Introduction to Software Engineering

John D. McGregor
Clemson University
Dept.of Computer Science
johnmc@cs.clemson.edu

Science & Engineering

- Science
  - basic knowledge about a domain
  - physics
  - action/reaction
- Engineering
  - solving problems by applying basic knowledge
  - aeronautical engineering
  - lift

Computer Science and Software Engineering

- Computer science
  - basic knowledge about computation
  - what can be computed using a Turing machine
- Software engineering
  - solving computational problems by applying basic knowledge
  - identifying specific needs that can be solved computationally

Engineers

- Apply basic knowledge
- Develop and adhere to standards
- Accept responsibility for their decisions and actions
- Cooperate with others to solve problems

Software Engineers

- Apply basic knowledge
  - Halstead's software science
- Develop and adhere to standards
  - IEEE-### Test Plan Standard
- Accept responsibility for their decisions
  - Sign off for architectural decisions
- Cooperate with others to solve problems
  - work in large groups with wide variety of skills

People in software

- Computer scientists
  - relatively small group
- Software engineers
  - rapidly growing profession
- Programmers
  - legacy of code writers
- Hackers/amateurs
  - anyone can buy a compiler

Engineering a Product

- Solve the problem correctly.
- Solve it effectively, according to existing standards
- Solve it completely, including error conditions
- Solve it efficiently, using the least amount of resources while still achieving acceptable levels of quality.

7

A Quality Product

- "know it when we see it"
- "fit for purpose"
- "does everything it is supposed to and nothing it is not supposed to"
- "inner beauty" - Volvo or Cadillac
- "whatever the traffic will bear"

8

Fig 1.5 - McCall's Quality Model

9

System Context

- Multiple applications may run at the same time
- Applications may share data via a clipboard or common file format
- Users range from novice to expert
- Applications are an integral part of the hardware or extraneous

10

Types of Systems

- Embedded real-time
  - toaster control

- GUI-based interactive
  - word processor
- 3/4-tiered, web-based e-commerce system
  - point of sale application
- Batch processing system
  - check processing

11

Development Projects

- 2 to 2000 people
- 1 month to 5 years or more
- thousands to millions of dollars
- in-house or out-sourced

12

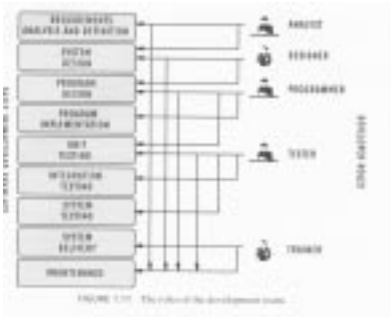
Process

- Question - How do you get 200 people to work together to write one program?
- Answer - Define a series of steps that accomplish the task at hand - a process

Process definitions

- Phase
  - a step in accomplishing the objective of the process
  - analysis is the phase of understanding the problem being solved
- Role
  - a work assignment within a process
  - analyst is the role that is responsible for performing analysis

Fig 1.11 - Process steps



Iterative, Incremental Process

- Iterative
  - repeat some steps in the process
  - because by doing some design we learn where the analysis needs to be clarified
- Incremental
  - partition a problem into a set of smaller pieces
  - because we can't understand large problems to the level of detail needed

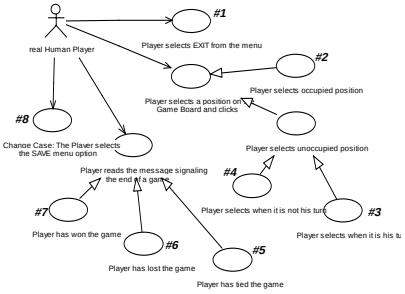
Software Engineering Directions

- Time to market shortened
- Software more expensive than hardware
- Developers can be sloppy because of the computing power
- World-wide networking
- Object technology
- Graphical user interface
- Failure of waterfall model

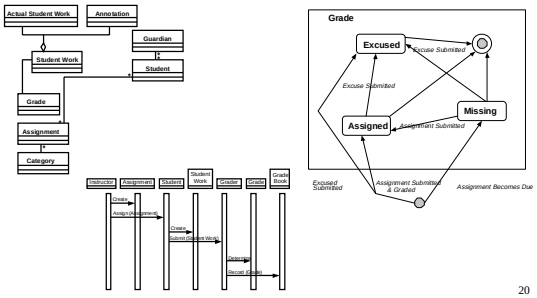
Techniques - 1

- Abstraction
  - ignore details to understand essentials
- A & D methods and notations
  - unified modeling language is a standard
- Prototyping
  - implement the essentials to identify problems
- Software architecture
  - structure is a fundamental, initial characteristic

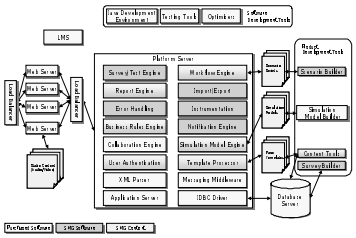
UML - 1



UML - 2



Architectural level view



Techniques - 2

- ▲ Software process
  - organizes people
- ▲ Reuse
  - reduces cost but is a difficult cultural shift
- ▲ Measurement
  - provides a rational basis for decisions
- ▲ Tools and environments
  - makes methods more repeatable

Outline of process phase

- ▲ **1. Phase Name**
  - **1.1. Description**
  - **1.2. Responsibility**.
  - **1.3. Input**
  - **1.4. Entry criteria**
  - **1.5. Activities**
  - **1.6. Output**
  - **1.7. Exit criteria**
  - **1.8. Metrics**

Conclusion

- ▲ Computer science and software engineering present two distinct perspectives
- ▲ This semester we will investigate the software engineering perspective
- ▲ We will work together as a team just as a software development project team works together