Boston University Metropolitan College

MET CS 673
Software Engineering
Online Course Syllabus

Instructor

Yuting Zhang, Ph.D., danazh@bu.edu

Course Duration

Start: September 2, 2025
End: October 20, 2025

Course credits

4 credits

Course Description

A comprehensive overview of the entire software development lifecycle, emphasizing modern software architectures, methodologies, practices and tools. Key topics include agile principles and methodologies such as Scrum and XP, DevOps concepts and practices, CI/CD pipeline, modern software architectures including microservices, REST and MVC, design patterns, refactoring, software testing, secure software development, and software project management. This course features a semester-long group project where students will design, develop, build and deploy a real world software system, applying Agile methodology, DevOps pipeline and various software tools. Students will also develop AI skills by integrating AI tools into SE lifecycle activities and by developing AI-empowered software systems.

Formatted: Font: Calibri, Font color: Auto,

Formatted: Font: Calibri,

~~This course is an overview of techniques and tools to develop high-quality software. Topics include the software-development life cycle, such as Agile and DevOps; requirements analysis; software design; programming techniques; refactoring; testing; and software-management issues. An overview of secure software development processes and techniques will also be introduced. This course also features a semester-long group project in which students will design and develop a real-world software system in groups using Agile methodology and various software-engineering (SE) tools, including Unified Modeling Language (UML) tools, project-management tools, programming frameworks, unit- and system-testing tools, integration tools, and version-control tools.~~

Prerequisites

- At least two programming-intensive courses at level 500 or above, or
- **OR** The instructor's consent

This course is not about programming. However, the programming skill is the prerequisite. You should be familiar with object-oriented (OO) concepts and proficient in at least one high-level programming language before taking this course. It is better to take this course as a capstone course towards the end of your program study.

Technical Notes

The table of contents expands and contracts (+/- sign) and may conceal some pages. To avoid missing content pages, you are advised to use the next- and previous-page icons in the top-right corner of the learning modules.

This course requires you to access files such as Word documents, PDFs, and/or media files. These files may open in your browser or be downloaded, depending on the settings of your browser.

Learning Outcomes

At the end of the semester, you will be able to do the following:

- Explain and compare major software-process models and activities in the software process
- Explain various architectural patterns and design principles and apply these to design robust, scalable, and maintainable software systems
- Explain methodology and techniques, such as Agile methodology and DevOps, and apply these in a real-world, team-based project to develop a high-quality software system on time
- Identify security risks in the software project and apply various techniques to enhance the software security
- Proficiently use various SE tools including the UML tool, the project management tool, programming tools, testing tools, the version control tool, etc.
- Integrate AI tools into the software development lifecycle, from code generation to testing and deployment.
- Demonstrate AI skills in prompt engineering, tool orchestration in building an AI-empowered software system.
- Communicate effectively with team members and customers
- Clearly present the software project in both the oral and written form
- Adhere to professional standards and practices in software engineering and AI

Formatted: Font: Font color: Black

Course Requirements

- Class participation
- Reading and study
- Labs
- Semester-long project
- Quizzes and Exam

This course features a semester-long, team-based project. Each team should have about four to six students. Every member of the team is expected to contribute a roughly equal share to the project.

Materials

There is no required textbook for this course.

Reference Books:

SE Textbooks

- Braude, E., & Bernstein, M. E. *Software engineering: Modern approaches* (2nd ed.). Waveland Press, Inc.
- Martin, R. C. *Agile Software Development, Principles, Patterns, and Practices.*
- Bruegge, B., & Dutoit, A. H. *Object-Oriented Software Engineering: Using UML, Patterns and Java*.
- Pfleeger, S. L., & Atlee, J. M. *Software Engineering: Theory and Practice*.
- Pressman, R. S. *Software Engineering: A Practitioner's Approach.*
- Van Vliet, H. *Software Engineering: Principles and Practice*.
- Sommerville, I. *Software Engineering*.
- Sommerville, I. *Engineering Software Products: An Introduction to Modern Software Engineering*.
- Farlay, D. *Modern Software Engineering: Doing What works to Build Better Software Faster.*

Other Essential Books for Software Engineers

- Brooks, F. P., Jr. *The Mythical Man Month*.
- Freeman, E., Freeman, E., Bates, B., & Sierra, K. *Head First Design Patterns*.
- Fowler, M., Beck, K., & Roberts, D. *Refactoring: Improving the Design of Existing Code*.
- McConnell, S. *Code Complete: A Practical Handbook of Software Construction*.
- Martin, Robert C. *Clean Code: A Handbook of Agile Software Craftsmanship.*
- Thomas, D. & Hunt, A. *The Pragmatic Programmer: Your Journey to Mastery.*

- Winters, T., Manshreck, T., & Wright, H. *Software Engineering at Google: Lessons Learned from Programming Over Time.*
- Humble, J. & Farley, D. *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation.*
- Kim, G., Behr, K., Spafford, G., and Ruen, C. *The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Win.*
- Forsgren, N., Humble, J., & Kim, G. *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performance Organizations.*
- Kim, G., Humble, J., Debois, P., Willis, J., & Forsgren, N. *The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organizations.*
- Farley, D. *Continuous Delivery Pipelines: How to Build Better Software Faster.*

Other Reading Materials

- SourceMaking
- Microsoft Security Development Lifecycle
- OWASP
  - SAMM project
  - TOP 10
  - Testing Guide
  - Developer Guide

Study Guide

Besides the book chapters, additional reading material will be assigned for each topic. Reading before and after class is required and essential to succeed in this course. Students are responsible for **ALL** materials covered in the lectures and lab sessions, including any topics not in the textbooks.

Both the lectures and the project use iterative approaches. Each lecture includes two iterations. The project includes initial planning and then three mini-project iterations.

This course starts on a **Tuesday**. The modules in this course run from **Tuesday to Monday**.

Module 1 Study Guide and Deliverables
(September 2 – September 8)

Module Topics:

- Topic 1: Introduction to Software Engineering
- Topic 2: Software Processes
- Topic 3: Agile Methodology
- Topic 4: Software Quality Assurance, Configuration Management, and Risk Management

Readings:

- Online lecture notes
- Braude, Parts I, II, and III (or related chapters in other textbooks)

Discussions:

- Introduce Yourself on the Class Discussion Board due **Wednesday, September 3 at 11:59 PM ET**
- Weekly Group Meeting

Assignments:

- Pre-Class Survey due **Wednesday, September 3 at 11:59 PM ET**
- Project Iteration 0 (Proposal) due **Thursday, September 11 at 6:00 AM ET**
- Lab 1 due **Wednesday, September 10 at 6:00 AM ET**

Live Classrooms:

- **Tuesday, September 2 from 7:00-9:00 PM ET**
- **Thursday, September 4 from 7:00-8:00 PM ET**

Module 2 Study Guide and Deliverables (September 9 – September 15)

Module Topics:

- Topic 1: Requirement Analysis and Management Using User Stories
- Topic 2: From Requirement to Design – UML Class Diagrams and State-Transition Diagrams.

Readings:

- Online lecture notes
- Braude, Part IV (or related chapters in other textbooks)

Discussions:

- Weekly Group Meeting

Assignments:

- Lab 2 due **Tuesday, September 16 at 6:00 AM ET**

Assessments:

- Quiz 1 due **Tuesday, September 16 at 6:00 AM ET**

Live Classrooms:

- **Tuesday, September 9 from 7:00-9:00 PM ET**
- **Thursday, September 11 from 7:00-8:00 PM ET**

Module 3 Study Guide and Deliverables (September 16 – September 22)

Module Topics:

- Topic 1: High-Level Design – Software Architecture
- Topic 2: Design Principles and Design Patterns

Readings:

- Online lecture notes
- Braude, Part V (or related chapters in other textbooks)

Discussions:

- Weekly Group Meeting

Assignments:

- Project Iteration 1 due **Tuesday, September 23 at 6:00 AM ET**
- Project Midterm Self and Peer Review due **Thursday, September 25 at 6:00 AM ET**

Live Classrooms:

- **Tuesday, September 16 from 7:00-9:00 PM ET**
- **Thursday, September 18 from 7:00-8:00 PM ET**

Module 4 Study Guide and Deliverables (September 23 – September 29)

Module Topics:

- Topic 1: Implementation
- Topic 2: Testing

Readings:

- Online lecture notes
- Braude, Parts VI and VII (or related chapters in other textbooks)

Discussions:

- Weekly Group Meeting

Assignments:

- Lab 3 due **Tuesday, September 30 at 6:00 AM ET**

Assessments:

- Quiz 2 due **Tuesday, September 30 at 6:00 AM ET**

Live Classrooms:

- **Tuesday, September 23 from 7:00-9:00 PM ET**
- **Thursday, September 25 from 7:00-8:00 PM ET**

Module 5 Study Guide and Deliverables (September 30 – October 6)

Module Topics:

- Topic 1: More UML Tools in Requirement Analysis and Design
- Topic 2: Testing Techniques

Readings:

- Online lecture notes
- Braude, Parts V and VII (or related chapters in other textbooks)

Discussions:

- Weekly Group Meeting

Assignments:
- Project Iteration 2 due **Tuesday, October 7 at 6:00 AM ET**

Live Classrooms:
- **Tuesday, September 30 from 7:00-9:00 PM ET**
- **Thursday, October 2 from 7:00-8:00 PM ET**

Module 6 Study Guide and Deliverables (October 7 – October 13)

Module Topics:
- Topic 1: Secure Software-Development Processes
- Topic 2: Software Security Practices

Readings:
- Online lecture notes

Discussions:
- Weekly Group Meeting

Assignments:
- Project Iteration 3 due **Tuesday, October 14 at 6:00 AM ET**
- Project Final Self and Peer Review due **Thursday, October 16 at 6:00 AM ET**

Assessments:
- Quiz 3 due **Tuesday, October 14 at 6:00 AM ET**

Course Evaluation:

Please complete the course evaluation once you receive an email or Blackboard notification indicating the evaluation is open. Your feedback is important to MET, as it helps us make improvements to the program and the course for future students.

Live Classroom:
- **Tuesday, October 7 from 7:00-9:00 PM ET**
- **Thursday, October 9 from 7:00-8:00 PM ET**

Final Exam Details

The Final Exam is a proctored exam available from **Wednesday, October 15 at 6:00 AM ET to Saturday, October 18 at 11:59 PM ET**. The Computer Science department requires that all final exams be administered using an online proctoring service that you will access via your course in Blackboard. In order to take the exam, you are required to have a working webcam and computer that meets the proctoring service system requirements. A detailed list of those requirements can be found on the How to Schedule page. Additional information regarding your proctored exam will be forthcoming from the Assessment Administrator. You will be responsible for scheduling your own appointment within the defined exam window.

Final Exam Duration: **3 hours**

This is a **closed-book/closed-notes exam**. You can bring 2 pages (either 1 double-sided sheets or 2 single-sided) of cheat sheet.

You can take the exam only once. The exam features **multiple-answer and essay questions**.

Grading Information

Grading Policy

The grade that a student receives in this class will be based on class participation, labs, quizzes, the project, and the final exam. The grade breakdown is as below. All percentages are approximate, and the instructor reserves the right to make necessary changes.

- 5% class participation
- 9% on Lab Assignments (3 small labs)
- 6% on Quizzes (3)
- 50% on Semester-Long Group Project
- 30% on the Final Exam

Letter-grade/numerical-grade conversion is shown below:

A (95-100)
A- (90-94)
B+ (85-89)
B (80-84)
B- (79-77)
C+ (74-76)
C (70-73)
C- (65-70)
D (60-65)
F (0 – 59)

Assignment Submission

All lab reports should be submitted (attached) directly on Blackboard. The project assignments are mostly done on GitHub and Google Drive. Students will work collaboratively on Google Drive for the project documents and commit project code on Github. At each iteration release, the group leader, the configuration leader, or some designated member will commit all required documents on GitHub, together with the source code, to create a release. After it is

done, the group leader should submit a brief summary and the release link on Blackboard. Only one submission is required per group.

Project Assignments

The project assignments are mostly done through GitHub and Google Drive.

There are two types of project assignments:

1. **Individual assignments**—Each student should submit his/her own assignment through Google Drive and Blackboard.
    - **Weekly report**—Fill in a row on your own sheet each week in the group weekly report on Google Docs.
    - **Midterm and final self- and peer review**—Fill in the iteration-review survey form on Google Forms.
2. **Group assignments**—Each group only needs to submit one copy of the whole group's work on GitHub and Blackboard. Students will work on the group documents collaboratively on Google Docs. At each iteration release, the group leader, the configuration leader, or some designated member will archive the documents on GitHub, together with the source code, to create a release. Your GitHub repository should have a readme file, a document subfolder to contain all documents such as meeting minutes, progress reports, iteration presentations, SPPP, SDD, STD, etc, and a code subfolder to contain all code. A final software demo video should also be added at the end of the semester.

Late-Assignment Policy

Each assignment has a deadline. For all individual assignments, late assignments submitted within three days after the deadline will be penalized. **No assignments will be accepted more than three days after the deadline**.

All project deadlines are firm. A deadline miss means a zero for the grade of that phase.

It is the students' responsibility to keep secure backups of all assignments.

Important Message on Final Exams

Dear Boston University Computer Science Online Student,

As part of our ongoing efforts to maintain the high academic standard of all Boston University programs, including our online MSCIS degree program, the Computer Science Department at Boston University's Metropolitan College requires that each of the online courses includes a proctored final examination.

By requiring proctored finals, we are ensuring the excellence and fairness of our program. The final exam is administered online.

Specific information regarding final-exam scheduling will be provided approximately two weeks into the course. This early notification is being given so that you will have enough time to plan for where you will take the final exam.

I know that you recognize the value of your Boston University degree and that you will support the efforts of the University to maintain the highest standards in our online degree program.

Thank you very much for your support with this important issue.

Regards,

Professor Lou Chitkushev, Ph.D.
Associate Dean for Academic Affairs
Boston University Metropolitan College

Policy for the Use of Generative AI

Students should learn how to use AI text generators and other AI-based assistive resources (collectively, AI tools) to enhance rather than damage their developing abilities as writers, coders, communicators, and thinkers.

When using Generative AI in coursework, students shall:

1. Give credit to AI tools whenever used, even if only to generate ideas rather than usable text or illustrations.
2. When using AI tools on assignments, add an appendix showing (a) the entire exchange, highlighting the most relevant sections; (b) a description of precisely which AI tools were used (e.g. ChatGPT private subscription version or DALL-E free version), (c) an explanation of how the AI tools were used (e.g. to generate ideas, turns of phrase, elements of text, long stretches of text, lines of argument, pieces of evidence, maps of conceptual territory, illustrations of key concepts, etc.); (d) an account of why AI tools were used (e.g. to save time, to surmount writer's block, to stimulate thinking, to handle mounting stress, to clarify prose, to translate text, to experiment for fun, etc.).
3. Not use AI tools during in-class examinations, or assignments, unless explicitly permitted and instructed.
4. Employ AI detection tools and originality checks prior to submission, ensuring that their submitted work is not mistakenly flagged.

5. Use AI tools wisely and intelligently, aiming to deepen understanding of subject matter and to support learning.

For more details, please see the Generative AI Assistance (GAIA) policy.

Academic Conduct Policy

Academic Integrity: Plagiarism is the passing off of another's words or ideas as your own, and it is a serious academic offense. Plagiarism and cheating also defeat the purpose of getting an education. Plagiarism and cheating cases will be handled in accordance with the disciplinary procedures described in the College of Arts and Sciences Academic Conduct Code. You are expected to know and abide by the code, which can be read online: Academic Conduct Code. Penalties range from failing an assignment or course (first offense) to suspension or expulsion from BU. If in doubt, cite your source. If you have any questions about academic integrity, please ask your instructor.

Incidents of academic misconduct will be reported to the Academic Conduct Committee (ACC). The ACC may suspend/expel students found guilty of misconduct.

Disability and Access Services

In accordance with University policy, every effort will be made to accommodate students with respect to speech, hearing, vision, or other disabilities. Any student who may need an accommodation for a documented disability should contact Disability and Access Services at 617-353-3658 or at access@bu.edu for review and approval of accommodation requests.

Once a student receives their accommodation letter, they must send it to their instructor and/or facilitator each semester. They must also send a copy to their Faculty & Student Support Administrator, who may need to update the course settings to ensure accommodations are in place. Accommodations cannot be implemented if the student does not send their letter.