Student Originated Software – Fall 2001

Object-Oriented Analysis & Design

Description

Often surprising to those whose main exposure to writing software has been with once-written, once-read, once-executed programs according to an academician's specification is how little time, comparatively speaking, is spent coding during the "real" software development process. Furthermore, most (and most egregious) errors in large-scale systems can be traced back, not to coding, but to requirements specification and design. Applying project management and systems engineering best practices within each phase of software development provides a disciplined approach to developing large-scale systems that, we hope, avoids such errors. *Student Originated Software* introduces software engineering principles by applying and integrating the following skills to software development: project management, feasibility analysis, requirements definition, preliminary and detailed design, programming, information architecture, database management, user interface design, and software testing and integration.

The main objective of this part of the program is to learn concepts of the software engineering process, to gain the ability to apply associated software development methods and to gain skill using modern tools. In addition to the case study workshop, exercises and assignments (see Case Study syllabus), each week there will be OOAD lectures and readings. There will also be a mid term and final exam.

We will use an object modeling language "Unified Modeling Language" (UML) to draw diagrams of object-oriented designs. We will also explore the Rational Rose modeling tool.

Expectations and Requirements:

To earn credit, students must attend weekly lectures, complete the reading assignments and perform satisfactorily on two exams.

Students will be evaluated on the basis of their understanding of the software development process and methods, as reflected in being able to explain key concepts through participation in discussions, pop quizzes, and exams. They will also be evaluated on the basis of their ability to implement these basic concepts through weekly case study assignments and a final project notebook.

Primary Texts:

Our primary texts are Martin Fowler's *UML Distilled* and Terry Quatrani's *Visual Modeling with Rational Rose 2000 and UML*

Typical Week:

Monday

11 – 12 (LIB 1316) – Lecture: covering new material
1 – 3 (LIB 1316) – Case Study Workshop: working example exercises
Previous week's Case Study Assignment returned and critiqued

Wednesday

Reading assignments due
11 – 12 (LIB 1316) – Lecture: covering new material

Thursday

2:00 PM Current week's Case Study assignment due.

Tentative Schedule:

Week	Mon	Workshop	Readings due by Wed am	Wed
1	9/24 Introduction		Fowl - Ch 1, 2 & 4; Quat - Ch 1,2 & 4	9/26 Problem Domain
2	10/1 Problem Domain		Fowl - Ch 3 & 9 (p 135-137); Quat - Ch 3 (p 21-29,32-38)	10/3 Use Cases
3	10/8 Use Cases		Quat - Ch 3 (p 29-32, 38-47) & Ch 5 (p 73-76, 84-85)	10/10 Use Case Realization
4	10/15 Use Case Realization		TBD	**10/17 MIDTERM**
5	10/22 Collaboration Model		Quat - Ch 6 (p77-82) & Ch 7	10/24 Design model
6	10/29 Design Model		Quat - Ch 9, Fowl - Ch 8 & TBD	10/31 Exception Handling
7	11/5 Testing		Quat - Ch 11, Fowl Ch 9 & 10	11/7 Implementation
8	11/13 Exception Handling	Lecture/Workshop on Tues		**11/14 FINAL**
9	11/26			11/28 Review of Final