Software Engineering

Professors M. Brian Blake and Iman Saleh-Moustafa

Course Overview

Class Philosophy

- Informal Class Environment
- 50% Lecture / 50% Hands on exercise each week
- Learn the material and your grades should reflect
- Teaching the course from a corporate perspective, preparation for the real world

Administration

- Joint Lecturers:
  - Dr. M. Brian Blake (305) 284-4154
  - Dr. Iman Saleh (305) 284-2658
- Office Location: Graduate School, Pick Hall, Brescia Avenue
- Teaching Assistants: Damian Clarke
  - (305) 284-4220 (d.clarke6@umiami.edu)
- Office Hours:
  - Blake T/Th 11:15am -12:15pm (located at Pick)
  - Saleh T/Th 12:15 pm – 1pm (located at Pick)
  - Clarke M/W 9-11am (or by appointment) (located 4th Floor Ungar)

Class Progression

- Will adhere to the syllabus
- Group projects (5 in a group)
  - Presentations at the end of the term
  - Ideas will be coming from outside companies problem sets
- About 4 to 5 homework assignments throughout the semester

Class Overview/Goals

- Understand how to work in a group to develop medium-scale software applications
- Become proficient at the industry-standard modeling language (Unified Modeling Language (UML))
- Understand the general software engineering life-cycle
- Gain an appreciation of solving an **open** problem that requires innovative software engineering practices
- Develop a product and take it "cradle to grave" through the full software engineering lifecycle and articulate the outcomes to an external customer
- Gain an appreciation of contemporary software engineering techniques such as design patterns, model-driven architecture, and service-oriented computing

Let's run through the Syllabus

Can we substitute a few classes with a night-time dinner class (corporate presentations)

Main Course Webpage: http://www.cs.miami.edu/~blake/SE.html

Document location: http://www.cs.miami.edu/~blake/SE/

Your Final Project Assignment

(yep, this early)

Night-time Class (Mon Jan 29))?? Syllabus Changes? Talk about Grading?

Next Class

Other Things that I forgot:
Cell Phones/ Laptops
Website, "Living Syllabus", "No Books" and Slide Location

Limitations of Non-engineered Software

Requirements

How would you start?

Software

Software Engineering

Professors M. Brian Blake and Iman Saleh-Moustafa

Lecture 1: Introduction to the Object-Oriented Software Engineering

Software Production has a Poor Track Record

Example: Space Shuttle Software

- Cost: $10 Billion, millions of dollars more than planned
- Time: 3 years late
- Quality: First launch of Columbia was cancelled because of a synchronization problem with the Shuttle's 5 onboard computers.
  - Error was traced back to a change made 2 years earlier when a programmer changed a delay factor in an interrupt handler from 50 to 80 milliseconds.
  - The likelihood of the error was small enough, that the error caused no harm during thousands of hours of testing.
- Substantial errors still exist.
  - Astronauts are supplied with a book of known software problems "Program Notes and Waivers".

- **Service Activation – Professor Blake, BellSouth Experience**
- **Case Management System – Professor Blake, FBI**
  - **Class seniors – Experience so far?**

Software Engineering

- Definitions:
  - 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. - IEEE Std 610-1990
  - Engineering is the systematic application of scientific knowledge in creating and building cost effective solutions to practical problems and service of mankind. Software engineering is that form of engineering that applies the principles of computer science and mathematics to achieving cost effective solutions to software problems - CMU/SEI-90-TR-003

Object Technology

- Many believe that object technology makes software quantifiable
- In a way, OOA/OOD engulfs all of our original definitions of Software engineering
- Later, we will discuss how object technology makes software systems quantifiable, in the opinions of some.

Can there really be Software Engineering ?

- Is software quantifiable as in other engineering fields ?
- Does current software development firms consider software engineering as a field ?
- In Microsoft's case, does it really work?
- Why is software engineering important ?

Background on Object Technology

- What is a Methodology ?
  - A process for organized production of systems and software using a collection of pre-defined techniques and notational conventions.
- What is an Object-Oriented Methodology?
  - A development approach that organizes a system as a collection of objects containing both data structure and behavior.

Decomposition

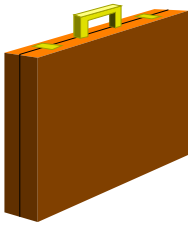
What is the least denominator, the smallest piece that a human body can be broken down into?

Is this the most practical piece to build upon?

Major Concepts of Object-Oriented Analysis and Design

- What is an Object?
  - An object has structure ~attributes
  - An object must be an entity ~ a thing that can have properties and not be a property itself.
  - An object has behavior
  - An object has unique identity
  - An object is generally stated as a noun
  - For example : Thermometer is an object, temperature is not an object it is a property (attribute) of the thermometer

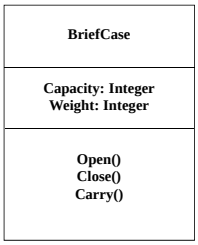
What is this Thing?



Exercise 1.1

- Identify the objects likely to be encountered in the following systems/domain:
  - A Convertible Car
  - An Airline
  - A Computer network

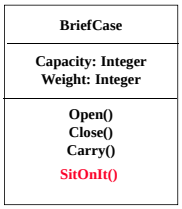
Modeling a Briefcase



OO Concepts

- What is a Class ?
  - A group of objects with similar properties (attributes), common behavior (operations), common relationships to other objects (associations), and common semantics.
- What is the difference between a Class and an Object ? Give an example.

A new Use for a Briefcase



Exercise 1.2 Class Interpretation

- What classes would you create for the following lists of objects?
  - 1. 747, Lear jet, twin engine plane, stealth bomber
  - 2. laser printer, dot matrix printer, ink jet printer, fax machine, photocopier
  - 3. Prodigy, Compuserve, America On-line, Erol
  - 4. Fog-light, headlight, blinker, brake light, back-up light, turn signal

Object-Oriented Concepts

(You will be Graded by these!!)

- What is Reuse/Reusability?
  - The sharing of common components within a single project and across multiple projects
- What is Generalization?
  - The relationship that organizes classes based on similarities and differences. Example?
- What is Inheritance?
  - The sharing of features (e.g., structure and behavior among classes related by generalization. Example ?

Patterns are used by many people

- Chess Master:
  - Openings
  - Middle games
  - End games
- Writer
  - Tragically Flawed Hero (Macbeth, Hamlet)
  - Romantic Novel
  - User Manual
- Architect
  - Office Building
  - Commercial Building
  - Private Home
- Software Engineer
  - Composite Pattern: A collection of objects needs to be treated like a single object
  - Adapter Pattern (Wrapper): Interface to an existing system
  - Bridge Pattern: Interface to an existing system, but allow it to be extensible

Reusability

- A good software design solves a specific problem but is general enough to address future problems (for example, changing requirements)
- Experts do not solve every problem from scratch
  - They reuse solutions that have worked for them in the past
- Goal for the software engineer:
  - Design the software to be reusable across application domains and designs
- How?
  - Use design patterns and frameworks whenever possible

OO Concepts

- What is Abstraction?
  - Generalizing such that design focus is on the inherent aspects of an entity and not those that are accidental of specialized. Example ?
  - Why is this important?
- What is Encapsulation?
  - Also information hiding, it consists of separating aspects of an object, which are accessible to other objects, from the internal implementation details of the object. Example ?
  - Why is this important?

Design Patterns and Frameworks

- Design Pattern:
  - A small set of classes that provide a template solution to a recurring design problem
  - Reusable design knowledge on a higher level than data structures (link lists, binary trees, etc)
- Framework:
  - A moderately large set of classes that collaborate to carry out a set of responsibilities in an application domain.
    - Examples: User Interface Builder
- Provide architectural guidance during the design phase
- Provide a foundation for software components industry

OO Concepts

- What is Sharing ?
  - Allows functionality and total modules to be implemented in various areas of the software system
- How does inheritance promote sharing ?
- What is Scalability ?
  - This is that property of a software system that allows other components to enter seamlessly.

My OO Philosophy

- Software design is not yet a engineering science, still subjective in some ways
- Believe in the basics
  - Scalability, Encapsulation, Reusability, Abstraction, Polymorphism
- Show me the above and you will do well in the class.
- << THE TOOLBOX PHILOSPHY>>

History of Selected OO Development Methodologies

- Booch
  - Originated for Ada development in the defense industry by Grady Booch
  - Heavier emphasis on design process
  - Extensive real-time and packing constructs
  - Comparatively complex notation
- Objectory
  - Originated in Swedish telecommunications market by Ivar Jacobsen
  - Popularized Use Cases
  - Heavy emphasis on textual description
  - Comparatively weak design phase
- Shlaer/Mellor
  - Originated in Berkeley Laboratories
  - Strong on information modeling
  - Real-time orientation
  - Discontinuities between analysis and design
  - More formal and less flexible
  - Less sophisticated to support complexities

Objected-Oriented Design Wrap-up

- Software Engineering
  - Is it engineering?
- Classes and Objects
- Object-Oriented Technology Concepts
  - Inheritance, Polymorphism, Generalization, Abstraction etc.
- My Philosophy - Think CONCEPTS!!

History of Selected OO Development Methodologies

- Object Modeling Technique (OMT)
  - Developed in early 1990's, highly accepted among the software environment
  - an object-oriented methodology originally developed at GE Research and Development Center. It covers the system development process from conceptualization phase through implementation based on there complementary views
  - Rumbaugh, Blaha, Premerlani, ***Eddy***, Lorensen
- OMT represented the first total solution
  - Method - OMT methodology
  - Automation - OMT Case Tool (Rational)
  - Maturity - Training, Mentoring, Consulting, System Integration

First real collaboration of Object Technologist, led to later unification

Software Engineering

Professor M. Brian Blake

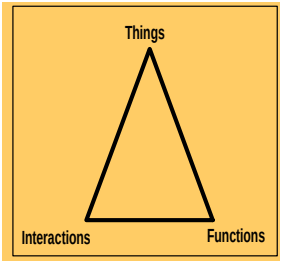
Lecture 2: The Software Engineering Lifecycle

History of Selected OO Development Methodologies

- Unified Modeling Language (UML)
  - Early development around 1996-97
  - First real publications by authors October, November and December of 1998
  - Newest Language, authors do not commit to a process of development (with reason)
  - Combines the best of OMT, Objectory, Booch and Schlaer Mellor
  - Rumbaugh, Booch, Jacobsen "Three Amigos"
- We will use the OMT (waterfall) process on UML notations
  - Great backbone for Object-oriented design
  - OMT is a "tried-and-true" methodology
  - To date, most designers use UML notation but still map development processes around OMT design process
  - UML fosters a hybrid development approach
  - We will attack OMT using mostly UML notations and semantics

The OMT Process

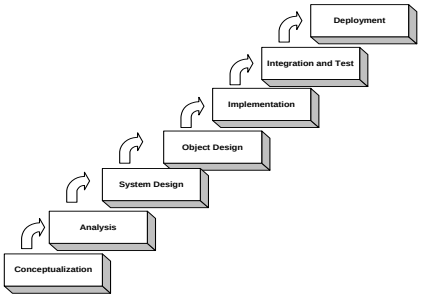
- Use Case
  - This view is shared by both OMT and UML.
  - It is the initiation step that provides the requirements for the system.
- Analysis Model
  - Object Model
    - What are the objects?
    - How are they related?
  - Dynamic Model
    - What are the object states?
    - What events cause the state to change?
  - Functional Model
    - What are the processes?



Three Complementary Views of an OMT System

Waterfall Lifecyle

Professor Blake's



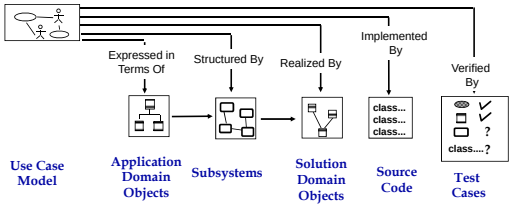
How do the models relate?

- Dynamic models shows the sequence of changes to the objects' states
- Functional model shows details of complex algorithms for operations on the objects
- *Basically, the object model shows the structure on which the other two models operate on.*

Software Lifecycle Activities

Bruegge (10 years later)

Requirements Elicitation	Analysis	System Design	Object Design	Implementation	Testing

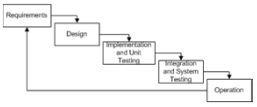


General Definition of Software Lifecycle

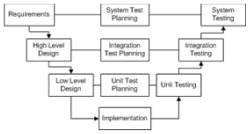
- Software lifecycle:
  - Set of activities and their relationships to each other to support the development of a software system
- Typical Lifecycle questions:
  - Which activities should I select for the software project?
  - What are the dependencies between activities?
  - How should I schedule the activities?

Variation of waterfall lifecycles

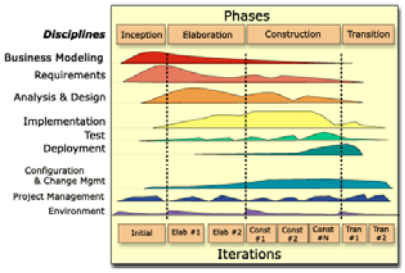
- Incremental Lifecycle
  - Full lifecycle, but over and over again



- V-Shaped
  - Promotes either parallel paths or significant depth in advance
  - Efficient for large application and big development teams



Rational Unified Process
IBM's Flavor of Waterfall



Exercise 1.1 Sample Objects

Convertible	Airline	Computer Network
Engine	Airplane	File
Chasis	Terminal	Protocol
Steering Wheel	Baggage Claim	Server
Brake	Schedule	Workstation
Accelerator	Ticket	Cable
Radio	Reservation	Port
Tire	Pilot	Printer
Auto Light	Flight Attendant	Disk
Windshield	Passenger	Process
Wiper	Flight Plan	Test Equipment
	Stock Holder	Access Priviledge
	Gate	

Spiral Model – Barry Boehm

- Identifying Characteristics
  - Run through the same cycle but at increasing degrees of detail
  - 6 month to 2 year iterations depending on product
  - Used this model within the Department of Justice

Exercise 1.2 Suggested Classes

- 1. Airplane
- 2. Printer - Things that Jam
- 3. On-line Service
- 4. Light, Auto Light, Motor Vehicle Light

- Attributes, Operations, Associations, etc. on Lecture 3 (THE OBJECT MODEL)

Agile Software Development

- Identifying Characteristics
  - Not really a process, but development approach
  - Absolutely requires close stakeholder interaction
  - Works best on the site of the customer or if you have a really knowledgeable Subject Matter Expert (SME)
  - A traditionalist's nightmare
    - Measuring risk is difficult.