CS 5386 – Software Design and Architecture
Course Information Sheet

Instructor: Dr. Oscar A Mondragon

office: CCSB 3.1020 **phone:** **747-8015**
email: oamondragon@utep.edu

Office Hours: Monday 9:00 – 9:45 am
Tuesday 5:00 – 5:40 pm or by appointment

Course Description:
This course is concerned with the principles and concepts of engineering of large software systems and programs. Software architecture is an abstraction of system details that helps in managing the inherent complexity of software systems development. Software architecture provides opportunities for early evaluation of user needs, analysis of requirements and design, and prediction of system properties. Architectural styles, views, notations, and description languages provide systematic frameworks for engineering decisions and design practices. The focus of the course is on advanced topics related to software architecture practices, technologies, and artifacts. Students participate in individual or group projects related to developing architectural representations of software systems.

Prerequisites:
- Knowledge of software development life cycles
- experience in the development of software-reliant systems
- some familiarity with modern software engineering concept

Course Purpose

Provide students with knowledge and understanding of issues related to developing the architecture of large-scale software systems. Students are exposed to all elements of architectural design and implementation of software systems; this includes architecture specifications, analysis, patterns, representations, methodologies, tools, and programming environments that are employed.

Learning Objectives:

Upon completion of this course, students should be able to:
- describe the essential elements of software architecture;
- discuss the issues related to architecting a large-scale software system
- describe and understand different software architectures views and styles
- describe, understand, and be able to use the Siemens four-views approach for developing and documenting a software architectures
- describe, understand, and be able to use the AADL (Architecture Analysis & Design Language)
- working as part of a team, develop, analyze and critique an architecture of a software system

Topic Outline:
1. Introduction to Software Architecture
2. Review and discussion of notations
3. Review of Architectural Styles
4. Designing, Describing, and Using Software Architectures
5. Software Architecture Practices and Techniques
6. Software Architecture Examples
7. Architecture Description Languages

Supporting Text Material:

[Bass 2003] Len Bass, et al., Software Architecture in Practice, Addison-Wesley, 2nd. Edition 2003.

[Hofmeister 2000] Christine Hofmeister, et al., *Applied Software Architecture*, Addison-Wesley, 2000.

[Clements 2003] Clements, P., et al., *Documenting Software Architectures: Views and Beyond*, Addison-Wesley, 2003.

[Gluch 2006] Gluch, D., Feiler. P., and Hudak. J., *The Architecture Analysis & Design Language (AADL): An Introduction*, CMU/SEI-2006-TN-011, Carnegie Mellon University, February 2006.

[SAE 2004] *Architecture Analysis & Design Language (AADL)*, SAE AS5506 (Draft), The Society of Automotive Engineers, 2004.

[Shaw 1996] Shaw, M. and Garlan, D., *Software Architecture: Perspectives on an Emerging Discipline*, Prentice-Hall, 1996.

Assignments:

Assignments consist of assigned reading, exercises, special reports, and working as part of a team developing a software product.

Grading Procedures:

1. Each student will work on a team project to gain hands-on experience about the development and documentations of a software architecture. Each team will be required to deliver a set of artifacts associated with the product (e.g., project process and associated documents, and artifacts related to the four-view Sieman approach),). The individual student grade on the project will be a combination of an evaluation of the overall completeness and quality of the product deliverables, project presentations and reviews, and the contribution of the individual to the team effort.

2. Individual and team in-class exercises, covering recent assignments and classroom work, will be given on weekly basis.

3. Two full period exams will be given during the term.

4. Individual and team homework will be assigned as necessary.

5. Participation in class discussion and attending classes is a must in this course

6. Points will be distributed as follows:

Team Project	25%
Exercises, Homework, Discussions, Assignments	20%
Module 1 Quiz	05%
Exam 1	20%
Exam 2	20%
Participation	10%

7. Anyone found cheating on an exam will receive an automatic F in the course.

8. There are no make-ups on in-class exercises. Absence from an exam is excused only in a medical emergency.