Introduction to Software Engineering

Idaho State University | Computer Science

Dr. Isaac Griffith

CS 3321
Department of Computer Science
Idaho State University

ROAR

Outcomes

After today's lecture you will:

- Understand the Course Syllabus and Course Outcomes
- Understand what software engineering is
- Understand the nature of software and why we need to engineer it
- Begin to shift your thinking from programming to software engineering

Syllabus Overview

CS 3321

ROAR

Prerequisite Overview

The Prerequisite for this course is CS 2263 and not CS/INFO 1182. Therefore I expect you to know and understand the following:

- From CS 2263
  - Git and the use of Git Flow
  - Dependency Management and Build Automation
  - Fundamental OO Design Principles and Practices
  - OO Design Patterns (GoF) and their implementation
  - Practices of Good Programming and Defensive Coding
  - Fundamentals of UML
  - Principles and Practices of TDD
  - Principles and Practices of CI/CD
  - Teamwork and having completed a large project

Prerequisite Overview

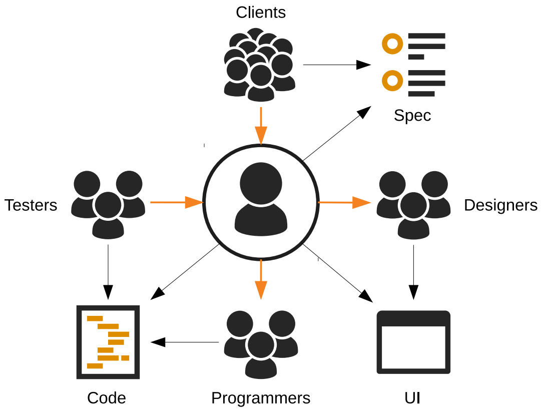
- From CS 2235 (prereq to 2263)
  - Java
  - Basic Data Structures and their Implementation
  - Use of OO in a larger context
  - Basic algorithm implementation
  - Basic algorithm design strategies
  - Ability to complete moderately complex programs
- From CS 1181 (prereq to 2235)
  - Python
  - Basics of programming
  - Basics of OO
  - Ability to complete simple programs

Software Engineering

CS 3321

ROAR

Why Software Engineering?



- Interact with clients to determine their system requirements
- Translate user requirements into technical specifications
- Interact with designers to convey the possible interface of the software
- Interact/guide the coders/developers to keep track of system development
- Perform system testing with sample/live data with the help of testers
- Implement the new system
- Prepare high quality documentation

Defining Software

What is software?

Defining Software

What is software?

1. instructions (computer programs) that when executed provide desired features, function, and performance;

Defining Software

What is software?

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

Defining Software

What is software?

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. descriptive information in both hard copy and virtual forms that describes the operation and use of the programs

Software Deterioration

85%

Software Project Failure Rate

UK, USA, and Norway Surveys

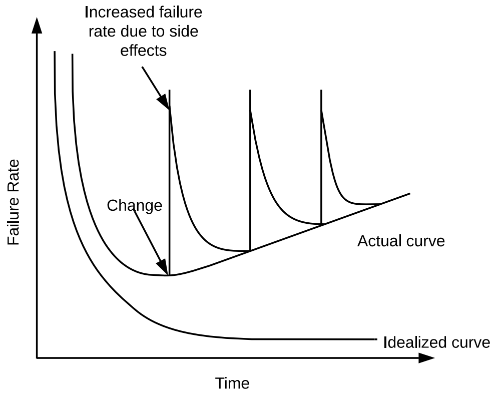
50%	40%
Total Failures	Partial Failures

Standish Group (Corporate Projects) – 1994

31%	53%	180%
Canceled	Challenged	Average Cost overrun

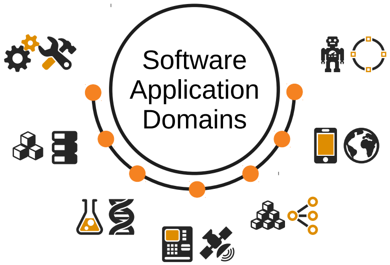
Standish Group 2007

46%	13%
Cost or Time Overruns	Outright Failures



ROAR

Software Domains and Challenges



- Open-world computing
  - Creating software to allow machines of all sizes to communicate with each other across vast networks
- Netsourcing
  - Architect simple and sophisticated applications that benefit targeted end-user markets worldwide
- Open Source
  - Distributing source code for computing applications so customers can make local modifications easily and reliably

Legacy Software

Legacy Software Systems:

Systems that were developed decades ago and have been continually modified to meet changes in business requirements and computing platforms. The proliferation of such systems is causing headaches for large organizations who find them costly to maintain and risky to evolve.

Reasons Legacy Systems Evolve:

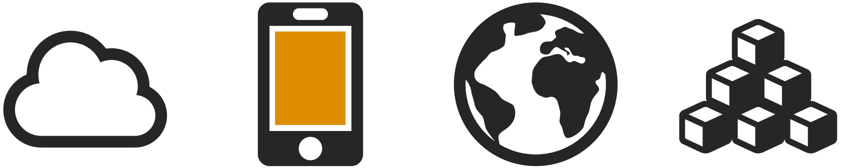
- Meet the needs of new computing environments/technologies
- To implement new business requirements
- To inter-operate with other more modern systems or databases
- To become viable within an evolving computing environment

Changing Nature of Software

The goal of modern software engineering is to “devise methodologies that are founded on the notion of evolution;”

Software systems continually change and all must interoperate and cooperate with each other.

Four broad categories are evolving to dominate:



Software Engineering

Software Engineering (IEEE):

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)

- Software engineering encompasses a process, a collection of methods, and an array of tools that allow professionals to build high quality software.
- Software engineers view computer software, as being made up of the programs, documents, and data required to design and build the system.
- Software users are only concerned with whether or not software products meet their expectations and make their tasks easier to complete.

Software Engineering Realities

- Problem should be understood before software solution is developed
- Design is a pivotal activity
- Software should exhibit high quality
- Software should be maintainable

Software Engineering

- The economies of ALL developed nations are dependent on software.
- More and more systems are software controlled
- Software engineering is concerned with theories, methods and tools for professional software development.
- Expenditure on software represents a significant fraction of GNP in all developed countries.

Software Costs

- Software costs often dominate computer system costs. The costs of software on a PC are often greater than the hardware cost.
- Software costs more to maintain than it does to develop. For systems with a long life, maintenance costs may be several times development costs.
- Software engineering is concerned with cost-effective software development.

Software Project Failure

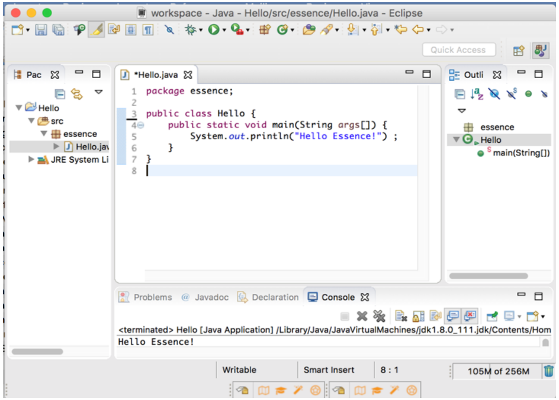
- *Increasing system complexity*
  - As new software engineering techniques help us to build larger, more complex systems, the demands change. Systems have to be built and delivered more quickly; larger, even more complex systems are required; systems have to have new capabilities that were previously thought to be impossible.
- *Failure to use software engineering methods*
  - It is fairly easy to write computer programs without using software engineering methods and techniques. Many companies have drifted into software development as their products and services have evolved. They do not use software engineering methods in their everyday work. Consequently, their software is often more expensive and less reliable than it should be.

Idaho State University Computer Science

From Programming to Software Engineering

- We assume you have Software Programming Experience

- You know programming languages, development environment, OS, DataBases, etc.
- You have developed some software projects as required by your previous classes



```
package essence;

public class Hello {
    public static void main(String args[]) {
        System.out.println("Hello Essence!") ;
    }
}
```

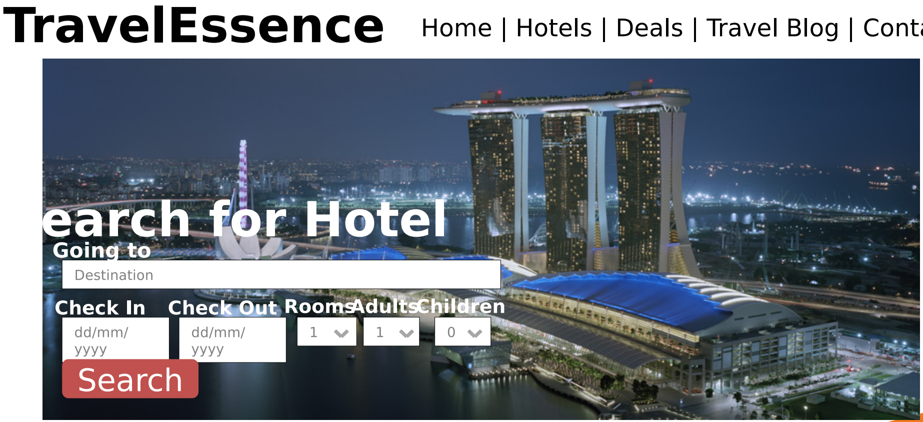
```
<terminated> Hello [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_111.jdk/Contents/Hom
Hello Essence!
```

- We are now starting a travel toward how to organize your approach to build any software system you are required to do.

ROAR

Example Project

Let’s assume you are just hired as the Software Engineer of a small company charged to develop a software system that is a web and mobile travel management and reservations system.

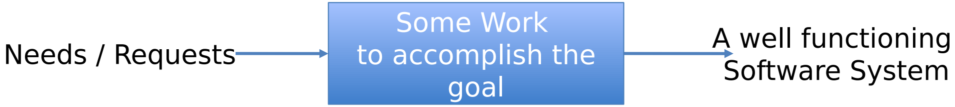


ROAR

Disclaimer

- Students, having spent most of their lives in school and not yet working in industry, tend to be more interested in the technologies related to software
  - The computer
  - Programming languages
  - Operating Systems
  - Algorithms
  - etc.
- and less interested in how to organize a team of people to develop software systems for a *customer*.
- The latter is the goal we have with this class

The Software Engineering Goal


Needs / Requests
Some Work to accomplish the goal
A well functioning Software System

- What is in the box?
- What is the work that we need to do in order to accomplish the goal?
  - …To have a well functioning Software System?

Other Questions

- What tasks do we need to perform?
- How do we ensure we will do a great job?
- How do we measure the quality of what we have done?
- How do we ensure that what we have done is really matching the initial needs and requests?
- How do we minimize the effort to accomplish the goal?
- What skills do we need?
- What tools do we need?
- How many people are needed?
- Which skills, profiles, experience?
- Etc.

For Next Time

- Review the Syllabus
- Review this Lecture
- Review Essentials Chapter 1
- Read Essentials Chapters 2 – 5
- Review Lecture 02
- Watch Lecture 02
- Come to Lecture

Are there any questions?

ROAR