Student Information Sheet **Fall 2015**

Computer Information Science 28
Object Oriented Analysis and Design

Ron Kleinman
kleinmanronald@fhda.edu

1: Course Description

Money Magazine identified *Software Architect* as the #3 "Best Job" to hold in the U.S. in 2013, and its practitioners earned half again as much as either of the top two finishers. If you have some prior exposure to an object oriented programming language such as C++, Java or C#, and have ever wondered what a Software Architect actually "does" for a living, this could be the course for you.

We will jointly tackle complex "back office" problems including the entire Student Registration System at De Anza College, and the complete data processing requirements of a small commercial airline (from Customer Reservations to Pilot, Plane and Gate assignments and Flight fulfillment). These problems should be especially interesting to those students who desire to understand the boundaries of software components in large organizations and how they interoperate to support important use cases (ex: *Student enrolls in a Section, Customer books an airline Reservation*).

Software Architects deeply "think about", discuss and develop solutions for problems like these using the Object Oriented paradigm, in which the basic abstractions defined in the problem description are identified, fleshed out and manipulated. Note that on truly large projects, it is only after both the system analysis and design are complete that the programmers get called in to implement and deploy the resulting solution. This is exactly the relationship between the architect and the craftsmen (plumbers, electricians and carpenters) during the building of a house. By necessity, they think about the house in very different ways. One of the most important take-aways from this course is "learning how to think like a software architect" and we will get a lot of practice doing that.

While programming a module is most often a solo effort, defining the architecture and design are both inherently collaborative, and students in this class will therefore be strongly encouraged to break themselves up into "solution teams" which will submit joint answers to many of the assigned homework problems. This both reflects the real world of the Software Architect, and additionally provides students with practical experience in software team participation.

Specifically this course defines and utilizes the Object Oriented paradigm for analyzing, designing and implementing computer applications. Topics include techniques for classifying objects in terms of both attributes and behavior, and designing systems of interrelated objects which leverage techniques such as inheritance, polymorphism, encapsulation and the reuse of both interface and implementation.

It is possible to successfully complete this course without writing a line of code. Coding is not required on the homeworks, midterm or final, although there is an extra credit problem which does involve "implementing" a design. However many in-class examples throughout the term will utilize sample code to illustrate the OO concepts (such as public inheritance and a wide variety of collection classes) so that after completing this course, students will understand exactly how to write code to support their future OO designs.

2: Course Goals

The two primary Student Learning Outcomes (SLOs) for this course are to:

- Design and develop complex software solution from raw requirements using Object Oriented Analysis and Design techniques

- Synthesize major architectural patterns and frameworks and apply them to create software solutions

More specifically, the Student will obtain a proficiency in the Object Oriented approach which will enable her/him to examine a wide range of system requirements and:

- Use Object Oriented Analysis (OOA) methodology to identify and "flesh out" major abstractions within the problem space.

- Use Object Oriented Design methodology to refine these abstractions into concrete classes, and determine the relationships between these classes and the operations they support. Several key design patterns will be introduced and used as "prefabricated" building blocks in developing some solutions.

- Understand and use various Unified Modeling Language (UML) artifacts including Collaborative Class Diagrams, Class Lists, State / Event Diagrams, Sequence Flows and Class Responsibility and Collaboration (CRC) cards. The UML artifacts chosen are the subset which support "thinking" about the problem, rather than those which document the design after the fact.

Trade-offs between various OO techniques will be illustrated with a series of real world applications to allow the student to optimize her / his solutions for robustness and reuse.

3: Schedule

Three and one half hours of classroom lecture each week, plus forty five minutes of lab. Plan to spend **at least** an additional five to six hours each week in outside study. Of necessity, this effort will be somewhat concentrated in the second half of the quarter.

Lectures will be used primarily to introduce new OO concepts. The homework will be in the practical application of the classroom material.

4: Materials Required and Recommended

- Various online references as noted during class lectures

- The official course notes for CIS 28, also available in the De Anza Bookstore. Purchase of these notes is required, as all of the lectures will make reference to them.

5: Attendance

This is not a self-study course. The necessary material will be introduced through classroom lecture. Regular and prompt attendance is therefore essential for a good learning experience.

If you reach a total of three (3) unexcused absences, and are behind in your work, you **may** be dropped from the section (but do not count on that). Homework is due on announced dates and late work will not receive full credit.

6: Academic Integrity

Whether an individual or team Homework, assignments will be considered as the work of the student(s) doing the submitting. If it is determined that unidentified outside collaboration occurred between two or more turned in assignments, the actual grade will be divided among the submitters (ex: a perfect grade of 100 will become two grades of 50).

Both the midterm and the final are open book. As noted, the best object analysis and design is often a team effort, so joint outside study and discussion before these tests is both recommended and encouraged. However **all** such collaborative efforts stop at the door of the testing room. A grade of zero will be assigned to any student discovered cheating on an exam – no exceptions. In addition, I must also reserve the right to ask for the ID of anyone taking either the midterm or the final exam to confirm their registration in this section.

7: Grading

Both the midterm and the final examination will be graded on a 100 point scale. A minimum passing grade is 60%. Your grade for this section will be computed by weighing the following factors in roughly the ratio shown:

A: Midterm:	30%
B: Homework:	30%
C: Final Examination:	40%