COMP 330 Syllabus: Software Engineering

George K. Thiruvathukal[1]

[1]Loyola University Chicago

August 17, 2018

Abstract

Students learn real-world theory and techniques organizations use to create high-quality software on time. Students work on a large programming team to create plans, review progress, measure quality, and make written and oral analyses of their project.

This matches the description at http://courses.cs.luc.edu/html/comp330.html, which is the department's official Course Handbook.

1 Important Info

Instructor George K. Thiruvathukal, Department of Computer Science, Loyola University Chicago

Teaching Assistant Nieky Allen, dallen4@luc.edu

Class Meeting Times Monday/Wednesday/Friday from 1:40PM-2:25PM

Class Location The course has been scheduled for Dumbach 125. We will meet there during the first week of classes. This classroom is not able to accommodate a collaborative experience where students bring their own laptops. As we are a hybrid onsite/online class, it is possible we will move the class 100% online, if we cannot find an alternate classroom.

Office Hours TBD by appointment (recommended) in Doyle 301 (Dr. Thiruvathukal's office at LSC)

E-mail gkt@cs.luc.edu

Google Collaboration thiruvathukal@gmail.com

Box Folder luc.box.com/v/comp330 (requires sign-in with Loyola user ID)

Online Slack Discussion Group luc-se.slack.com (join with your Loyola e-mail)

1.1 In greater detail

Until now, most students coming to this course have likely taken COMP 170, 271, and (possibly) 313. These courses all help to establish a solid foundation for how to craft basic computer programs. Both COMP 313 and 330 are about making the transition toward the development and maintenance of real-world software. The hope is that the student is set up for the transition to "professional life" and more independent work, which may come in the form of an industry job, a start up, or perhaps continued academic/research life.

In this course, we are going to focus on the design and engineering processes with a bit less emphasis (but not zero) on programming. Design and engineering are both crucial in a distributed world, where there are often many moving parts running on a multitude of systems (mobile devices, web browsers, desktops,

clouds, and other networked things, a.k.a. IoT). Writing reliable software is therefore more important than ever. The primary focus of this course is to cover design methods within the context of the software engineering process. A term project is assigned, but the deliverable is a complete software requirements and design specification that is considerably more detailed than what you are likely to have experienced in earlier courses (or internships). We'll place emphasis on the implementation of a high quality[^1] software prototype, which will be evaluated according to multiple criteria besides whether it works.

1.2 Prerequisites

COMP 271 background is an absolute must. Even if you "know the material", the purpose of this prerequisite is to know that you have spent significant time behind a computer and have learned to write functioning programs.

COMP 313 is highly recommended and would prove helpful, especially for the powerful object-oriented background it provides. Fluency with classes and objects is going to be assumed, regardless of whether you have this course, since you should know it from COMP 271. If you don't know OOP, you need to see me right away.

All students are expected to be familiar with core data structures and collection classes in at least one language. I generally do most of my work in Python, Java, and Scala (and formerly in C/C++). There is a substantial development project, and all are expected to contribute. We will assess language fluency/interests in the first couple of weeks of class.

1.3 Recommended Textbook

Pressman and Maxim, Software Engineering: A Practitioner's Approach, 8th Edition

Book Publisher Site: *http://highered.mheducation.com/sites/0078022126/information_center_view0/index.html*

Please note that earlier editions of this book are available. We use this book only as a reference text, so any version you can find in print or online will do. We will discuss this topic during the first class.

1.4 Communication

All communication for this course takes place through Slack, which also allows for private, direct messages with the instructor. Please do not use e-mail unless there is some issue with Slack's availability (which is unlikely). This will greatly enhance your experience as a student.

The Slack URL is *http://luc-se.slack.com/signup*.

We will also use the university Sakai setup for forum-based communication, especially when it needs to be graded. :-)

1.5 Collaboration

We will make extensive use of cloud computing technology throughout the semester. You might as well get started by ensuring you have IDs on these core services:

- Google
- GitHub
- Slack

We will make use of these for project work:

- Trello - allows signup with Google. Teams are required to use Trello for project management.

- Cloud 9 - great for having a team sandbox and sharing with me. I will ask that anyone wanting me to look at code use Cloud 9.
- Bitbucket - private repositories (beyond what GitHub allows)

Others may be added to this list.

1.6 Topics

This is a tentative schedule. I'm loathe to say that this is by week, because some topics will take more time than others. We also need time for project-related manners. A schedule of due dates/deliverables will be posted on Sakai.

Software Process and Practice

- Generic, prescriptive and agile process models
- Software engineering principles and concepts
- Readings: Chapters 1 – 5

Practice and Quality Concepts

- Readings: Chapter 6, 7, 19

Requirements Engineering

- Readings: Chapters 8 - 11
- Readings: Chapters 8 - 11

Quiz 1, date TBD

Design Engineering

- Fundamental concepts
- The design pyramid
- Readings: Chapter 12

Design Methods

- Architectural design
- Readings: Chapter 13

Quiz 2, Date TBD

Design Methods, Continued

- Component design
- Readings: Chapter 14
- User interface design
- Pattern-based design
- Readings: Chapter 15, 16

Web and Mobile Design

- Readings: Chapter 17, 18

Complementary topics

- SQA—an overview
- Testing—an overview
- Test case design
- *Readings:* SEPA, Part 3
- Maintenance and Reengineering
- *Readings:* Chapters 36

Quiz 3, Date TBD

1.7 Special Course Requirements

The following is adapted from William Honig's COMP 474 syllabus (the graduate version of this class). I think it is apropos to this class.

Students will be expected to spend considerable time beyond formal class sessions on this course. Each student will be part of a programming team that will be required to meet at regular times once or twice a week with each session lasting 1 to 2 hours. To ensure that teams are actually meeting, there are required meeting minutes (counted as participation) for every student on every team.

Students are expected to attend all these team meetings in person throughout the course; if you cannot make and keep this commitment, you will not succeed. In addition to the team meeting, each student will need to devote time to their portion of the team's software and generating of plans and metrics on their work; this individual programming time commitment will be similar to other programming intensive courses. Expect a total time requirement of at least 10 to 15 hours a week in addition to classroom time to be devoted to team meetings, your own work on documents and software for the team, and personal reading and study of course materials.

Students may also have difficulty with key course learning areas, especially process-related concepts. Software engineering with real world process methods is very different from an individual programming course where all of the requirements and details are essentially handed down to you. It will seem slow and awkward to do the necessary planning, documentation, and reports. Expect to be frustrated at some point during the course. This feeling is a necessary and valuable experience for the real world and inherent in the team learning process.

1.8 Course Grading/Breakdown

- **Class Participation** - 10% - There is a required weekly posting to the forum on what you have learned each week, whether it comes in the form of lectures, readings, and/or your project team's investigations.
- **Assignments and Quizzes** - 30% - Quizzes will come in the form of in-class, online, or take-home format. Expect 3 of them. Regardless of the number of quizzes, they will count as 30% of your grade, weighted equally.
- **Project** - 50% - The project will comprise many phases, not just coding. We will discuss this in class, but it amounts to a substantial percentage of the grade, because you will spend most of your time on the project, beginning in just a few weeks of the start of class.

1.9 Late Work Policy

All late work will be be subject to a 10% deduction. It is at my discretion whether to deduct the full amount or not, but you should plan for 10%, especially if you are habitually late in turning in work.

1.10 Quiz Makeup Policy

Effective this semester, anyone missing a test without a prior excuse or documented medical/emergency reason will be given a zero for that quiz. Your only option will be to complete an alternate test to make up for the absence, which I reserve the right to make different from the test others have already taken (or longer).

In general, if you are planning to miss a quiz, you should take it before the rest of the class does.