CSCI 335

Software Development

Spring 2014 MFW 3:15–4:20 pm SCI 131

http://cs.wheaton.edu/~tvandrun/cs335

Thomas VanDrunen ☎630-752-5692 630-639-2255 ✉Thomas.VanDrunen@wheaton.edu
Office: SCI 163 Office hours: MWF 8:45–10:15 am; Th 8:45–11:15 am.

Contents

CATALOG DESCRIPTION. Principles and practices of software development including design patterns, validation and testing, coordination of team projects. Introduction to data bases and user interface design. Professional issues in computing.

INFORMAL DESCRIPTION. This course has two purposes: it covers the principles of software development (sometimes called *software engineering*), hence the name of the course; it also is a landing place for various topics that need to be covered in the CS core curriculum, but don't fit nicely in any other course.

From the first purpose we have concepts and terms, UML, group dynamics, project and team management, methodology, software life cycle, etc; design patterns; professional and ethical issues; readings, including *The Mythical Man-Month* and "The Cathedral and the Bazaar"; and some discussions about career path options.

From the second purpose we have the basics of database use and design; user interface design, plus GUIs and event-driven programming; regular expressions, automata, and formal languages; and, time permitting, programming languages, distributed computing, web programming, scripting languages, and the history of the computing industry.

TEXTBOOKS. Jalote, *A Concise Introduction to Software Engineering*, Springer, 2008.
Gamma et al, *Design Patterns*, Addison-Wesley, 1994.
Brooks, *The Mythical Man-Month*, Anniversary Edition, Addison-Wesley, 1995

OBJECTIVES. The chief goal of this course is to gain experience working with a team on a large software project. Subordinate goals include

- understanding and using the standard terminology and documentation tools for object-oriented software development.
- articulating the fundamentals of database design and management, languages and automata, and language systems.
- employing design patterns in their design of software systems.
- giving thought to career path and planing accordingly.

TOPICS. Here is a basic topical outline of the course contents. The ordering of the coverage will not match this ordering completely. See the course website for the sequence and schedule of topics.

I. Software engineering concepts

We use the Jalote textbook to explore standard terminology and methods used in software development and engineering. This also provides opportunities to talk about what software development is, the need for good development or engineering practices, and useful tools for development.

A. The software process
B. Requirements, specification, and planning
C. Software architecture
D. Coding and testing

II. Design patterns

This is the best part of the course. The catalog of design patterns found in Gamma et al. codifies much experience and insight in good object-oriented design. It also provides opportunities to talk about programming language features.

A. Review of object-oriented features and introduction to patterns
B. Creational patterns
    1. Abstract Factory
    2. Builder
    3. Prototype
    4. Proxy
C. Prototypical patterns
    1. Composite
    2. Bridge
D. Specialized patterns
    1. Flyweight
    2. Memento
    3. Command
    4. Observer
    5. Façade
    6. Interpreter
    7. Visitor
    8. Chain of Responsibility
E. Refactoring

III. Smaller pieces

A variety of orphan topics have landed in this course. Everything in computer science somehow relates to software development, I guess.

A. Databases
B. Languages
    1. Formal languages
        a. Automata
        b. Regular expressions
        c. Grammars
        d. Parsing
    2. Compilers and interpreters
C. User interfaces
D. Career paths and professional issues

IV. Readings in software development

We discuss some readings about the philosophy of software development and its relation to other fields of study and human activities.

A. "The Cathedral and the Bazaar"
B. *Mythical Man-Month*
C. *The Design of Design*

V. Dormant topics

For historical purposes, we list here other topics that have been covered in this course or that at some time were proposed as topics. These are excluded either from a lack of time or because they are now deemed sufficiently covered by other courses.

A. Concurrency
B. Graphical user interfaces and event-driven programming
C. Apps for mobile devices
D. Eclipse and other IDEs
E. eXtreme Programming
F. Functional programming
G. Occam
H. Scripting languages
I. Distributed computing
J. PHP and other web-programming topics
K. The history of the computing industry

Course procedures

How we do this course. Most of your work outside of class will be on the project. However, some other work is required of you to prepare for class and reinforce the concepts we explore. Most class days will have an accompanying reading; sometimes it will be imperative to read the material before class so that we can discuss it; other times you will have the choice (based on your personal learning style) to read before class to prepare or after class to review. Many class periods will also require you either to write a summary of the reading beforehand or complete a short exercise after class.

Textbooks and readings. I have adopted a textbook for giving an overview of software development, Jalote's *A Concise Introduction to Software Engineering*. We will use this mainly in the first half (especially the first 3 weeks) of the course. See the course website for assigned readings. The reading found at the bottom of a day on the calendar indicates what you must read before the *next* class period. Moreover, there will be questions assigned; write a short (one or two sentence or so) answer for these. Turn in your answers to me *by email* no later that 5:00 pm *the night before class.*

The two other textbooks for this course are important texts that the students should plan to keep. Gamma et al is the standard source for the material covered in the design patterns theme of the course. Students should decide for themselves how they can use the text most effectively: some may choose to read the relevant sections prior to their presentation in class, some may choose to read them after, some may choose to skim first and read carefully after. But by all means students are responsible for the material in them, and mastery of the material will not be obtained without the combination of class presentation, textbook reading, and participation in related exercises.

Brooks' *The Mythical Man-Month*, will be the last topic covered in the course. Selections from this book (which will be supplemented by excerpts from Brooks' more recent book, *The Design of Design*) must be read prior to class discussion on the material.

For each chapter from *The Mythical Man-Month*, *The Design of Design*, and "The Cathedral and the Bazaar", write a paragraph (or so) in which you identify the three (or so) most important points the author is trying to make; what you find the be the most useful insight (which may or may not be one of the most important points to the author); and what you find to be the least useful assertion (such as because it is wrong or obsolete). Turn in this response to me *by email* by 5:00 pm on the *due date*. Please also bring your response and any other notes you have taken to class for use during class discussion.

Library. In the bookcase of the computer science lab, there is a dedicated shelf of books providing additional resources for this course. You are referred to those books for fuller treatments of some of the topics touched on in this course.

Short exercises. On some class days you will be given a short exercise or a response to a reading to reinforce the concept presented or discussed in that class period or prepare you for the next. Short exercises should be turned in by email (unless otherwise noted) before the start of the next class period. These exercises are primarily instructive, not evaluative. Your grades on these are mainly credit for doing the assignment, though a lack of effort or thoroughly incorrect answers will be penalized. Moderate collaboration on these exercises is permitted, though in exercises that require writing code, each student must turn in code written independently.

Project. Most of the work in this course will be on a group project in three phases. In the first phase you will work in teams of three; in the second phase in teams of four and five. These teams will be chosen randomly (with some restrictions to maximize the number of partners people work with). In the third phase, you will all work together as a single team. Your grades will be a combination of whole-team evaluation and individual evaluation. Dates for completing the phases are, tentatively, Feb 14, Mar 7, and Apr 23.

GRADING. The grading scheme below applies only to students who have made a competent contribution to the project, as judged by repository commits, code-documentation author citations, and peer review. Students who do not meet this minimum threshold will fail the class regardless of their scores on other metrics. Students who are not making a competent contribution in the first two phases of the project will be notified promptly, and the instructor and student will discuss a way to remediate the student's performance.

There will be one midterm (scheduled for Mar 21) and a final exam (Thursday, May 8, 10:30 AM).

instrument	*weight*
Project phase 1	15
Project phase 2	20
Project final submission	25
Peer evaluation of project contribution	10
Short exercises and participation	10
Midterm	10
Final	10

Notice that in total the project counts for 70% of a student's grade.

Policies etc

ACADEMIC INTEGRITY. Since all programming in this course will be group work, academic integrity concerns are not about code sharing among students. Instead, students must be careful to document and cite all ideas that come from outside sources. Failure to do so may result in point deduction or rejection of the project altogether. Repeated offenses will be handled through the college's official disciplinary procedures.

LATE ASSIGNMENTS. If a programming project is running late, the group should negotiate with the instructor before the due date (ie, the negotiation should take place before the day on which it is due) to determine a revised due date and a point deduction. Short exercises will not be accepted late.

ATTENDANCE. Students are expected to attend all class periods *on time*. It is courtesy to inform the instructor when a class must be missed.

EXAMINATIONS The midterm is scheduled for Mar 21. The final exam is Thursday, May 8, 10:30 AM. I do not allow students to take finals early (which is also the college's policy), so make appropriate travel arrangements.

GENDER-INCLUSIVE LANGUAGE. For academic discourse, spoken and written, the faculty expects students to use gender inclusive language for human beings. This is of limited relevance for this course, but it's an official college policy that the syllabus contain some sort of notice like this.

SPECIAL NEEDS. *Institutional boilerplate:* Wheaton College is committed to providing reasonable accommodations for students with disabilities. Any student with a documented disability needing academic adjustments is requested to contact the Academic and Disability Services Office as early in the semester as possible. Please call 630.752.5941 or send an e-mail to `jennifer.nicodem@wheaton.edu` for further information.

My own statement: Whenever possible, classroom activities and testing procedures will be adjusted to respond to requests for accommodation by students with disabilities who have documented their situation with the registrar and who have arranged to have the documentation forwarded to the course instructor. Computer Science students who need special adjustments made to computer hardware or software in order to facilitate their participation must also document their needs with the registrar in advance before any accommodation will be attempted.

OFFICE HOURS. My office hours this semester are MWThF 8:45–10:15 am with an extra hour (ie, until 11:15 am) on Thursdays. My teaching schedule this semester is very afternoon-heavy (12:45–4:20 pm MWF, 1:15–3:05 pm Th), so please do not expect me to be available in the afternoons, especially MWF.

If these times do not work for you, please make an appointment for sometime Tuesday or late Thursday afternoon. An email saying "May I stop by in five minutes?" is adequate for making an appointment.

Also, any time my door is closed, it means I'm doing something uninterruptable, such as making an important phone call. Rather than knocking, please come back in a few minutes or send me an email.

Dress and deportment. Please dress in a way that shows you take class seriously—more like a job than a slumber party. (If you need to wear athletic clothes because of activities before or after class, that's ok, but try to make yourself as professional-looking as possible.) If you must eat during class (for schedule or health reasons), please let the instructor know ahead of time; we will talk about how to minimize the distraction.

Electronic devices. Please talk to me before using a laptop or other electronic device for note-taking. I will discourage you from doing so; if you can convince me that it truly aides your comprehension, then I will give you a stern warning against doing anything else besides note-taking. Trying out programming concepts on your own during class time is not productive because it takes you away from class discussion; that is what lab time is for. You cannot multi-task as well as you think you can. Moreover, please make sure other electronic devices are silenced and put away. ***Text in class and DIE.***