Course Policy

Administrative
Office Hours
Course Content
Prerequisite
Resources
Textbooks
   Required
   Recommended
   Reference
Tutorials
   In-class
   External
   References
Electronic resources
   Online
   Course Lab
Grades
   Composition
   Best practice writeup
   In-class labs
   Quizzes
   Homework
   Extra credit
Collaboration

Administrative

- Instructor: Ari Trachtenberg (trachten@bu.edu, 358-1581, PHO 427, Zoom),
- TA: Sumatra Dhimoyee (sumatrad@bu.edu)
Graders:
- Xingyu "Kevin" Chen (chxy517@bu.edu)
- Mingji Han (mjhan@bu.edu)

Office Hours

These are our standard office hours (still under construction). Please pay attention to the announcements for any changes.

<table>
<thead>
<tr>
<th>Day</th>
<th>Time</th>
<th>Staff</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuesday</td>
<td>10 - 11am</td>
<td>Prof. AriTrachtenberg</td>
<td>PHO 427</td>
</tr>
<tr>
<td>Wednesday</td>
<td>6:30-7:30pm</td>
<td>XingyuChen</td>
<td>PHO 307</td>
</tr>
<tr>
<td>Thursday</td>
<td>3-4pm</td>
<td>Prof. AriTrachtenberg</td>
<td>PHO 427</td>
</tr>
<tr>
<td></td>
<td>6:30-7:30pm</td>
<td>MingjiHan</td>
<td>PHO 307</td>
</tr>
<tr>
<td></td>
<td>7:30-8:30pm</td>
<td>SumatraKamalDhimoyee</td>
<td>PHO 307</td>
</tr>
<tr>
<td>Friday</td>
<td>6:30-8:30pm</td>
<td>SumatraKamalDhimoyee</td>
<td>PHO 307</td>
</tr>
</tbody>
</table>

Course Content

Scaling software to many developers, modules, or machines requires a fundamentally different skill-set than writing short prototype code. Software engineers working on large-scale web, financial and healthcare systems, or even multiplayer games must synthesize a wide variety elements at all abstraction layers, ranging from hardware and operating system constraints through full-stack considerations of distributed execution, security, databases, and front-end development.

This course will focus on the principles of scalable and full-stack software engineering, including:

- **Design** – how to design code for efficiency, modularity, interoperability, security, and extensibility;
- **Distribution** – the issues inherent in efficient and reliability distributed processing over many machines;
- **Optimization** – the effects of design and distribution on performance, and optimization through all abstraction layers;
- **Security** – fundamental elements of secure software design and implementation.

Prerequisite

The course will include significant amounts of programming in a number of different languages, primarily JavaScript, Java, C, and C++ and variants thereof. Though there may be short short tutorials on languages, as they are encountered, students will be expected to learn language elements and software tools on the fly, as needed, and to collaborate on increasingly complex and collaborative programming projects. A fundamental background in software programming is thus required, although a background in data structures and algorithms may also be helpful (but is not required).
Resources

Textbooks

Required


Recommended


Reference


Tutorials

In-class

The following tutorials are recommended for students who want more experience with material of relevance to the class.
<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Place</th>
<th>Topic</th>
<th>Instructor</th>
<th>More</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wed, 9/13</td>
<td>6:30-7:30pm</td>
<td>PHO 305</td>
<td>Foswiki and markdown</td>
<td>XingyuChen</td>
<td></td>
</tr>
<tr>
<td>Fri, 9/15</td>
<td>6:30-7:30pm</td>
<td>PHO 305</td>
<td>Tools: git/linux</td>
<td>SumatraKamalDhimoyee</td>
<td></td>
</tr>
<tr>
<td>Wed, 9/20</td>
<td>6:30-7:30pm</td>
<td>PHO 305</td>
<td>Java</td>
<td>XingyuChen</td>
<td></td>
</tr>
<tr>
<td>Wed, 9/27</td>
<td>6:30-7:30pm</td>
<td>PHO 305</td>
<td>C++/gdb</td>
<td>MingjiHan</td>
<td></td>
</tr>
</tbody>
</table>

External

- BU Research Computing has various training modules and online courses: linux, C, C++, and SQL.
- The Java tutorial: https://download.oracle.com/javase/tutorial/ - "A practical guide for programmers"
- The Java language specification: https://docs.oracle.com/javase/specs/ - The formal specification for Java.

References

- How To
  - Code signing - how to cryptographically sign gdb so that it can be used on Mac machines
  - Another course: lecture notes from MIT's core course in software engineering
- Languages
  - JavaScript school - some good documentation on JavaScript; also includes a sandbox for trying things out.
- Online sandboxes:
  - JavaScript - CodeHS
  - Java - CodeHS
  - C++ - CodeHS, replit
- Platforms
  - Documentation for the IntelliJ IDE: https://www.jetbrains.com/idea/documentation/
  - Documentation for the NetBeans IDE: https://netbeans.org/kb/
  - The Clion IDE: https://www.jetbrains.com/clion/

Electronic resources

Online

We will utilize a variety of professional web-based technologies in this course. Note that for security reasons, class servers can only be accessed from the BU network (or VPN).

Course wiki

https://agile.bu.edu/ec500. You are responsible for checking the course wiki page regularly. It will
contain handouts, homeworks, and related material. The wiki will also eventually allow you to check your grades.

**Gitlab**

https://agile.bu.edu/gitlab - This will server as our class git server for version control. We will also use it for issue tracking and peer-review.

**Questions & Answers**

https://piazza.com/bu/fall2023/ec500a3. You may post questions about course material on our class Piazza site. Extra credit problems will also be served through this page.

**Collaboration tools**

You can use Zoom's whiteboard, of Google's Jamboard to collaborate with other students on projects.

**Course Lab**

As part of this course, you also have access to the Signet/VLSI Labs (PHO 305/307), which contain Linux workstations:

- All workstations should have a variety of IDEs installed (IntelliJ, Netbeans, Eclipse, Clion ...), which you may use in assignments or your project.
- If you do not have door access to either PHO 305 or PHO 307, please submit your request through Zaius.
- PHO 307 is available for shared lab hours every weekday from 6:30-9pm.

Unless otherwise stated, you should utilize C++17 and Java 17, for widest compatibility.

**Alternatives**

Several alternatives exist for accessing lab software:

1. **Citrix** - remote access to our licensed applications, although I don't vouch for its security posture.
2. **Eng-grid**: Connect remotely to the lab machines. You can use export windows on the lab machine to your own display.
3. Your own device. Much of our software has freely-available community editions. Please keep in mind that I will not provide support for your own installations - it is up to you to properly maintain and install the software. If you choose this path, you may want to install the following software:
   - The Java Development Kit (jdk 8) for most platforms, which includes the java compiler
   - IntelliJ, a well-designed professional Integrated Development Environment (IDE) that also sports a free Community version.
   - NetBeans, a mature IDE for Java sporting a project management system, debugger, and profiler, together with a graphical layout manager:
   - Eclipse, an alternative open-source Integrated Development Environment for java with a slightly sleeker (and possibly easier to use) interface than Netbeans:
   - Android Studio - an IntelliJ-based IDE for designing and building Android apps.
   - Clion - an IntelliJ-based C++ IDE.
Grades

All grades will be curved according to the class median. Thus, it is your relative score (compared to the rest of the class) that really matters, rather than your objective score. For a course at this level, I expect to center the median at a B/B+, but the final grade will depend on my assessment of the class as a whole.

Composition

Raw scores will be computed based on the following approximate weights:

- 2 Best practice writeups [10%]
- ~10 In-class labs [20%]
- ~6 Quizzes [30%]
- ~4 Homeworks [40%]

Best practice writeup

Many of our classes will cover one more best practices from our reference texts. Each student will sign up to provide an explanation of one best practice, with the following elements:

- Summary of the best practice, in your own words.
- A crisp example from some open-source software project (at least one year old) demonstrating the practice.
- Your own assessment of when or how this practice should be used.

The writeup must be completed within one week of when it is covered in class.

In-class labs

Much of the theoretical material in class will be accompanied by an in-class lab. Labs will typically be completed with assigned lab partners.

Quizzes

Quizzes will be administered from time to time in class, to gauge absorption of class material. The best way to prepare for these quizzes is to keep up to date on the course material and get help from the course staff, where needed.

Unless otherwise stated, quizzes must be completed individually without any external aid (e.g., books, the Internet or other people).

Homework

We will have several homework projects of increasing sophistication. Your homework will consist of several components:
• **Groups:** The homeworks will be completed in groups of various sizes.

• **Peer review:** You will be expected to review code of other students, both within your group and outside your group.

• **Revisions:** Each homework project will evolve over time, adding requirements and features as directed by the instructor.

Homework solutions will be graded based on correctness and the course foci (design, distribution, optimization, and security). Your reviews and revisions will also be graded as part of your homework.

**Extra credit**

Extra credit may be gained by asking or answering questions on the course q2a site that are tagged as extra credit by the instructional staff. You will need to register on the q2a site in order to answer questions.

**Collaboration**

I take cheating and plagiarism very seriously. At the same time, for a course of this type it is appropriate to have a very broad collaboration policy.

With exception of Quizzes (which must be done individually and without external references) and unless otherwise stated:

• You may use external references (books, web, other people) to produce work in this class, but:
  ○ You may only examine static web references, limited to:
    ▪ books and research papers
    ▪ web material that was created before the first day of class
    ▪ material publicly accessible from the course server (agile.bu.edu).
  ○ You may talk to other humans (students, professors) about your work, but:
    ▪ You *may only* look at code of those in your group:
      ▪ Best practice - no one
      ▪ Labs - your lab partners
      ▪ Quizzes - no one
      ▪ Homeworks - *your* homework partners

In any event:

• You must clearly acknowledge all your sources near the locations where they are used.
  ○ This includes any github repositories or web sites that inspired you in completing your work.

• You must make clear what code is your own:
  ○ All borrowed or adapted code *must* be referenced clearly in comments within the code.
  ○ External libraries can only be added in source form. Libraries whose source code is not available or cannot be submitted with your solution may not be used.

• You are responsible for being able to fully explain all code that you submit upon demand (and I will demand it!).

---

CoursePolicy < EC500 < Foswiki

https://agile.bu.edu/fw/bin/view/EC500/CoursePolicy

7 of 8

9/7/23, 1:45 PM
Failure to meet any of the above conditions will constitute a violation of academic integrity in this class. If you are not sure whether something is permitted by the course policy, ASK THE INSTRUCTOR! - it is much more awkward to explain your actions after the fact to the college disciplinary committee.