

CS 5381: Topics in Software Design

Software Architecture

Spring 2008

CRN: 26000

Lecture: TR 4:30-5:50 pm in CS 321

Website: <http://www.cs.utep.edu/~cheon/cs5383>

Instructor: Yoonsik Cheon (x-8028, ycheon@utep.edu); office hours: TR 1:30-3:00 pm in CS 202-B

Prerequisite: CS 4311 or instructor approval

Description

The following course description is excerpted from the Graduate Course Catalog: “The study of methods and approaches to software design. Topics may include advanced object-oriented design, meta-object protocols, software architectures, and design patterns. May be repeated for credit when topic varies.”

The goals of this course are (1) to learn about software architectures and architectural styles and (2) to have hands-on experience on some of well-known architectural styles. A *software architecture* represents a mapping of the functionality of a software system to software elements and data flows, and provides a high-level system design that does not contain implementation details. Specifically, it consists of a high-level description of system components, definitions of relationships between components, and definitions of relationships between system components and external systems. An *architectural style* or *pattern* defines a family of software systems in terms of a pattern of structural organization, consisting of components (computational elements), connectors (descriptions of interactions between components), and a set of constraints (on how components and connectors can be combined). There are many different architectural styles and a flurry of *architecture description languages (ADLs)* for software architectures and architectural styles. There are also implementations (programming languages, libraries, frameworks, and middleware) supporting particular architectural styles.

Topics:

- Software architectures
- Architectural styles or patterns
- Architecture description languages (ADLs)
- Analysis and reasoning of architectural constraints
- Support tools for software architectures, styles, and ADLs

Textbook

There is no required textbook, as we will read representative or recent research papers. However, the following two books are recommended as references.

- [1] Marry Shaw and David Garlan, *Software Architecture: Perspectives on an Emerging Discipline*, Prentice Hall, 1996.
- [2] Len Bass, Paul Clements, and Rick Kazman, *Software Architecture in Practice*, 2nd edition, Addison-Wesley, 2003.

Examinations

There will be one mid-term exam and the final exam. The mid-term exams will take place during the regular class session and will be 80 minutes in length.

Assignments

There will be two kinds of assignments: in-class presentation and written homework assignments. Students are expected to read and present research papers related to software architectures. The number of presentations will be one or two depending on the class size. The suggested list of papers is found from the course website; the list is

tentative, as the course topics will be decided upon during the first few course meetings. There will be occasional written homework or programming assignments. All assignments shall be done individually unless otherwise specified, and no late submission will be accepted.

Projects

You should do a small semester-long class project. The purpose of this project is to apply the ideas and skills learned from the course to your thesis or dissertation research. Sample project topics will be suggested by the instructor, but you may choose your own project topics; your topics must be approved by the instructor. You are expected to write a project proposal, present your project in class, and submit a final project report.

Grading

Your grade is independent of anyone else's grade; that is, we do not grade on a curve. Everyone can get an A in this course. The purpose of grading is not to rank you, but to uphold a standard of quality and to give you feedback. The final letter grade will be based on a combination of assignments, project, exams, and class participation. The approximate percentages are as follows:

Assignment:	30%
Project:	35%
Exam:	35%

There are also up to 5% bonus points for lecture attendance and class participation. To earn this, you must arrive at lecture on time and participate in class discussion in a constructive and prepared manner, e.g., by asking or answering questions that demonstrate that you have read and attempted to understand the material.

The nominal percentage-score-to-letter-grade conversion is as follows:

90% or higher:	A
80-89%:	B
70-79%:	C
60-69%:	D
below 60%:	F

I reserve the right to adjust these criteria downward, e.g., so that 88% or higher represents an A, based on overall class performance. The criteria will not be adjusted upward, however.

Attendance

Lecture attendance is not mandatory but is recommended. You should understand that your success in the course will improve greatly by attending class regularly. It is your responsibility to keep up to date with notes, assignments and projects.

Standards of Conduct

You are expected to conduct yourself in a professional and courteous manner, as prescribed by the UTEP Standards of Conduct. Graded work (assignments, projects, exams) is to be completed independently and should be unmistakably your own work, although you may discuss your work with others in a general way. You may not represent as your own work material that is transcribed or copied from another source, including persons, books, or Web pages. Instructors are required to—and will—report academic dishonesty and any other violation of the Standards of Conduct to the Dean of Students.

Disabilities

If you feel that you may have a disability that requires accommodation, contact the Disabled Student Services Office at 747-5184, go to Room 106E Union, or email dss@utep.edu.

Schedule

The following table shows a planned schedule for the course; refer to the course website for an up-to-date schedule.

Dates		Topics	Readings	Assignments
Week 1	Jan. 15	Software architectures and styles	[Garlan-Shaw94]	Homework 1
Week 2	Jan. 22	Software architectures and styles		
Week 3	Jan. 29	Architecture description languages	[Medvidovic-Taylor02]	
Week 4	Feb. 5	Acme	[Garlan-Monroe-Wile04]	Homework 2
Week 5	Feb. 12	AchJava/Armani	[Aldrich-Chambers-Notkin02] [Monroe01]	
Week 6	Feb. 19	ArchJava/Armani		Homework 3
Week 7	Feb. 26	C2 architecture and style	[Taylor-etal96]	
Week 8	Mar. 4	Exam 1		
Week 9	Mar. 11			
Week 10	Mar. 18	Project proposal presentation		Project proposal
Week 11	Mar. 25	Spring Break		
Week 12	Apr. 1			
Week 13	Apr. 8			
Week 14	Apr. 15			
Week 15	Apr. 22			
Week 16	Apr. 29	Project presentation		Project report
Week 17	May 6	Final at 4:00 pm – 6:45 pm		

Important Dates

- January 14: Class begins
- January 21: Martin Luther King Jr. Day – University closed
- January 30: Census day
- March 6: Exam 1
- March 20: Course drop deadline
- March 21: Good Friday – no classes
- March 24-28: Spring Break – no classes
- March 31: Cesar Chavez Day – no classes
- May 2: Dead day
- May 6: Final on Tuesday at 4:00 pm – 6:45 pm

Readings

- [Aldrich-Chambers-Notkin02] J. Aldrich, C. Chambers, and D. Notkin, ArchJava: Connecting Software Architecture to Implementation, *Proceedings of the 24th International Conference on Software Engineering*, pages 187-197, Orlando, Florida, May 2002.
- [Galan-Monroe-Wilie00] D. Garlan, R. T. Monroe, and D. Wile, Acme: Architectural Description of Component-Based Systems, *Foundations of Component-Based Systems*, pages 47-68, Cambridge University Press, 2000.
- [Garlan-Shaw94] D. Garlan and M. Shaw, An Introduction to Software Architecture, *Advances in Software Engineering and Knowledge Engineering, Volume I*, edited by V. Ambriola and G. Tortora, World Scientific Publishing Company, New Jersey, 1993.
- [Medvidovic-Taylor00] M. Medvidovic and R. N. Taylor, A Classification and Comparison Framework for Software Architecture Description Languages, *IEEE Transactions on Software Engineering*, 26(1):70-93, January 2000.
- [Monroe01] R. T. Monroe, *Capturing Software Architecture Design Expertise with Armani*, Technical Report CMU-CS-98-163, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, January 1992.
- [Taylor-etal96] R. N. Taylor, N. Medvidovic, K. M. Anderson, E. J. Whitehead, J. E. Robbins, K. A. Nies, P. Oregzy, and D. L. Dubrow, A Component- and Message-Based Architectural Style for GUI Software, *IEEE Transactions on Software Engineering*, 22(6): 390-406, June 1996.