Object-Oriented Analysis and Design

Kenneth M. Anderson

University of Colorado, Boulder

CSCI 4448/6448 — Lecture 1 — 08/28/2007

A bit about me...

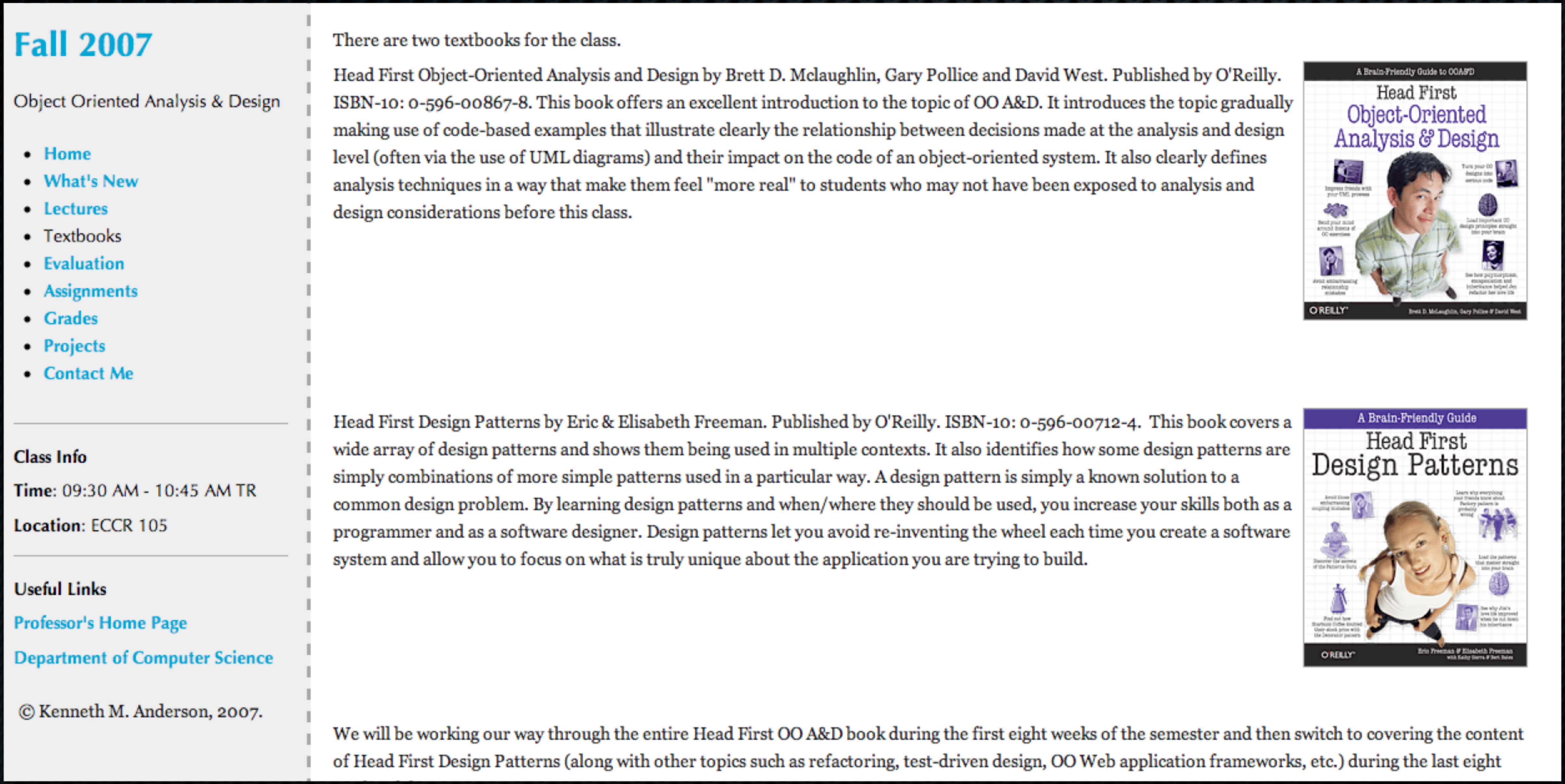
- Associate Professor
- At CU since July 1998
- Ph.D. at UC Irvine
- Research Interests
  - Software Engineering
  - Hypermedia / WWW

A little bit more…

- 19th Semester at CU
- 6th time teaching CSCI 4448/6448
- Software Development Experience
  - Approximately 16 systems, 30K-100K LOC each
  - Some industry experience with IBM & Unisys
  - Experience with academic/industry collaboration

Office Hours

- ECOT 822
- Monday: 10:30 AM — 11:30 AM
- Wednesday: 1:00 PM — 2:00 PM
- … or by appointment



Fall 2007

Object Oriented Analysis & Design

- Home
- What's New
- Lectures
- Textbooks
- Evaluation
- Assignments
- Grades
- Projects
- Contact Me

Class Info

Time: 09:30 AM - 10:45 AM TR

Location: ECCR 105

Useful Links

Professor's Home Page

Department of Computer Science

© Kenneth M. Anderson, 2007.

There are two textbooks for the class.

Head First Object-Oriented Analysis and Design by Brett D. Mclaughlin, Gary Pollice and David West. Published by O'Reilly. ISBN-10: 0-596-00867-8. This book offers an excellent introduction to the topic of OO A&D. It introduces the topic gradually making use of code-based examples that illustrate clearly the relationship between decisions made at the analysis and design level (often via the use of UML diagrams) and their impact on the code of an object-oriented system. It also clearly defines analysis techniques in a way that make them feel "more real" to students who may not have been exposed to analysis and design considerations before this class.

A Brain-Friendly Guide to OOA&D
Head First
Object-Oriented Analysis & Design
O'REILLY
Brett D. McLaughlin, Gary Pollice & David West

Head First Design Patterns by Eric & Elisabeth Freeman. Published by O'Reilly. ISBN-10: 0-596-00712-4. This book covers a wide array of design patterns and shows them being used in multiple contexts. It also identifies how some design patterns are simply combinations of more simple patterns used in a particular way. A design pattern is simply a known solution to a common design problem. By learning design patterns and when/where they should be used, you increase your skills both as a programmer and as a software designer. Design patterns let you avoid re-inventing the wheel each time you create a software system and allow you to focus on what is truly unique about the application you are trying to build.

A Brain-Friendly Guide
Head First
Design Patterns
O'REILLY
Eric Freeman & Elisabeth Freeman
with Kathy Sierra & Bert Bates

We will be working our way through the entire Head First OO A&D book during the first eight weeks of the semester and then switch to covering the content of Head First Design Patterns (along with other topics such as refactoring, test-driven design, OO Web application frameworks, etc.) during the last eight

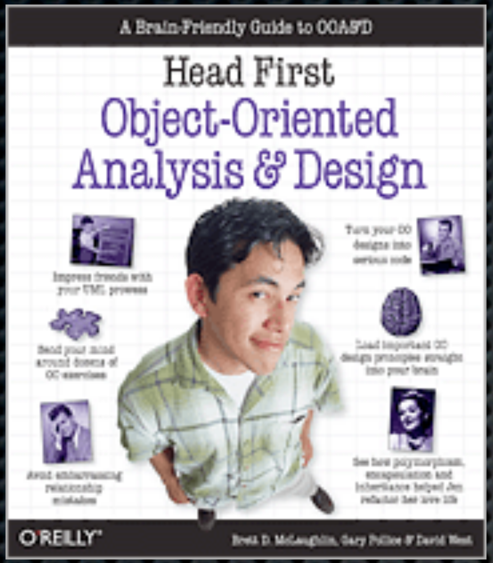
Class Website

<http://www.cs.colorado.edu/~kena/classes/6448/f07/>

About the Class Website

- Check the website *every day!*
  - An RSS feed of the What’s New page is available
- The website is your source for
  - class schedule
  - homework assignments
  - announcements
  - etc.

Textbooks


A Brain-Friendly Guide to OOA&D
Head First
Object-Oriented
Analysis & Design
O'REILLY
Brett D. McLaughlin, Gary Pollice & David West

Head First Object-Oriented Analysis and Design

A Brain-Friendly Guide
Head First
Design Patterns
O'REILLY
Eric Freeman & Elisabeth Freeman
with Kathy Sierra & Bert Bates

Head First Design Patterns

Head First Series

- Lots of examples
  - Including tight Integration of UML and code
- Covers fundamental concepts well
- Employs multiple strategies for teaching concepts
- Repetition, Repetition, Repetition!

- Note: I expect you to do the reading for each lecture! I may decide to “surprise you” with pop quizzes (or worse) if I get the sense that students are skipping the readings

Class Participation

- I welcome participation by students
  - Feel free to interrupt me during lecture to ask questions!
- Stupid Questions — No such thing!
- No participation leads to “silent tomb” — Boring!
- If I’m speaking too fast, stop me and tell me to slow down!

Teaching Philosophy

- “sage-on-stage” vs. “guide-at-your-side”
- Answering questions
  - Sometimes the answer will be “I don’t know!”
- Current Tech
  - I try to incorporate references to current technology in my teaching
    - Ruby, Python, Ruby on Rails, Django, etc.

Bias?

- I don’t use Microsoft Windows and so have no experience with
    - .Net
    - C#
    - etc.
- However, I’m not “anti-Microsoft” and I will welcome student presentations on Microsoft technology

Goals of the Class

- Provide students with knowledge and skills in:
  - object oriented concepts
  - OO analysis, design, and implementation techniques
  - object oriented design methods
    - (aka software development life cycles)
- Students should view OO software development as a software engineering process that has well-defined stages with each stage requiring specific tools and techniques

Course Structure

- First 8 weeks
  - Cover fundamental OO A&D concepts
    - objects, UML, use cases, analysis activities, design activities
- Second 8 weeks
  - Cover additional design and implementation techniques
    - design patterns, refactoring, testing, concurrency, etc.

Course Evaluation

- Homeworks
- OO Framework Analysis and Presentation
  - Teams of 2 to 3 people
- Class Project
  - Teams of 2 to 4 people
- Haven’t decided percentages yet
  - most likely 30/30/40 or 20/30/50.
- NO MIDTERM OR FINAL!

Programming Languages (I)

- Examples will be in Java, Python and Ruby
- OO Programming is not a central topic of the class
    - If you have never programmed using an OO language, start working your way through a tutorial:
        - Java: <http://java.sun.com/docs/books/tutorial/>
        - Ruby: <http://tryruby.hobix.com/>
        - Python: <http://docs.python.org/tut/>
    - Many more available online; use your favorite search engine!

Programming Languages (II)

- Assignments
  - You may use any OO language (within reason) when working on your assignments
    - If you pick C#, I may have to meet with you to see you run your code.
    - Please no Object-Oriented Perl! ☺

Moodle

- We will be using a Web-based system called "The Moodle" to submit homework assignments
- You will need to enroll in the CSCI 4448/6448 course
  - Go to http://moodle.cs.colorado.edu/
  - If you do not have a moodle account, create one
  - Log in
  - Click on "All Classes" button and look for 4448/6448
  - Click on it and follow instructions to enroll
    - You will need an enrollment key: hfdpFall2007

Honor Code

- I encourage collaboration in this class via the framework assignment and the class project
- I'd like, however, for homeworks to be worked on individually
  - As such, the honor code statement will be printed on each homework assignment to remind you that collaboration on the homeworks is not allowed
- The Student Honor Code applies to classes in all CU schools and colleges. You can learn about the honor code at <http://www.colorado.edu/academics/honorcode/>.

Late Policy

- Assignments handed in late incur a 20% penalty
- Assignments can be handed in up to two weeks after initial due date (except for the final assignment of the class project)
    - after that you are out of luck…

Syllabus Statements

- The University asks that various statements be presented to students at the start of each semester
  - Disability Accomodations
  - Religious Observances
  - Classroom Behavior
  - Discrimination and Harassment
  - Honor Code
- These statements are on the class website at: <http://www.cs.colorado.edu/~kena/classes/6448/f07/ss.html>

First Assignment

- Call my office phone at (303) 492-6003
- In your message:
  - State and SPELL your name
  - State that you are enrolled in CSCI 4448/6448
  - Provide me with a nickname for you
  - Spell your nickname for me
- Nicknames used to post grades on the class website
  - As a result, your nickname should not resemble your name or any of your current nicknames!

Coming Up Next

- Lecture 2 and 3: Introduction and Review of Fundamental Object-Oriented Concepts
  - Read Appendix 2 of OO A&D book
- Lecture 4: Great Software
  - Read Chapter 1 of OO A&D book