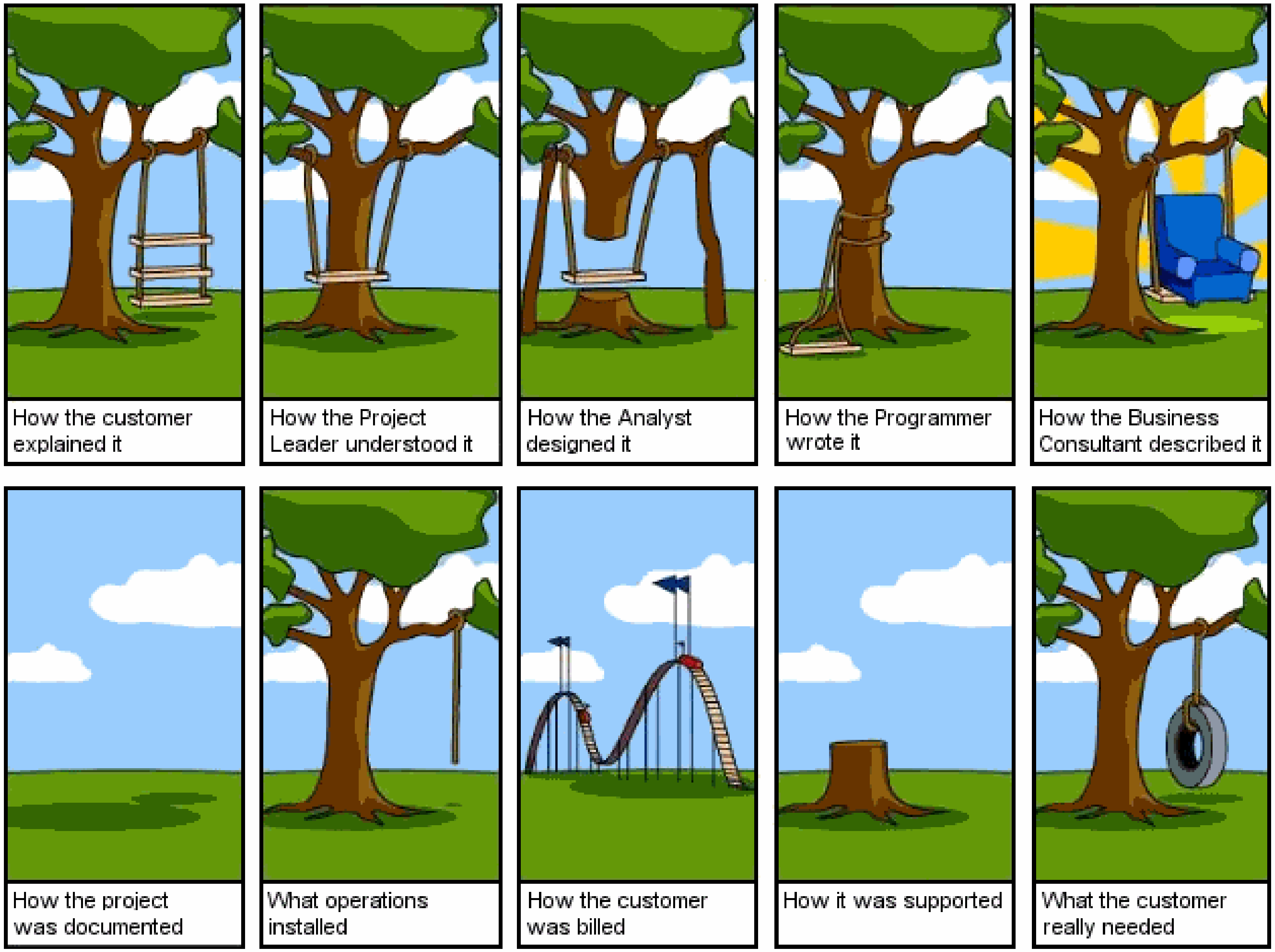
How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Image Source: https://lansa.com/blog/rapid-application-development/ambiguity-in-requirements#

Key points

- Software engineering is an engineering discipline that is concerned with all aspects of software production.
- Essential software product attributes are maintainability, dependability and security, efficiency and acceptability.
- The high-level activities of specification, development, validation and evolution are part of all software processes.
- The fundamental notions of software engineering are universally applicable to all types of system development.

Key points

- There are many different types of systems, and each requires appropriate software engineering tools and techniques for their development.
- The fundamental ideas of software engineering are applicable to all types of software system.
- Software engineers have responsibilities to the engineering profession and society. They should not simply be concerned with technical issues.

Disclaimer

- Slides come from two main sources:
  - Ian Sommerville developed slides; author of Software Engineering textbook.
    - https://iansommerville.com/software-engineering-book/slides/
  - Roger S. Pressman and Bruce R. Maxim developed slides, the authors of Software Engineering: A Practitioner's Approach textbook

References

- Ian Sommerville, Software Engineering, 10th Edition, 2015.
- Software Engineering: A Practitioner's Approach, 8/e (McGraw-Hill 2014). Slides copyright 2014 by Roger Pressman.

Thanks!

Mojtaba Shahin

mojtaba.shahin@rmit.edu.au