17-633 and 17-882 Architectures for Software Systems

[M/W/F 9:30am-10:50am]
[Spring 2024, 12 Units]

Instructor	Email	Office Location and Hours
Prof. David Garlan	garlan@cs.cmu.edu	Wednesdays, 4-5 Remote (15 min appointment slots)
Dr. Bradley Schmerl	schmerl@cs.cmu.edu	Fridays, 11:30-12:30 TCS 421 (15 min appointment slots.)
Dr. George Fairbanks	gfairban@andrew.cmu.edu	

Course Description. *Architectures for Software Systems* teaches you how to design, understand, and evaluate systems at an architectural level of abstraction. By the end of the course you will be able to:

- Understand the technical, organizational, and business role of software architecture in software engineering.
- Recognize how architectural drivers influence the choice of architectural designs, and will be able to identify and define the key drivers for realistic software systems.
- Identify key architectural structures (styles, patterns, tactics, etc.).
- Generate architectural alternatives in a given context and choose among them Apply principles of good architectural documentation and presentation to describe a software architecture and the rationale behind its design.
- Understand the impact that open source and third party components have on architectural designs, including ways to integrate mismatched elements.
- Understand how formal notations can be used to specify architectures.
- Evaluate the fitness of an architectural design in meeting a set of system requirements and balancing quality tradeoffs.
- Integrate software architecture and software architecture thinking into a range of software processes and domains.
- Be aware of and respond to the future trends in software architecture.

Prior Knowledge. Students should have had experience developing medium- to large-scale software, preferably in an industrial and team-based setting.

Lectures and Recitation: There will be two lectures and one recitation weekly. Attendance at lectures and recitations is mandatory. We believe that in-person, in-class conversation and participation is critical to learning from each other; therefore, we will not record or allow remote participation, except in exceptional circumstances.

Computing: A personal computer or laptop is required for this course. For some course assignments we will be using the Java programming language. You will need to download and install the Java Software Development Kit (the latest version of J2SE). You may use any editor or development environment that you like.

Course Meetings: All lectures and recitations will be in person.

Class: Monday/Wednesday, 9:30AM - 10:50PM, SCR 265
Recitation: Fridays, 9:30– 10:50PM, SCR 265

Learning Resources.

Required Text: *Software Architecture in Practice, Fourth Edition*, by Bass, Clements, Kazman, Addison-Wesley 2021 [BCK21].

We will also use a collection of supplementary readings, which will be available through the course web site.

Reference/Auxiliary Texts: (not required, but worth owning)

- *Architecting Software Intensive Systems: A Practitioner's Guide*, Anthony Lattanze, Auerbach 2008 [Lat08].
- *Documenting Software Architectures: Views and Beyond, Second Edition*, by Paul Clements, et al. Addison-Wesley 2011 [C+11].
- *Software Architecture: Perspectives on an Emerging Discipline*, Mary Shaw and David Garlan, Prentice-Hall, 1996. [SG96]

Assessments. Student learning is enhanced through applying and explaining ideas to others; thus, the course includes the following activities:

Assessment	Description	Grade %
Quizzes	Most weeks, there will be a short quiz on the topics of classes of the week, that will be discussed at the beginning of the recitation. No makeups. The score of the lowest quiz will be dropped.	10%
Assignments	There will be 5 assignments. 2 assignments will be team assignments. Late assignments will be penalized 10% per day.	45%

Projects	There will be two projects: an architecture design of a fictitious system, and an architecture design for your Studio.* Each team project will produce a report focusing on a preliminary architectural design for a relevant system, and an in-depth analysis of several architectural design choices for it.	40%
Attendance	Attendance in class (including recitations) will be tracked. 1% will be deducted for every 2 classes missed (without a valid excuse pre-arranged with the instructors). 1% will be deducted for every 4 classes where you are more than 20 minutes late.	5%
Instructor Discretion	The instructors reserve the right to adjust grades by grade point, depending on factors such as in-class participation, effort, teamwork based on peer reviews for team projects.	

* If you are enrolled in 17-882 (the PhD version of the course), there will be an additional research project that you will need to produce. This project will be worth 20% of the grade, with the other 80% being proportionally distributed as above. If you are enrolled in this section, please reach out to the instructors for details.

Course and Grading Policies

- **Submission policy:** assignments and projects should follow the following:
- Individual submitted assignments should be named `LastName-FirstName-AssignmentNumber`(where `LastName` = your last name, `FirstName` = your first name, and `AssignmentNumber` = the assignment number), or for team projects, `TeamID-AssignmentNumber` (where `TeamID` = the team id and `AssignmentNumber` = the assignment number). Example: `Team1-Assignment1.docx`
- The time that the assignment is uploaded will be counted as the submission time.
- Assignments are to be submitted via Canvas.
- Assignments should be in either Microsoft Word or pdf format
- **Late-work policy**: Work is expected to be handed in at the respective due date and time. Late assignments will be penalized 10% per day.

Accommodations for Student Disabilities. If you have a disability and have an accommodations form from the Disability Resources office, we encourage you to discuss your accommodations and needs with us as early in the semester as possible. We will work with you to ensure that accommodations are provided as appropriate. If you suspect that you may have a disability and would benefit from accommodations but are not yet

registered with the Office of Disability Resources, we encourage you to contact them at access@andrew.cmu.edu.

Academic Integrity. Honesty and transparency are important to good scholarship. Plagiarism and cheating, however, are serious academic offenses with serious consequences. If you are discovered engaging in either behavior in this course, you will earn a failing grade on the assignment/task in question, and further disciplinary action may be taken. For a clear description of what counts as plagiarism, cheating, and/or the use of unauthorized sources, please see the University's Policy on Academic Integrity. If you have any questions regarding plagiarism or cheating, please ask us as soon as possible to avoid any misunderstandings. For more information about Carnegie Mellon's standards with respect to academic integrity, you can also check out the Office of Community Standards & Integrity website.

Use of Generative and other AI tools:

The point of being a software architect is to have a deep understanding of architecture drivers, design alternatives, and tradeoffs to make educated judgements about the design of software. We recognize that generative AI tools are now a fact of life, and can be helpful if used appropriately. But they cannot substitute for the creative and critical thinking you need to be an architect. You are permitted to use them in assignments in class, subject to some restrictions:

- **Attribution**: If you use AI tools to help you in assignments or projects, you must include a brief statement (e.g., as comments in the code or as an appendix to the paper) saying how they were used.
- **Responsibility**: You must not assume that anything you get from an AI tool is correct or appropriate; you must verify the correctness and appropriateness before using its outputs.
- **Understanding**: You must not use AI as a substitute for learning. If you get something from an AI tool, you must understand it before you incorporate it, and we reserve the right to ask you to explain your work.

If you're not sure whether some use of AI tools is acceptable, ***seek advice from the instructors***.

You are not permitted to use AI tools for the open book quizzes.

Failure to follow these policies may result in an Academic Integrity Violation and a risk to your position in the program.

Student Wellness. As a student, you may experience a range of challenges that can interfere with learning, such as strained relationships, increased anxiety, substance use, feeling down, difficulty concentrating and/or lack of motivation. These mental health concerns or stressful events may diminish your academic performance and/or reduce your ability to participate in daily activities. CMU services are available, and treatment does work. You can learn more about confidential mental health services available on campus

at the Counseling and Psychological Services website. Support is always available (24/7) from Counseling and Psychological Services: 412-268-2922.

Respect for Diversity. We intend that students from all backgrounds and perspectives be well served by this course, that students' learning needs be addressed both in and out of class, and that the diversity that students bring to this class be viewed as a resource, strength, and benefit. We intend to present materials and activities that are respectful of diversity: gender, sexuality, disability, age, socioeconomic status, ethnicity, race, and culture. We encourage and appreciate your suggestions. Please let us know if any of our class meetings conflict with your religious observations so that we can make alternate arrangements for you.

Course Schedule. **Note that bold face indicates readings in the course textbook: [BCK21]**

#	Date	Major Topic	Lectures & Recitations	Assignments and Quizzes	Readings
L1	M Jan 13	**Introduction and Big Ideas**	What is software architecture		
L2	W Jan 15		Some Big Ideas in Software Architecture		**Ch 1-2** [GS94] pp.1-5 [Fai23]
R1	F Jan 17		Thinking like an Architect	Quiz 1	
	M Jan 20	No Class			
L3	W Jan 22	**Requirements: Drivers and Quality Attributes**	Architecture Drivers and Quality Attributes	A1 Assigned	**Ch 3**
R2	F Jan 24		Architecture Drivers	Quiz 2	
L4	M Jan 27		Quality Attribute Trees and Trade-offs		[GS94] pp.5-16
L5	W Jan 29	**Styles**	Intro to Architecture Styles.	A1 Due A2 Assigned	
R3	F Jan 31		More Architecture Drivers	Quiz 3	

L6	M Feb 3		Dataflow: Theory and Practice		**Ch 17** [Bos09]
L7	W Feb 5		Events: Theory and Practice	A2 Due A3 Assigned	
R4	F Feb 7		Architecture Styles	Quiz 4	
L8	M Feb 10		Call Return: Theory and Practice		[Mic09], Ch3
L9	W Feb 12		Repository Style. Styles in Practice.		
R5	F Feb 14			Quiz 5	
L10	M Feb 17	**Architecture Design: Tactics and Frameworks**	Tactics: Availability, etc.	A3 Due Project 1 Assigned	**Ch 4, 8, 9**
	W Feb 19		Tactics: Modifiability		**Ch 8**
R6	F Feb 21		Tactic Practice	Quiz 6	

L12	M Feb 24		Platforms, Frameworks, Product Lines, and Ecosystems		
L13	W Feb 26		Frameworks and Mismatch		
R7	F Feb 28		Project 1 Progress Consult	Quiz 7	
SPRING BREAK					
L14	M Mar 10		Case Studies: ROS, YouTube, etc.		
L15	W Mar 12	**Representation and Reasoning**	Architecture Evaluation	A4 Assigned	**Ch 21, [Mar+05]**
R8	F Mar 14		Project 1 Presentations	Project 1 Due	
L16	M Mar 17		Principles of Architecture Documentation		**Ch 18**
L17	W Mar 19		Architecture Decision Records, Michael Keeling, Senior Staff Software Engineer, Kiavi		[Nyg11] [KR18]

R9	F Mar 21		Architecture Decision Records	Quiz 9	
L18	M Mar 24		Architecture modeling in C4		[Bro22, 14:10-]
L19	W Mar 26		Modeling and Analysis		**Ch 24.3, [GS06]**
R10	F Mar 28		In-Class A4 Presentations	A4 Due A5 Assigned	
L20	M Mar 31	**Architecture Process**	Architecture in the 21st Century Architecture Processes		[Bro22, video [0:00-14:10] **Ch 20**
L21	W Apr 2	**Architecture and Code**	Architecture Recovery		[LCG+15, TDS+22, SAG+06]
			No Class (Spring Carnival)		
L22	M Apr 7		Architecture Hoisting and Evident Coding	A5 Due Final Project Assigned	
L23	W Apr 9		QA Analysis with F' Security		[Can22, B+18 skim]
R11	F Apr 11		Architecture as Theory Building		

L24	M Apr 14	**Architecture and AI**	Architecture and AI (Grace)		
L25	W Apr 16		Tactics for AI		
	F Apr 18		Final Project Progress Consults		
L26	M Apr 21		TBD		
L27	W Apr 23		Trends and Research, Wrap up		
R13	F Apr 25		Final Project Presentations		
	M Apr 28			Final Project Report Due	

References

GS94	Introduction to Software Architecture Download Introduction to Software Architecture . David Garlan and Mary Shaw. CMU Technical Report CMU-CS-94-166. 1994.
Bos09	From Software Product Lines to Software Ecosystems Download From Software Product Lines to Software Ecosystems . Jan Bosch.
Bro22	The lost art of software design. Simon Brown. Devoxx 19th Edition, Belgium, 2022. Video Links to an external site.
Can22	The Mars Ingenuity Helicopter -- A Victory for Open-Source Software Download The Mars Ingenuity Helicopter -- A Victory for Open-Source Software . Timothy Canham. *2022 IEEE Aerospace Conference (AERO)*, Big Sky, MT, USA, 2022

Fai23	Software Architecture is a Set of Abstractions Download Software Architecture is a Set of Abstractions . G. Fairbanks, IEEE Software 40(4), 2023.
Fai22	Two Kinds of Iteration Links to an external site. . G. Fairbanks, IEEE Software 39(1), 2022
Fai21	Why Is It Getting Harder To Apply Software Architecture? Links to an external site. G. Fairbanks, IEEE Software 38(4), 2021.
FGH06	The Architecture Analysis & Design Language (AADL): An Introduction. Download The Architecture Analysis & Design Language (AADL): An Introduction. P. Feiler, D. Gluck, J. Hudak, CMU Technical Note CMU/SEI-2006-TN-011, 2006.
Mar+05	Architecture Reviews: Practice and Experience Download Architecture Reviews: Practice and Experience . J. Maranzano, S. Rozsypal, G. Warnken, D. Weiss, P. Wirth, and G. Zimmerman. *IEEE Software* 2005
GA095	Architectural Mismatch: Why Reuse is so Hard Download Architectural Mismatch: Why Reuse is so Hard . David Garlan, Robert Allen and John Ockerbloom. In IEEE Software, Vol. 12(6):17-26, 1995.
GCSS04	Rainbow: Architecture-Based Self-Adaptation with Reusable Infrastructure Download Rainbow: Architecture-Based Self-Adaptation with Reusable Infrastructure . Garlan, Cheng, Schmerl, Steenkiste. *IEEE Computer* 0018-9162/04/2004.
Gar14	Software Architecture: A Travelogue Download Software Architecture: A Travelogue . David Garlan, ACM978-1-4503-2865-4/14/05.
Gar+24	Challenges in Creating Effective Automated Design Environments: An experience report from the domain of generative manufacturing. Download Challenges in Creating Effective Automated Design Environments: An experience report from the domain of generative manufacturing. David Garlan, Bradley Schmerl, Rebekka Wohlrab and Javier Cámara. In *Proc. the 1st International Workshop on Designing Software*, 15 April 2024.

GS06	Architecture-driven Modelling and Analysis. Download Architecture-driven Modelling and Analysis. David Garlan and Bradley Schmerl, 2006.
KR18	Share the Load: Distribute Design Authority with Architecture Decision Records. Links to an external site. Michael Keeling and John Rundle. Agile18, 2018.
LCG+15	Comparing software architecture recovery techniques using accurate dependencies. Thibaud Lutellier, Devin Chollak, Joshua Garcia, Lin Tan, Derek Rayside, Nenad Medvidović, and Robert Kroeger. In Proceedings of the 37th International Conference on Software Engineering - Volume 2 (ICSE '15). IEEE Press, 69–78, 2015.
Mic09	Microsoft Application Guide, Chapter 3: Architecture Patterns and Styles. Links to an external site. Microsoft, October 2009.
Nyg11	Documenting Architecture Decisions, by Michael Nygard. 2011.
Pace+24	The Architect in the Maze: On the Effective Usage of Automated Design Exploration Download The Architect in the Maze: On the Effective Usage of Automated Design Exploration . Andres Diaz-Pace and David Garlan. In Proc. the 1st International Workshop on Designing Software, 15 April 2024.
SAG+06	Discovering Architectures from Running Systems. Bradley Schmerl, Jonathan Aldrich, David Garlan, Rick Kazman and Hong Yan. In IEEE Transactions on Software Engineering, Vol. 32(7), July 2006.
TDS+22	"ROSDiscover: Statically Detecting Run-Time Architecture Misconfigurations in Robotics Systems," C. S. Timperley, T. Dürschmid, B. Schmerl, D. Garlan and C. Le Goues, *2022 IEEE 19th International Conference on Software Architecture (ICSA)*, Honolulu, HI, USA, 2022, pp. 112-123

VB19	Vogelsang, Andreas, and Markus Borg. "VB19.pdf Download VB19.pdf ." In Proc. of the 6th International Workshop on Artificial Intelligence for Requirements Engineering (AIRE), 2019.