Software Design & Modelling 1

# What is 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**
* The goal is to solve customers’ problems
* Sometimes the solution is to buy, not to build
* Software engineers must communicate effectively to identify and understand the problem

# Systematic Development and Evolution:

* An engineering process involves applying well understood techniques in a organized and disciplined way
* Many well-accepted practices have been formally standardized
* e.g. by the IEEE or ISO
* Most development work is evolution
* Teamwork and co-ordination are required
* Key challenge: Dividing up the work and ensuring that the parts of the system work properly together
* The end-product must be of sufficient quality

# Cost, Time, and Other Constraints:

* Finite resources
* The benefit must outweigh the cost
* Others are competing to do the job cheaper and faster
* Inaccurate estimates of cost and time can cause many project failures

A diagram of different colored circles

Description automatically generated

# Stakeholders in Software Engineering:

A diagram of a company

Description automatically generated

# Software Quality:

* Usability = Users can learn it and fast and get their job done easily
* Efficiency = It does not waste resources, such as CPU time and memory
* Reliability = It does what it is required to do without failing
* Maintainability = It can be easily modified
* Reusability = Its parts can be used in other projects

## Conflicts and Objectives:

* **The different qualities can conflict**
* Increasing efficiency can reduce maintainability or reusability
* Increasing usability can reduce efficiency
* **Setting objectives for quality is a key engineering activity**
* You design to meet the objectives
* Avoids ‘over-engineering’ which wastes money
* **Optimizing is also sometimes necessary**
* E.g. obtain the highest possible reliability using a fixed budget

# Software Engineering Projects:

* **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

# Activities Common to Software Projects:

A diagram with text and icons

Description automatically generated with medium confidence

# Software Design:

* **Systems Engineering**: Hardware vs. software decisions
* **Architecture**: Define subsystems and interactions
* **Detailed Design**: Design classes, algorithms, etc.
* **UI Design**: Define user interaction and system look
* **Data Storage**: Database and file storage decisions

# Software Modelling:

* **A model is an abstraction of reality for a given purpose (cannot represent all aspects of reality)**
* **Use Case Modeling**: Represents user actions
* **Structural Modeling**: Represents classes and objects
* **Dynamic/Behavioral Modeling**:Represents system states, activities, and interactions
* **Visual Modeling**: Use diagrams and semi-formal languages (e.g., UML)

# Version Control Systems (VSC):

* **A system (e.g., Git or SVN) that tracks changes to files over time**
* Manage code changes and maintain version history
* Collaborate with others (internally or externally)
* **Key Features**
* Tracking: what, when, and who
* Reverting: Undo changes or revert to previous versions
* Branching: Create separate paths for new features or bug fixes

# Git:

* A *distributed* VCS, where each developer has a complete copy of the repository
* Git keeps snapshots of files over time
* Each snapshot is a picture of your entire project
* Every change is stored in your local machine
* Git adds data to the repository using snapshots (commits)

A diagram of a checklist

Description automatically generated with medium confidence

# Branching in Git:

* In Git, project development is seen as a list of commits
* When development track splits, a branch is created
* In Git, branches are pointers to commits
* The head pointer is the pointer to your current branch

A diagram of a family tree

Description automatically generated

# Merging in Git:

* When work is done on a certain branch, it can be merged back to the master branch
* During merging, you may have to resolve any arising conflicts
* You can learn more about git from <https://git-scm.com>

A diagram of a flowchart

Description automatically generated

# GitHub:

* A *web-based* platform for hosting Git repositories
* Centralized repository hosting
* Collaboration: Work with others through pull requests and code reviews
* Issue Tracking: Manage bugs and feature requests
* CI/CD Integration: Automate testing and deployment
* Access control and project management

Website: [www.github.com](http://www.github.com)