OMS CS6310: Software Architecture & Design

(version 2 as of Thursday, August 24, 2017)

Introduction

Welcome to this Fall 2017 offering of CS6310: Software Architecture & Design! I am Mark Moss, the Instructor of Record for this semester. Professor Spencer Rugaber is the original designer of this course, and he will introduce much of the course content through the Udacity course videos. I will work with our talented team of Teaching Assistants (TAs) to:
(1) Facilitate discussions, and answer your questions via Piazza and Office Hours (Blue Jeans); and,
(2) Evaluate your assignment submissions through T-Square.

Our current TA Team includes Ray Reese (Head TA), Benjamin Bernhardt, Matt Carter and Roger Kerr. We have more TAs who are new to the team, and I will add their names here as soon as possible. Besides welcoming you and introducing the team, I also wanted to provide some important information about the course. This document includes all of the initial policy and scheduling information for this session of the course. If any of this information changes during the course, then we will update the document as soon as reasonably possible. You must also review Piazza and the Office Hour discussions on a regular basis for any policy, scheduling and/or assignment updates.

Learning Goals and Outcomes

This course teaches the principles and concepts involved in the analysis and design of large software systems. After completing this course, a student should have obtained the skills and knowledge necessary to accomplish the following:

- Express the analysis and design of an application using UML
- Specify functional semantics of an application using OCL
- Specify and evaluate software architectures
- Select and use appropriate architectural styles
- Understand and apply object-oriented design techniques
- Select and use appropriate software design patterns
- Understand and perform a design review

Prerequisites

An undergraduate software engineering course or industrial software development experience is required. Though there are no formal prerequisites in terms of specific Georgia Tech courses (e.g. OMS CS6300: Software Development Process), you are expected to have a solid proficiency in writing computer programs. This course will require you to develop and implement programs using Java using sequence, selection and iteration-based control structures; and, read from (and writing to) external files. Also, you will be required to express your designs using the Unified Modeling Language (UML). Though the lessons include a brief review of UML, you should have a basic understanding of UML syntax and principles, or some similarly well-defined design language.

Schedule

Week / Activity	Reading	Due Date
Week 1 (21 – 25 August 2017)		
Assignment #1: Background Survey [25 pts]		*4 Sep 2017*
Assignment #2: Individual Project Design [100 pts]		*7 Sep 2017*
P1L1: Introduction (14:24)		
P1L2: Text Browser Exercise (14:34)		
P1L3: Design Concepts (24:58)		
P2L1: Review of UML (29:32)	UML Specification, Ch1	
Week 2 (28 August – 1 September 2017)		
P2L2: Object Oriented Analysis (20:12)		
P2L3: UML Class Models (32:50)		
P2L4: Design Studies (09:55)		
P2L5: Library Example (UML) (37:20)		
Week 3 (4 – 8 September 2017)		
OFFICIAL SCHOOL HOLIDAY (Labor Day: 4 September 2017)		
Assignment #3: Individual Project Peer Reviews [75 pts]		*15 Sep 2017*
P2L6: Formal Specification (44:33)		
Week 4 (11 – 15 September 2017)		
Assignment #4: Individual Project Implementation [100 pts]		*25 Sep 2017*
P2L7: OCL (19:21)	OCL Specification, Ch7	
P2L8: Library Example (OCL) (42:44)		
Week 5 (18 – 22 September 2017)		
P2L9: Behavior Modeling (46:38)	Harel Paper	
P2L10: Clock Radio Exercise (31:47)		
Week 6 (25 – 29 September 2017)		
Assignment #5: Statechart *and/or* ***OCL Exercise [100 pts]***		*11 Oct 2017*
Assignment #6: Group Project Team Formation [50 pts]		*2 Oct 2017*
P3L1: KWIC Exercise (18:36)		
P3L2: Software Architecture (34:21)	Garlan and Shaw Paper	
Week 7 (2 – 6 October 2017)		
P3L3: Architectural Views (17:38)	Kruchten Paper	
P3L4: Text Browser Example (Architecture) (24:49)		
P3L5: Architectural Styles and Non-Functional Requirements (13:33)		
Week 8 (9 – 13 October 2017)		
OFFICIAL SCHOOL HOLIDAY (Fall Student Recess: 9 & 10 October 2017)		
Assignment #7: Group Project Design [100 pts]		*30 Oct 2017*
Week 9 (16 – 20 October 2017)		

Assignment #8: Statechart** and/or **OCL Exercise Peer Reviews [75 pts]		*23 Oct 2017*
P3L6: Connectors (24:31)	Mehta Paper	
P3L7: ACME (14:26)	Medvidovic and Taylor Paper	
Week 10 (23 – 27 October 2017)		
Assignment #9: Design Patterns Paper [100 pts]		*6 Nov 2017*
P4L4: Design Patterns (34:04)	Gamma Paper	
P4L5: Design Principles (19:00)	Martin Engineering Notes	
Week 11 (30 October – 3 November 2017)		
P3L8: Refinement (21:25)		
P3L9: Middleware (42:48)	Emmerich Paper	
Week 12 (6 – 10 November 2017)		
Assignment #10: Group Project Implementation [100 pts]		*4 Dec 2017*
Assignment #11: Design Patterns Paper Peer Reviews [75 pts]		*15 Nov 2017*
P3L10: LayerBlox Guest Interview (58:48)		
Week 13 (13 – 17 November 2017)		
P4L1: Components (24:23)		
P4L2: Coffee Maker Example (20:11)	Martin, Chapter 11	
P4L3: Object Design (26:58)		
Week 14 (20 – 24 November 2017)		
OFFICIAL SCHOOL HOLIDAY (Student Recess & Thanksgiving Break: 22 – 24 November 2017)		
Week 15 (27 November – 1 December 2017)		
P4L6: Design Reviews (23:38)		
Geeks in Black: The Code Review (32:59)		
Week 16 (4 – 8 December 2017)		
Assignment #12: Group Project Team Member Assessments [25 pts]		*11 Dec 2017*
Week 17 (11 – 15 December 2017)		
FALL 2017 TERM HAS ENDED (Official grades should be released on 19 December 2017)		

*** There are no midterm or final exams during this session of the course.**

Scoring

The grades will be calculated based on a total of 1000 points, where an A will be earned with 900+ points; a B will be earned with 800 - 899 points; a C will be earned with 700 - 799 points; a D will be earned with 600 - 699 points; and, an F will be earned with any value less than 600 points. The assignment grades are listed above. The remaining 75 points will be granted based on Classroom Participation as a combination of various factors including:

- reviewing the Udacity course videos and completing the Udacity quizzes;
- participating in Piazza discussions; and,
- being an active and supportive member of your team during the Group Design and Implementation assignments as interpreted from the Team Member Assessments.

Grades for graduate students are somewhat different from grades for undergraduates at Georgia Tech: for graduate students, a grade of C or lower normally indicates unsatisfactory performance. We will most likely not offer other extra or bonus point assignments, so you should plan to do your best on the assignments currently listed in the schedule.

Policies

General/Honor Code

- Students are expected to abide by the Georgia Tech Honor Code and academic policies as specified in the Georgia Tech Catalog
    - Honest and ethical behavior is expected at all times
    - All incidents of suspected dishonesty will be reported to and handled by the Office of Student Affairs
    - You are to complete all assignments yourself, unless the assignment instructions explicitly state otherwise
    - You may discuss the assignments with your classmates, but you may not copy any solution (or part of a solution) from a classmate
    - You are welcome and encouraged to form informal study groups at any time, but please do not form formal "project teams" for the group assignments until directed
    - If you are in an informal study group, discussing basic concepts and ideas contained in the course materials and lectures is generally O.K. and encouraged; however, you are not allowed to develop specific answers and/or program code with other students or people outside of the class
    - Do not collaborate beyond what is allowed by the Georgia Tech Academic Honor Code
- Students should complete the GT OMS Orientation before the first day of class GT_Orientation.zip
- Readings should be completed before the lesson for which they are listed
- Any changes to these policies and other course announcements will be posted on T-Square, which you are expected to read frequently

Submissions

- All assignment and project policies, due dates, and submission information will be listed on T-Square
- Most assignment deadlines are set for Mondays at 11:59 PM (2359 Hours) Anywhere-On-Earth (AOE), which generally corresponds to Tuesdays at 8:00 AM Eastern Standard Time (EST)
- We recommend that you go to T-Square => Preferences (left menu) => Time Zone (on top) and configure T-Square appropriately, so that you can see all deadlines in your own time zone
- Being aware of the time zone differences is your responsibility - late submissions will not be excused because of time zone misunderstandings/configuration issues
- Submissions must be made via T-Square; and, if T-Square is down, then you must alert us via Piazza (before the deadline/due date), and then submit your files via e-mail or some alternate method
- We normally publish assignments no earlier than the start date as listed on the schedule
- Late submissions will be accepted in accordance with the following policy:

    - Submissions up to 24 hours late will result in your final score being limited to a maximum of 85% of the total possible score.
    - Submissions up to 48 hours late will result in your final score being limited to a maximum of 70% of the total possible score.
    - Submissions beyond 48 hours will not be accepted.
- We will not support any other late submissions and/or extension requests. In a class of this size, supporting numerous individual extension requests for assignments based on personal scheduling conflicts (e.g. business trips, personal holidays, etc.) can cause tremendous admin difficulties, and create conflicts of fairness for all students
- For team-based assignments, it is essential that you let your teammates know as early as possible about any availability and/or participation problems
- If truly exceptional and/or catastrophic circumstances arise, please feel free to contact The Dean of Students with health emergencies, family emergencies, personal disabilities, or other significant events – they are much better equipped to verify these situations than we are in our individual courses
- Feel free to consult the Georgia Tech policy on Incompletes if needed

Grading

- If you have any questions concerning a grade that you received in this course, first contact the Teaching Assistant who did the grading/evaluation
- All submitted (non-source file) documents must conform to the CS6310 Writing Guidelines
- You have one week after a given grade has been released to submit a regrade request contest; after which, we will consider the grade final
- Regrade requests are comprehensive: if the regrading reveals deductions or other issues that the TA had initially missed, then the regrading might result in a lower grade
- Be careful about "cherry picking" issues for regrading requests, and take time to review your entire assignment before making a request
- All revised grades get updated in "batch mode" on T-Square after the grade contest period has been completed, even if you were notified individually that your grade will be updated earlier in the process
- We will upload grade changes to T-Square as quickly as reasonably possible

Asking Questions and Finding Answers

- Piazza is our online forum, and checking Piazza regularly will contribute to your class participation grade
- Use Piazza, rather than email, for all class-related communications
- Ask questions publicly (visible to the entire class) whenever possible, rather than privately to the Instructors, since that maximizes your chances to get a prompt reply
- Ask questions privately (visible only to the TAs and Instructors) whenever your question includes material that might provide an answer to a specific assignment
- Piazza includes a simple and effective search feature – please search for answers to your questions before posting duplicate questions

- Check updated, unresolved, and unread posts on a regular basis – you are responsible for maintaining awareness of changes and clarifications, especially to assignment requirements

Class Participation

- Class participation will be determined by a number of factors:
    - Completion of the Udacity quizzes;
    - Contributions on Piazza, especially providing solid and well-thought out responses to fellow student's questions;
    - Actively participating and providing substantial contributions to you team during the Group Project Design and Implementation Phases
    - Other, subjective "above-and-beyond" initiatives that you take as judged by the Instructor(s) and TAs
- You are not required to get the correct answer on the first try for the Udacity quizzes; feel free to submit answers even if you are not 100% sure that you are correct
- If you do not get the correct answer after several attempts, try watching the solution video, and then come back to the quiz – only your last submission will be checked

Minimal Technical Requirements

For students taking GT OMS courses, there are the following minimum technical requirements:

- Georgia Tech Computing Guide
    - Georgia Tech's Office of Student Computer Ownership issues the following Minimum Hardware Requirements to incoming undergraduates; you must meet or exceed these guidelines to ensure you have sufficient computing power to complete all course work and assignments
- Browser and Connection Speed
    - An up-to-date version of Chrome or Firefox is strongly recommended
    - We also support Internet Explorer 9 and the desktop versions of Internet Explorer 10 and above (not the metro versions)
    - 2+ Mbps download speed is recommended
- Operating System: 64-bit OS for compatibility with 64-bit VM's
    - PC: Windows XP or higher with latest updates installed
    - Mac: OS X 10.6 or higher with latest updates installed
    - Linux: Any recent distribution that has the supported browsers installed
- Video Submissions
    - Some of the assignments and projects include a video submittal
- Project Submissions
    - Project submissions will require files to be tarred and zipped using only *zip*, *gzip* or *compress*. Please do not use *rar* or other proprietary formats.
    - How To: Tar and Zip Project Files
- Site Support
    - For Udacity site support and course questions not appropriate for the forums (student-specific) please email Udacity Support

Technology Platforms

You will have to use/access four technology platforms (systems) when taking this course: Udacity, Piazza, T-Square and Blue Jeans. Also, we will provide a Virtual Machine (VM) image that you can use to complete your implementation assignments (where applicable).

- **Udacity** – Classroom/Online Videos
  - On Udacity, students will watch online lessons
  - Also on Udacity, students will find this wiki page, a schedule for lessons and assignments, and additional course resources
  - How to Use Udacity
- **Piazza** – Discussion Forums and Announcements (primary)
  - Piazza serves as the class forum; rather than email, all non-personal class-related communications should take place there
  - Students are encouraged to ask their non-personal questions publicly on Piazza so that the instructor, TA's, and classmates can benefit from discussion; it is important that students check Piazza postings regularly
  - How to Use Piazza
  - Piazza Help Portal
- **T-Square** – Assignment Access and Submission, and Announcements (secondary/backup)
  - All class announcements and assignments will be posted on T-Square
  - T-Square will contain the instructions for how to complete each assignment as well as the grading criteria
  - T-Square is where students will submit all assignments
  - T-Square Help Portal
- **Blue Jeans** – Office Hours
  - Discussions about the assignments and course materials will be conducted here, including the opportunity receive direct feedback from the Instructor(s) and TAs
- **Virtual Machines/VMs** – Development and Testing
  - If you decide to use the VM and have any issue with it, please ask for help on Piazza (Virtual Machine Setup)
  - You are not required to use the VM we have provided for your development; you are welcome to develop systems in your own separate development environment/IDE
  - The VM image we've provided will be used as the "reference environment" for grading, so it is highly recommended that you check your submission in this environment before submission

Each of the following design programs are packaged as part of the Ubuntu Virtual Machine. For students who would like to install the design programs on their host machine, we have provided links where you can find out more about each program.

- AcmeStudio - Architecture Description Language (ADL) Tool, (Tutorials)
- ArgoUML - UML Modeling Tool, (User Manual)
- Dresden OCL Parser - OCL Interpreter Plugin for Eclipse, (Documentation)
- Eclipse - Integrated Development Environment, (Help Index)
- Git - Version Control and Source Code Management, (Documentation)

- PHP - General Purpose Scripting Language, (Documentation)
- MySQL - Relational Database, (Documentation)

I am really looking forward to working with all of you this semester. Please remember that we are always happy to hear from you, answer your questions, and gather your feedback. Even with all of the effort that we have put into developing this course so far, we are continually looking to improve the course for you and for future students, and to make your time during the course as productive as possible.

Mark and the OMSCS 6310 Teaching Team

Mark Moss
Ph.D., Computer Science (Georgia Tech, 2009)
Instructor of Record, OMSCS 6310

Other Useful Links

General/Course-Related

- GT's Udacity Portal: https://www.udacity.com/courses/georgia-tech-masters-in-cs
- Udacity OMSCS 6310 Site: https://www.udacity.com/course/software-architecture-design--ud821
- GT OMSCS 6310 Web Page: http://www.omscs.gatech.edu/cs-6310-software-architecture-design/
- T-Square (assignments): https://t-square.gatech.edu/portal

Optional Textbooks

There are no required texts for this course; however, these books cover much of the material:

- Baldwin, Carliss Y. and Kim B. Clark.
  Design Rules, Vol. 1: The Power of Modularity.
  The MIT Press, March 15, 2000.

- Taylor, R. N., N. Medvidovic, and E. M. Dashofy.
  Software Architecture: Foundations, Theory, and Practice.
  Wiley, January 9, 2009.

- Qian, Kai, Ziang Fu, Lixin Tao, Chong-Wei Xu, and Jorge L. Di'az-Herrera.
  Software Architecture and Design Illuminated.
  Jones and Bartlett, 2010.

- Craig Larman.
  Applying UML and Patterns, An Introduction to OO Analysis and Design, 3d Edition.
  Prentice-Hall, 2004.
  Available electronically from the GT Library: Select ebooks, then Safari. Search for Larman.

- Amy Brown and Greg Wilson (Eds.).
  The Architecture of Open Source Applications.
  Published under the Creative Commons Attribution 3.0 Unported license, 2012.

Also, if you don't already own one, access to a Software Engineering introductory text such as Pressman or Sommerville may be helpful.

Other Readings

- Emmerich, Wolfgang.
  [*] "Software Engineering and Middleware: A Roadmap".
  International Conference on Software Engineering - Future of Software Engineering Track, 117-129, 2000.

- Gamma, Erich, Richard Helm, Ralph Johnson, and John Vlissides.
  [*] Design Patterns: Elements of Reusable Object-Oriented Software.
  Addison-Wesley, pp. 6-9, pp. 275-343, 1995.

- Garlan, David, and Mary Shaw.
  [*] "An Introduction to Software Architecture."
  Carnegie Mellon University Technical Report, CMU-CS-94-166, January 1994.

- Harel, David.
  [*] "On Visual Formalisms."
  Communications of the ACM, 32(5):514-530, May 1988.

- Kruchten, Philippe.
  [*] "The 4+1 View Model of Architecture."
  IEEE Software, 12(6):42-50, November-December 1995.

- Martin, Robert.
  [*] "Granularity."
  Excerpt from *The C++ Report,* Vol. 8, No. 10, November 1996.

- Martin, Robert.
  [*] "Heuristics and Coffee."
  Chapter 11 in *UML for Java Programmers,* Prentice Hall, 2003.

- Medvidovic, N., and R. N. Taylor.
  [*] "A Classification and Comparison Framework for Software Architecture Description Languages."
  IEEE Transactions on Software Engineering, 26(1):70-93, January 2000.

- Mehta, Nikunj R., Nenad Medvidovic, and Sandeep Phadke.
  [*] "Towards a Taxonomy of Software Connectors."
  Proceedings of the 22nd International Conference on Software Engineering, Limerick, Ireland, pp. 178-187, 2000.

- Object Management Group.
  [*] OCL Specification V2.3.1.
  January, 2012.

- Object Management Group.
  [*] UML Superstructure Specification V2.4.1.
  August, 2011.

Books

- Baldwin, Carliss Y. and Kim B. Clark.
  Design Rules, Vol. 1: The Power of Modularity Hardcover.
  The MIT Press,March 15, 2000.

- Booch, Grady, James Rumbaugh, and Ivar Jacobson.
  The Unified Modeling Language User Guide, 2nd Edition.
  Addison-Wesley, 2005.

- Jan Bosch.
  Design and Use of Software Architectures.
  Addison-Wesley, May 19, 2000.

- Clements, Paul, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Robert Nord, and Judith Stafford.
  Documenting Software Architectures: Views and Beyond.
  Addison-Wesley, 2003.

- Czarnecki, Krzysztof, and Ulrich Eisenecker.
  Generative Programming: Methods, Tools, and Applications.
  Addison-Wesley, 2000.

- Fowler, Martin.
  Patterns of Enterprise Application Architecture.
  Addison-Wesley, 2003.

- Fowler, Martin.
  UML Distilled: Applying the Standard Object Modeling Language, 3rd Edition.
  Addison-Wesley, 2003.

- Fowler, Martin, Kent Beck, John Brant, William Opdyke, and Don Roberts.
  Refactoring: Improving the Design of Existing Code.
  Addison-Wesley, 1999.

- Gamma, Erich, Richard Helm, Ralph Johnson, and John Vlissides.
  Design Patterns: Elements of Reusable Object-Oriented Software.
  Addison-Wesley, 1995.

- Martin, Robert.
  Clean Code / A Handbook of Agile Software Craftsmanship.
  Prentice Hall, 2008.

- Norman, Don.
  The Design of Everyday Things, Revised and Extended Edition.
  Newprint, 2013.

- Petroski, Henry.
  The Evolution of Useful Things: How Everyday Artifacts-From Forks and Pins to Paper Clips and Zippers-Came to be as They are.
  Vintage, February 1, 1994.

- Prieto-Diaz, Ruben, and Guillermo Arango.
  Domain Analysis and Software Systems Modeling.
  IEEE Computer Society Press, 1991.

- Riel, Arthur.
  Object-Oriented Design Heuristics.
  Addison-Wesley, 1996.

- Rumbaugh, James, Ivar Jacobson, and Grady Booch.
  The Unified Modeling Language Reference Manual, 2nd Edition.
  Addison-Wesley, 2004.

- Shaw, Mary, and David Garlan.
  Software Architecture: Perspectives on an Emerging Discipline.
  Prentice Hall, 1995.

- Simon, Herbert.
  The Sciences of the Artificial, 3rd Edition.
  MIT Press, 1996.

- Szyperski, Clemens.
  Component Software, 2nd Edition.
  Addison-Wesley, 2002.

- Warmer, Jos, and Anneke Kleppe.
  The Object Constraint Language.
  Addison-Wesley, 1999.

- Woodcock, Jim, and Martin Loomes.
  Software Engineering Mathematics.
  Addison-Wesley, 1989.

Papers

- Aksit, M., K. Wakita, J. Bosch, L. Bergmans, and A. Yonezawa.
  "Abstracting Object-Interactions using Composition-Filters."
  Object-based Distributed Processing, R. Guerraoui, O. Nierstrasz, and M. Riveill (eds.), Springer-Verlag, 1993.

- Batory, Don.
  [*] "Feature-Oriented Programming and the AHEAD Tool Suite."
  Proceedings of the 26th International Conference on on Software Engineering, ICSE 2004, pp. 702-703.

- Batory, Don and Sean O'Malley.
  [*] "The Design and Implementation of Hierarchical Software Systems with Reusable Components."
  ACM Transactions on Software Engineering and Methodology (TOSEM), 1(4):355-398, October, 1992.

- Beugnard, Antoine, Jean-Marc Jezequel, Noel Plouzeau, and Damien Watkins.
  [*] "Making Components Contract Aware."
  IEEE Computer, 32(7):38-45, July 1999.

- DeLine, Robert.
  [*] "Avoiding Packaging Mismatch with Flexible Packaging."
  International Conference on Software Engineering '99, Los Angeles, California, 1999, pp. 97-106.

- Ehn, Pelle.
  [*] *"Scandinavian Design: On Participation and Skill."*
  Chapter 4, Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1993, pp. 41-77.

- Eichberg, Michael, Sven Kloppenburg, Karl Klose, and Mira Mezini.
  [*] "Defining and Continuous Checking of Structural Program Dependencies."
  ICSE '08: Proceedings of the 30th International Conference on Software Engineering, Leipzig, Germany, May 2008, pp. 391-400.

- Fagan, M. E.
  "Design and Code Inspections to Reduce Errors in Program Development."
  IBM Systems Journal, 15(3):182-211, 1976.

- Fielding, Roy Thomas.
  "Architectural Styles and the Design of Network-based Software Architectures."
  Department of Information and Computer Science, University of California, Irvine, 2000.

- Lau, Kung-Kiu, and Zheng Wang.
  [*] "Software Component Models."
  IEEE Transactions on Software Engineering, 33(10):709-724, October, 2007.

- Medvidovic, N., D. S. Rosenblum, D. F. Redmiles, and J. E. Robbins.
  [*] "Modeling Software Architectures in the Unified Modeling Language."
  ACM Transactions on Software Engineering and Methodology, 11(1):2-57, January, 2002.

- Meyer, Bertrand.
  "On Formalism in Specifications."
  IEEE Software,, 2(1):6-26, January, 1985.
  Available [*] here, if you enter through the Georgia Tech Library.

- Parnas, David L.
  [*] "On the Criteria to be Used in Decomposing Systems into Modules"
  Communications of the ACM, 15(12):1053-1058, September, 1972.

- Parnas, David L.
  [*] "Designing Software for Ease of Extension and Contraction."
  IEEE Transactions on Software Engineering, SE-5(2):128-138, March, 1979.

- Parnas, David L.
  [*] "On the Design and Development of Program Families."
  IEEE Transactions on Software Engineering, SE-2(1):1-9, 1976.

- Spitznagel, Bridget, and David Garlan.
  [*] "A Compositional Formalization of Connector Wrappers."
  25th International Conference on Software Engineering (ICSE'03), Portland, Oregon, May 3-10, 2003.

- Smaragdakis, Yannis and Don Batory.
  [*] "Implementing Layered Designs with Mixin Layers."
  ECOOP'98—Object-Oriented Programming, pp. 550-570.

- Stirewalt, Kurt and Spencer Rugaber.
  "Automated Invariant Maintenance Via OCL Compilation."
  Proceedings of Model Driven Engineering Languages and Systems'05, October 2-7, 2005, Montego Bay, Jamaica, Lionel C. Briand and Clay Williams, editors, Springer-Verlag, Lecture Notes in Computer Science, volume 3713, pp. 616-632.

Web Resources

- Acme Website
- Allen Holub, *JavaWorld*, "Why extends is evil", August 2003
- Cunningham & Cunningham OO Wiki
- [*] Design Study Template (.doc)
- [*] First Order Logic Paper
- Fowler Enterprise Patterns Page
- JUnit A Cook's Tour
- MSDN Chapter on Architectural Patterns and Styles
- [*] OO Design Review Guidelines
- Patterns Home Page
- [*] Sample Exam Questions
- Wikipedia article on Software Architecture Styles and Patterns

Student Recommendations for Books and Other Design Resources

- Eric Freeman, Elisabeth Robson, Bert Bates and Kathy Sierra. *Head First Design Patterns*. O'Reilly Media, October 25, 2004, link
- Visual Paradigm
- LucidChart free for .edu mails

External Videos

- Java Refresh Video - Part 1 of 2
- Java Refresh Video - Part 2 of 2
- Slides to Accompany Java Refresh Video
- Objects Refresh Video
- Slides to Accompany Objects Refresh Video

Screen Capture/Video Recording and Video Editing

- Windows
  - Camtasia
  - Camstudio
  - Windows Movie Maker
- Mac
  - Photobooth
  - Screenflick
  - Screenflow
  - iShowU HD Pro
  - iMovie

- Linux
  - SimpleScreenRecorder
  - WebcamStudio
  - Cheese
  - Kdenlive
- Web
  - Youtube Video Editor
  - Niaveo.com