Topic 1

Software and Software Engineering

Contents

- The nature of software
- Software Engineering
- Software Projects

Differences?

Software

- Software is intangible
  - Hard to understand development effort
- Software is easy to reproduce
  - Cost is in its development
- The industry is labour intensive
  - Difficult to automate

Software

- Can be hacked together
  - Quality problems can be hard to notice
  - Low scalability
- Software is (too?) easy to modify
  - People make changes without full understanding
- Software does not wear out
  - It can deteriorate
    - by having its design changed
    - may become unusable as systems evolve

Software

- Demand for software is high
- Time is an important factor
- Increasingly complex
- Demand for high quality

Software Quality

- Usability
  - Can be learned quickly and facilitates users needs
- Efficiency
  - Low waste of resources like CPU time, memory and power use
- Compatibility
  - Works with a variety of systems and platforms
- Reliability
  - Fault-tolerant
  - Handles stress and unexpected conditions well

Software Quality

- Maintainability
  - Can be changed or extended with minimal effort
- Security
  - Protects sensitive data/systems
  - Few vulnerabilities
- Reusability
  - Parts of the code can be reused elsewhere

Types of Software

- Custom
  - For a specific customer
  - Usually developed in-house
- Generic
  - Many distribution types including:
    - COTS (commercial-off-the-shelf)
    - Free-to-download
    - Freeware
    - Open-source
- Embedded
  - Built into hardware
  - Difficult to change

Types of Software

- Real-time software
  - e.g. control and monitoring systems
  - Must react immediately
  - Safety often a concern
  - Soft vs. hard real-time
- Data processing software
  - Used to store data and/or perform analysis
    - e.g. to run businesses
  - Accuracy and security of data are key

Some software has both aspects

Software Engineering

Software Engineering

- Coined in 1968
- People realized the principles of engineering should be applied to software development

Engineers design artifacts following accepted best practices which involve the application of science, math and economics

- Engineering is a licensed profession
  - to protect the public
  - Ethical practice is a key tenet of the profession
  - Personal responsibility for the work taken

Software Engineering

“The process of solving customers’ problems by the systematic development and evolution of large, high-quality software systems within cost, time and other constraints”

- Textbook definition

Solving customers' problems

- This is an obvious goal of software engineering
- Software engineers must identify and understand the problem
- May be difficult to elucidate

Systematic development

- An engineering process involves applying well understood techniques in an organized and disciplined way
- Many well-accepted practices have been formally standardized (e.g. IEEE, ISO)
- Most development work is evolution

Large, high quality software systems

- Software engineering techniques are needed
  - large systems cannot be completely understood by one person
  - Teamwork and co-ordination are required

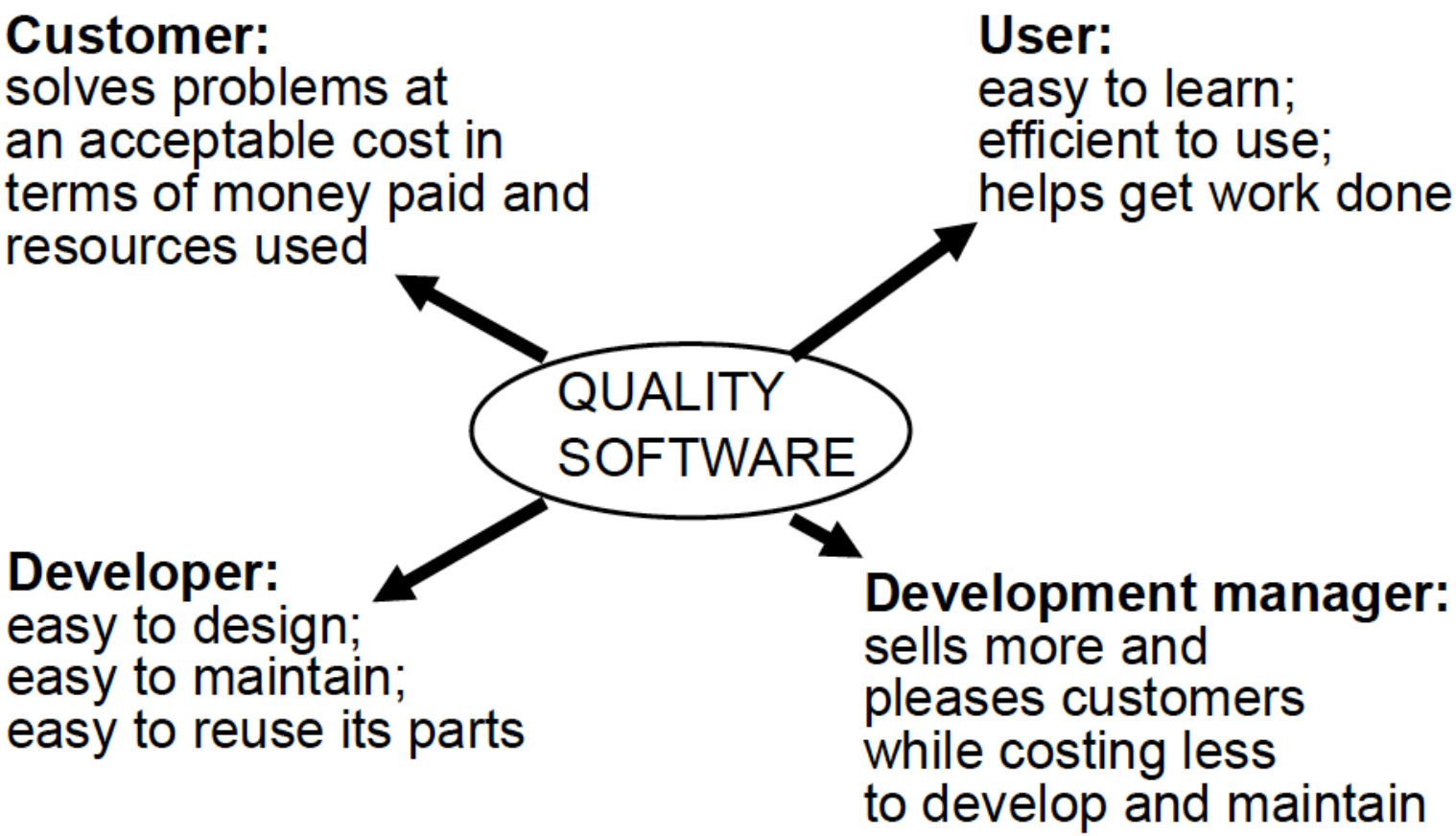
- The end-product that is produced must be of sufficient quality

Stakeholders in Software Projects

- Users
  - Use the software
- Customers / Clients
  - Pay for the software to be developed
- Software Developers
- Development managers

One person can perform multiple roles

Stakeholder interests



For each of the following systems, which attributes of quality would be most and least important? Justify your answer.

- a web-based banking system, enabling the user to do all aspects of banking online

- an air traffic control and management system

- a program that will enable users to view digital images or movies stored in all known formats

- a system to manage the work schedule of nurses that respects all the constraints and regulations in force at a particular hospital

Web Banking System

Most Important:

- Security
  - private banking information
  - people want to know their transactions are going to the right place
- Reliability
  - For everyone involved

Web Banking System

Least Important:

- Efficiency
  - Want it done - does not matter if it takes a while
- Reusability
  - Maybe want to keep code in-house

Arguably both:

- Compatibility
  - Not Important: typically can just use website
    - also banking hasn't changed much under the hood
  - BUT
    - users may value being able to use App on their new device
    - new transaction types come out like email transfers

Air Traffic Control & Management System

Most Important:

- Efficiency
  - Is a real-time demand, must work quickly
- Security
  - Do not want it to be subverted somehow (ex: hackers)
- Reliability
  - Want information to be stored – display correctly so planes do not crash or get put on the same gate.
  - Correctness (more on this later)

Air Traffic Control & Management System

Least Important:

- Compatibility
  - Installed on few types of system
- Reusability
  - Highly specialized system for specific context

Arguably both:

- Usability (Hard to argue)
  - Not important:
    - Can train users who are skilled
  - Important
    - Needs to be time-efficient and not infuriate controllers

Digital Video Streaming Service

Most Important:

- Efficiency
  - File sizes are large
  - Soft real-time system
- Reliability
  - Shouldn’t corrupt movie files or images, but error tolerance is rel. high
- Maintainability
  - Should be easy to add new types of media/formats
- Compatibility
  - Should work on a variety of systems and

Digital Movie/Image Viewing Program

Least Important

- Security
  - It is just viewing a file, not storing/editing
  - Basic authentication is probably sufficient
- Reusability
  - Parts of the codebase are unlikely to be useful in other types of applications

http://arstechnica.com/security/2014/12/sloppy-security-hygiene-made-sony-pictures-ripe-for-hacking/

Schedule System for Nurses in Hospitals

- Most Important:
- Reliable
  - Cannot have people missing/unaware of their shift
  - This is a matter of public health and safety
- Usability
  - Straightforward to use quickly
  - Nurses, while trained, have many other duties
- Maintainability
  - Want to be able to modify in case new regulations are made

Schedule System for Nurses in Hospitals

- Least Important:
- Compatibility
  - Only on a few systems, if not just one
  - May be a matter of government funding/policy, so they would be similar system across multiple hospitals

Internal Software Quality

- Internal quality also important
  - Characterize aspects of the design of the software
    - The amount of commenting of the code (SLOC)
    - The complexity of the code
    - The coupling of the code
  - Also affects external quality attributes

Quality timescales

- Short term
  - Does the software meet the customer's immediate needs?
  - Is it sufficiently efficient for the volume of data we have today?
- Long term
  - Maintainability
  - Continually meeting customer's future needs

Discussion

Can you think of an application you've used that you think was probably engineered poorly?

What are your biggest software gripes?

Software Projects

Always code as if the guy who ends up maintaining your code will be a violent psychopath who knows where you live.

- John Woods

https://groups.google.com/forum/#!msg/comp.lang.c++/rYCO5yn4lXw/olTtSkZOtoUJ

Software Project Activities

- Requirements and specification
- Design
- Modeling
- Programming
- Quality assurance
- Deployment
- Management of the process

Software Project Activities

Requirements and specification

- Domain analysis
- Defining the problem
- Requirements gathering
- Requirements analysis
- Requirements specification

Software Project Activities

Design

- Deciding how the requirements should be implemented, using the available technology
  - Systems engineering
    - Deciding what should be in hardware and what in software
  - Software architecture
    - Dividing the system into subsystems and deciding how the subsystems will interact
  - Detailed design of the internals of a subsystem
  - User interface design
  - Determining how data will be stored

Systems, UI, Data

Software Project Activities

Modeling

- Creating representations of the domain or the software
- Use case modeling
- Structural modeling
- Dynamic and behavioural modeling

Programming

Quality assurance

- Reviews and inspections
- Testing

Deployment

Management of the process

Legacy Systems

- Most projects are evolutionary or maintenance projects, involving work on *legacy* systems
  - Corrective projects: fixing defects
  - Adaptive projects: changing the system in response to changes in
    - Operating system
    - Database
    - Rules and regulations
  - Enhancement projects: adding new features for users
  - Reengineering or perfective projects: changing the system internally so it is more maintainable
- Very few projects, in industry, start from scratch

Green-Field Projects

- New developments
- Smaller but increasing number of projects
- Your team project for CS2212

Project Extension and Update

- Building on an existing project or framework
  - Adds new features in a novel setting
  - LOTS of this in game development

- Benefits from software reuse
  - Increases reliability (theoretically)
  - Lowers labour cost
  - Existing software is already documented

Your Team Project Type

- Similar to Green-Field or Startup

- Software development phases will be extremely rushed but are critical to follow.

Your Team Project Type

- Before demonstrating a product
  - Clear statement of purpose, user benefits
    - What problem does this solve for the user?
    - Where does this product fit into the market or target field?
    - How will users interact with the software?
  - Well documented information about design and implementation details
    - How was the prototype developed?
    - Using what systems and frameworks?
    - What will additional features look like? How will they work?
  - A polished, functional, TESTED prototype
    - Be confident that what you demonstrate to potential stakeholders will WORK

Why does the process matter?

Especially as a team or startup, why bother being so formal about the process?

- New stakeholders / investors / recruited team members need to be confident that you
  - Understand your market / field
  - Understand the dev process
  - Can deliver WORKING software
  - Have the capacity to grow your team

More About Projects

- Check course site for details about Stage 0
  - Meant to be open-ended, I want to see what you come up with
- Submit Stage 0 by email to TA and me – not via OWL
- Your project proposal should include features that cover the Key Technical Features mentioned last time
  - Check course site for final specification from last year's project
  - The project can be wildly different – but I want every team to incorporate key technical features (APIs, authentication, data analysis, database, graphical interface, user-focused)