San José State University
Department of Computer Engineering

CMPE 135 Object-Oriented Design and Analysis

Section 1
Fall 2017

Course and Contact Information

Instructor:	Ron Mak
Office Location:	ENG 250
Email:	ron.mak@sjsu.edu
Website:	http://www.cs.sjsu.edu/~mak/
Office Hours:	TuTh 3:00 - 4:00 PM
Class Days/Time:	TuTh 1:30 - 2:45 PM
Classroom:	BBC 226
Prerequisites:	For SE Majors: CS 046B or for others CMPE 126.

Course Format

This course will be taught primarily face-to-face instruction. Course materials, syllabus, assignments, grading criteria, exams, and other information will be posted on the SJSU Canvas course site at http://sjsu.instructure.com/ You are responsible to check Canvas regularly for class work and exams. You also can find Canvas video tutorials and documentations at http://ges.sjsu.edu/canvas-students

Faculty Web Page and MySJSU Messaging

Course materials such as syllabus, handouts, notes, assignment instructions, etc. can be found on my faculty web page at http://www.sjsu.edu/people/firstname.lastname and/or on Canvas Learning Management System course login website at http://sjsu.instructure.com. You are responsible for regularly checking with the messaging system through MySJSU at http://my.sjsu.edu to learn of any updates.

Piazza will be available for announcements and to serve as an online discussion forum for the class. You are responsible for responding to enrollment invitations.

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 Goals

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

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 mostly in C++, but the material will apply well to other object-oriented languages such as Java and Ruby.

Course Learning Outcomes (CLO)

- **Requirements gathering:** Gather the requirements for a software application, to distinguish between functional and nonfunctional requirements, and to express the requirements in the form of use cases.
- **Object-oriented analysis:** Derive the appropriate objects from the requirements and to define their internal structures, behaviors, and interrelationships.
- **Unified Modeling Language (UML):** Draw UML use case, class, and sequence diagrams to document and communicate the analysis results.
- **Object-oriented design:** Apply the results of analysis and write well-designed programs in an object-oriented language. Design classes and interfaces at various abstraction levels.
- **Object-oriented concepts and techniques:** Apply important concepts such as inheritance and polymorphism, Programming by Contract, Programming to the Interface, the Open-Closed Principle, and the Liskov Substitution Principle. Learn key testing techniques and how to design code that is easily tested.
- **The C++ object model:** Understand how C++ implements the object model, including the Standard Template Library (STL).
- **Design patterns:** Know the major "Gang of Four" design patterns and recognize when it is appropriate to apply which design patterns.
- **Industry-standard best practices and tools:** Follow the best practices and to use the software development tools that are standard in today's software development 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. Your application can then interact with applications from other teams to exchange data and results.

Required Texts/Readings

There is no required textbook. You will receive reading material during the semester.

Course Requirements and Assignments

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

You will work during the semester in small four-person teams. Weekly assignments will provide practice with OOAD techniques. Each student team will develop a working game program that uses simple machine learning. By using shared interfaces, we should be able to have a tournament that pits each team's program against other teams' programs.

Each team will write a report (5-10 pp.) that describes the OOAD techniques that it used and which includes a high-level architecture description with UML diagrams of the major classes.

This is a challenging course that will demand much of your time and effort throughout the semester.

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 Programs' Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/

The University's Credit Hour Requirement:

"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."

Final Examination

Besides a final project from each team, there will be a written in-class final examination for each student. The exam will test understanding (not memorization) of the material taught during the semester and now well each you participated in your project team.

Grading Information

Each assignment will be worth 100 points. For each team assignment, each team member will receive the same score. Late assignments will be penalized 25% and an additional 25% for each subsequent day.

Individual *total scores* will be computed with these weights:

30%	Assignments
35%	Project
15%	Midterm exam
20%	Final exam

Class grades will be based on a curve. The median total score will earn a B. Depending on how all the total scores cluster above and below the median, approximately one quarter of the class will earn higher grades, and another one quarter will earn lower grades.

There can be no make-up midterm or final exams without a valid medical excuse.

Postmortem report

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

- A brief description of what you learned in the course.
- An assessment of your accomplishments for your project team on the assignments and the 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 class grade.

Classroom Protocol

It is very important for each student to attend classes and to participate. Cell phones in silent mode, please.

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 Programs' Syllabus Information web page at http://www.sjsu.edu/gup/syllabusinfo/.

CMPE 135 Object-Oriented Design and Analysis

Section 1
Fall 2017

This schedule is subject to change with fair notice which will be communicated through emails and announcements via Canvas and Piazza.

Course Schedule

Week	Date	Topics
1	Aug 24	Introduction Manage change and complexity Iterative development
2	Aug 29 Aug 31	Analysis activities Gather functional and non-functional requirements
3	Sept 5 Sept 7	Create use cases Identify objects, behaviors, and dependencies The Functional Specification
4	Sept 12 Sept 14	Encapsulation Loose coupling and high cohesion
5	Sept 19 Sept 21	Design artifacts UML class diagrams
6	Sept 26 Sept 28	UML sequence diagrams UML state chart diagrams The Design Specification
7	Oct 3 Oct 5	Object-oriented design techniques Interfaces Inheritance
8	Oct 10 Oct 12	Polymorphism *Midcourse review* ***Midterm exam Thursday, October 12***
9	Oct 17 Oct 19	The C++ object model Pointers and references Constructors, destructors, and copy constructors
10	Oct 24 Oct 26	Public, protected, and private members Concrete, abstract, overridden, and overloaded member functions
11	Oct 31 Nov 2	Friend functions Operator overloading Exceptions
12	Nov 7 Nov 9	Unit testing Test-driven design (TDD) Design patterns
13	Nov 14 Nov 16	Design patterns Frameworks
14	Nov 21	C++ templates The Standard Template Library (STL)
15	Nov 28 Nov 30	GUI frameworks Inversion of control and dependency injection

Week	Date	Topics
16	Dec 5 Dec 7	Multi-threaded programming *Course review*
Final Exam	Tuesday Dec 19	Time: 12:15 - 2:30 PM Room: BBC 226