San José State University
Department of Computer Engineering

CMPE/SE 135
Object-Oriented Analysis and Design

Section 1
Spring 2021

Course and Contact Information

Instructor:	Ron Mak
Office location:	ENG 250 (but working from home)
Email:	ron.mak@sjsu.edu
Website:	http://www.cs.sjsu.edu/~mak/
Office hours:	TuTh 4:30 - 5:30 PM online via Zoom
Class days/time:	TuTh 1:30 - 2:45 PM online via Zoom
Classroom:	Zoom
Prerequisites:	SE Majors: CS 046B (C- or better); CMPE Majors: CMPE 126 (C- or better). Computer Engineering or Software Engineering Majors Only.

Course Catalog Description

"Feasibility analysis and system requirements determination, object-oriented design methodology, and information systems design using object-oriented modeling techniques. Emphasis on both theoretical and practical aspects of object-oriented systems analysis and design. Team-based design project."

Course Format

This course adopts a synchronous online classroom delivery format. To participate in classroom activities, submit assignments, and take tests/exams remotely, a student must have a computer with adequate internet connection and bandwidth for accessing Canvas and attending Zoom video meetings. A smartphone or tablet with a camera capable of running Zoom is also needed for video recording of your test environment during the tests/exams.

Faculty Web Page and Canvas

Course materials, syllabus, assignments, grading criteria, exams, and other information will be posted at my faculty website at http://www.cs.sjsu.edu/~mak and on the Canvas Learning Management System course login website at http://sjsu.instructure.com. You are responsible for regularly checking these websites to learn of any updates. For help with using Canvas see the Canvas Student Resources page at http://www.sjsu.edu/ecampus/teaching-tools/canvas/student_resources.

Course Goals

Become familiar with object-oriented analysis and program design. Employ industry-standard practices of an object-oriented approach to software development. Avoid the pitfalls of object-oriented design.

The primary goal of this course is to become a much better programmer.

The instructor will share decades of experience as a successful software developer in industry, government, and scientific research institutions. The programming examples will be in C++, but the material will apply well to other object-oriented languages such as Java.

Course Learning Outcomes (CLO)

Upon successful completion of this course, students will be able to:

CLO 1: **Requirements gathering:** Gather the requirements for a software application, distinguish between functional and nonfunctional requirements, and express the requirements in the form of use cases.

CLO 2: **Object-oriented analysis:** Derive the appropriate classes from the requirements and define their responsibilities, behaviors, interrelationships, and internal structures. Draw UML use case, class, and sequence diagrams to document and communicate the analysis results.

CLO 3: **Object-oriented design:** Apply the results of analysis to implement the classes and interfaces. Incorporate concepts such as inheritance and polymorphism, programming by contract, coding to the interface, the open-closed principle, the Liskov substitution principle, and the Law of Demeter. Write code that is easily tested and use proven testing techniques.

CLO 4: **Design patterns:** Learn the major "Gang of Four" design patterns and recognize when it is appropriate to apply them.

CLO 5: **The C++ object model:** Understand how C++ implements the object model, including the Standard Template Library (STL). Become aware of the hazards of C++.

CLO 6 **GUI programming:** Develop interactive programs that have a graphical user interface (GUI). Use callback routines with a software framework and comprehend inversion of control.

CLO 7: **Multi-threaded programming:** Learn the basics of programming multiple threads of control using semaphores, mutexes, and critical regions.

You will follow industry-standard best practices and use software development tools that are common in today's software industry.

You will develop the *critical job skill* of working in a small project team to successfully develop a software application that uses shared interfaces and data formats.

Recommended Texts

The following is the original "Gang of Four" design patterns book. The examples are in C++.

Title:	*Design Patterns: Elements of Reusable Object-Oriented Software*
Author:	Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides
Publisher:	Addison-Wesley Professional, 1994
ISBN:	978-0201633610

The following book provides optional background material. The examples are in Java.

Title:	*Object-Oriented Analysis, Design and Implementation: An Integrated Approach*, 2nd edition
Author:	Brahma Dathan and Sarnath Ramnath
Publisher:	Springer, 2015
ISBN:	978-3319242781

Software to Install

You should install and use an interactive development environment (IDE) such as Eclipse CDT. To write interactive programs that have a graphical user interface (GUI), you will need to download and install the wxWidgets package. This is relatively straightforward on the Mac and Linux platforms. However, the Windows platform often has significant compatibility challenges. Therefore, if you're on Windows, you should install the Ubuntu distribution: https://tutorials.ubuntu.com/tutorial/tutorial-ubuntu-on-windows#0.

Some useful tutorials:

- "Install Ubuntu on Windows 10 and on VirtualBox" http://www.cs.sjsu.edu/~mak/tutorials/InstallUbuntu.pdf
- "Configure Ubuntu for Software Development" http://www.cs.sjsu.edu/~mak/tutorials/ConfigureUbuntu.pdf
- "Install Eclipse for Java and C++ Development" http://www.cs.sjsu.edu/~mak/tutorials/InstallEclipse.pdf
- "Install and Configure wxWidgets on MacOS X Ubuntu" http://www.cs.sjsu.edu/~mak/tutorials/InstallwxWidgets.pdf

Course Requirements and Assignments

You should have good C++ programming skills and be familiar with software development tools such as Eclipse.

You will work during the semester in small teams. Programming assignments will provide practice with OOAD techniques and will include developing a game program that uses simple machine learning. Each assignment will include rubrics for its grading criteria.

Each team will also have a semester design project to develop an application that it can demonstrate to the class. Each team will write a short report (10-15 pp.) that describes the design patterns and other OOAD techniques that it used, including a high-level architecture description with UML diagrams.

Each team will submit its assignments and project into Canvas, which will display the scoring rubrics for grading. At the end of the semester, each team will give a presentation and demo of its design project, and students will help to score each presentation.

Each assignment and project will be worth up to 100 points. Late assignments will lose 20 points and an additional 20 points for each 24 hours after the due date.

The university's syllabus policies:

- University Syllabus Policy S16-9 at http://www.sjsu.edu/senate/docs/S16-9.pdf.
- Office of Graduate and Undergraduate Program's Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/

"Success in this course is based on the expectation that students will spend, for each unit of credit, a minimum of 45 hours over the length of the course (normally 3 hours per unit per week with 1 of the hours used for lecture) for instruction or preparation/studying or course related activities including but not limited to internships, labs, clinical practica. Other course structures will have equivalent workload expectations as described in the syllabus."

Technology Requirements

Students are required to have an electronic device (laptop, desktop, or tablet) with a camera and microphone. SJSU has a free equipment loan program available for students.

Students are responsible for ensuring that they have access to reliable Wi-Fi during tests. If students are unable to have reliable Wi-Fi, they must inform the instructor, as soon as possible or at the latest one week before the test date to determine an alternative. See Learn Anywhere website for current Wi-Fi options on campus.

Exams

The exams will test understanding (not memorization) of the material taught during the semester and now well each of you participated in your team assignments and project. Instant messaging, e-mails, texting, tweeting, file sharing, or any other forms of communication with anyone else during the exams will be strictly forbidden.

There can be no make-up quizzes and midterm examination unless there is a documented medical emergency. Make-up final examinations are available only under conditions dictated by University regulations.

Academic Integrity

"Major exams in this class may be video recorded to ensure academic integrity. The recordings will only be viewed if there is an issue to be addressed. Under no circumstances will the recordings be publicly released."

Grading Information

Individual total scores will be computed with these weights:

35%	Assignments*
30%	Design project*
15%	Midterm exam**
20%	Final exam**

* *team scores*
** *individual scores*

Each assignment and exam will be scored (given points) but not assigned a letter grade. The average score of each assignment and exam will be available in Canvas after it has been graded.

Course grades will be based on a curve. Per CMPE Department policy, the median total score will earn a B–. Approximately one third of the class will earn higher grades, and another one third will earn lower grades.

Postmortem Report

At the end of the semester, each student must also turn in a short (under 1 page) individual postmortem report that includes:

- A brief description of what you learned in the course.
- An assessment of your accomplishments for your team assignments and design project.
- An assessment of each of your other project team members.

Only the instructor will see these reports. How your teammates evaluate you may affect your course grade.

Zoom Classroom Etiquette

- **Mute your microphone.** To help keep background noise to a minimum, make sure you mute your microphone when you are not speaking.
- **Be mindful of background noise and distractions.** Find a quiet place to "attend" class, to the greatest extent possible.
    - Avoid video setups where people may be walking behind you, people talking, making noise, etc.
    - Avoid activities that could create additional noise, such as shuffling papers, listening to music in the background, etc.
- **Position your camera properly.** Be sure your webcam is in a stable position and focused at eye level.
- **Limit your distractions and avoid multitasking.** You can make it easier to focus on the meeting by turning off notifications, closing or minimizing running apps, and putting your smartphone away (unless you are using it to access Zoom).
- **Use appropriate virtual backgrounds.** If using a virtual background, it should be appropriate and professional and should not suggest or include content that is objectively offensive and demeaning.

Recording Zoom Classes

This course or portions of this course (i.e., lectures, discussions, student presentations) will be recorded for instructional or educational purposes. The recordings will be posted to the class webpage. The recordings will be deleted at the end of the semester. **If you prefer to remain anonymous** during these recordings, then please communicate with the instructor about possible accommodations (e.g., temporarily turning off identifying information from the Zoom session, including student name and picture, prior to recording).

Students are Not Allowed to Record

Students are prohibited from recording class activities (including class lectures, office hours, advising sessions, etc.), distributing class recordings, or posting class recordings. Materials created by the instructor for the course (syllabi, lectures and lecture notes, presentations, etc.) are copyrighted by the instructor. This university policy (S12-7) is in place to protect the privacy of students in the course, as well as to maintain academic integrity through reducing the instances of cheating. Students who record, distribute, or post these materials will be referred to the Student Conduct and Ethical Development office. Unauthorized recording may violate university and state law. It is the responsibility of students that require special accommodations or assistive technology due to a disability to notify the instructor.

Proctoring Software and Exams

Exams will be proctored in this course through Respondus Monitor, LockDown Browser, and Zoom video meeting. Please note it is the instructor's discretion to determine the method of proctoring. If cheating is suspected the proctored videos may be used for further inspection and may become part of the student's disciplinary record. Note that the proctoring software does not determine whether academic misconduct occurred but does determine whether something irregular occurred that may require further investigation. Students are encouraged to contact the instructor if unexpected interruptions (from a parent or roommate, for example) occur during an exam. Please refer to the online exam instructions for details of the setup and requirements.

Technical Difficulties

- **Internet connection issues:** Canvas autosaves responses a few times per minute as long as there is an internet connection. If your internet connection is lost, Canvas will warn you but allow you to continue working on your exam. A brief loss of internet connection is unlikely to cause you to lose your work. However, a longer loss of connectivity or weak/unstable connection may jeopardize your exam.
- **Other technical difficulties:** Immediately notify the instructor and explain the problem you are facing. Your instructor may not be able to respond immediately or provide technical support. However, the current state of your exam and communication will provide a record of the situation.

Contact the SJSU technical support for Canvas:

Technical Support for Canvas
Email: ecampus@sjsu.edu
Phone: (408) 924-2337
https://www.sjsu.edu/ecampus/support/

University Policies

Per University Policy S16-9, university-wide policy information relevant to all courses, such as academic integrity, accommodations, etc. will be available on Office of Graduate and Undergraduate Program's Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/.

CMPE/SE 135
Object-Oriented Analysis and Design

Section 1
Spring 2021

Course Schedule (subject to change with fair notice)

Week	Dates	Topics
1	Jan 28	Introduction Manage change and complexity An example of iterative development *Form programming teams*
2	Feb 2 Feb 4	Encapsulation Gather functional and non-functional requirements Create use cases Identify objects, behaviors, and dependencies The Functional Specification
3	Feb 9 Feb 11	Key points for good design Abstract classes Designs that scale well The Boost library
4	Feb 16 Feb 18	Analysis precedes design Where do classes come from? UML class, sequence, and state chart diagrams The Principle of Coding to the Interface The Design Specification
5	Feb 23 Feb 25	Class design example Accessors and mutators Immutable classes The Law of Demeter and the Principle of Least Knowledge Cohesion and consistency
6	Mar 2 Mar 4	Programming by contract Preconditions, postconditions, invariants, and assertions Inheritance principles Polymorphism The Liskov Substitution Principle Simple machine learning for the Rock-Paper-Scissors game
7	Mar 9 Mar 11	Code reuse Abstract superclasses The Principle of Favoring Delegation over Inheritance “Has a” vs. “is a” Virtual destructors Overloading vs. overriding Pointers vs. references

Week	Dates	Topics
8	Mar 16 Mar 18	***Midterm exam Tuesday, March 16*** The Model-View-Controller architecture Interactive programming with a graphical user interface (GUI) Introduction to wxWidgets Inversion of control Callback functions Events and event handlers
9	Mar 23 Mar 25	Software frameworks A GUI version of Rock-Paper-Scissors What are design patterns? Strategy Design Pattern Observer Design Pattern Loose coupling
	Mar 29 – Apr 2	***Spring break***
10	Apr 6 Apr 8	The Open-Closed Principle Decorator Design Pattern Factory Method Design Pattern
11	Apr 13 Apr 15	Adapter Design Pattern Facade Design Pattern Principle of Least Knowledge Template Design Pattern State Design Pattern
12	Apr 20 Apr 22	Iterator Design Pattern Composite Design Pattern Lambda expressions Constructor and destructor calls How does a vector grow? Why did my program crash? Shallow vs. deep copy
13	Apr 27 Apr 29	A "safe" array type Overloaded assignment and [] operators Copy constructor The "Big Three" The **`auto`** keyword The **`decltype`** pseudo-function Exception handling Function objects C++ template classes and functions
14	May 4 May 6	The Standard Template Library (STL) Raw pointers vs. unique and shared smart pointers Move semantics The "Big Five"
15	May 11 May 13	Multithreaded programming Project presentations
Final exam	***Thursday May 20***	Time: 12:15 - 2:30 PM Zoom