CMU CS 17-651 & 17-751 Models of Software Systems Fall 2018

Garlan & Kang

Course Information and Syllabus

Instructors

David Garlan
garlan@cs.cmu.edu
Wean Hall 4218 (x8-5056) or 300 S. Craig (SCR) 273
Assistant: *Margaret Weigand* (x8-1252, Wean Hall 4212 weigand@cs.cmu.edu)

Eunsuk Kang
eskang@cmu.edu
Wean Hall 5319 (x8-3761)

Teaching Assistants

Kenji Yonekawa
kyonekaw@andrew.cmu.edu
300 S. Craig (SCR), Tomayko Library

Changjian (CJ) Zhang
changjiz@andrew.cmu.edu
300 S. Craig (SCR), Tomayko Library

Objectives

Scientific foundations for software engineering depend on the use of precise, abstract models and logics for characterizing and reasoning about properties of software systems. A number of basic models and logics over time have proven to be particularly important and pervasive in the development of software systems. This course is concerned with that body of knowledge. It considers many of the standard models for representing sequential and concurrent systems, such as state machines, algebras and traces. It shows how you can use different logics to specify properties of software systems, such as functional correctness, deadlock freedom, and internal consistency. Concepts such as composition mechanisms, abstraction relations, invariants, non-determinism, and inductive and denotational descriptions are recurrent themes throughout the course.

By the end of the course, you should be able to understand the strengths and weaknesses of certain models and logics, including state machines, algebraic and trace models, probabilistic models, and temporal logics. You should be able to apply this understanding to select and describe abstract formal models for certain classes of systems. Further, you should be able to reason formally about the important properties of modeled systems.

Course Organization

Lectures. Classes meet Mondays and Wednesdays 10:30-11:50 a.m. in 300 S. Craig, Room 265.

Recitation Sessions. To allow more time to answer questions, and to cover certain topics in greater depth, we will hold weekly recitation sessions Fridays 12:00–1:30 p.m.in 300 S. Craig, Room 265. These are optional, but strongly recommended.

Communication. We will use the course website for distributing most course materials, providing a general course bulletin board, and keeping track of student email addresses. Be sure to check the site periodically for postings about the class.

Office Hours: The instructors and the TAs have weekly office hours, listed on the course website. We are also available at other times by appointment.

Email: We welcome email about the course at any time.

Readings. Most lectures will have a reading assignment that we expect you to complete *before* you come to class. There are **three** required **textbooks** for the course:

- *Models of Software Systems*, by D. Garlan, J. Wing, O. and Celiku [GWC10].
- *Concurrency: State Models and Java Programs*, Second Edition, by Magee and Kramer [MK06], and
- *Software Abstractions*, Revised Edition: Logic, Language, and Analysis, by Daniel Jackson. MIT Press, 2011 [Jac11]
- One optional reference books may also be useful. (Copies of this book are available for loan in the Tomayko LIbrary.)

One optional **reference** book may also be useful:

- *Logic in Computer Science, Modelling and Reasoning about Systems, Second Edition* Michael Huth and Mark Ryan. Cambridge University Press, 2004. (http://www.cambridge.org/uk/computerscience/huth). Available for loan in the Tomayko Library in 300 S. Craig.

Some readings are in the form of **handouts** to supplement lectures; other additional readings are **technical papers**. We will make these available as needed throughout the course. Finally, for supplementary detail, there are several books and articles noted in the Supplementary References section of the course website.

Homework Assignments. We have organized the course around weekly homework assignments and three projects. The purpose of these is to give you practice in using the models, logics, and tools of the course.

To give you the most opportunities to learn from the homework assignments, we will allow you to redo problems for which you did not receive a passing grade, provided you made a reasonable initial attempt. Normally, you will turn in your redone homework at *the class following the one on which it we returned the graded initial attempt*. Details on grading can be found at the course website.

Projects. We will be assigning two group projects. These are designed to give you a chance to apply the ideas of the course to semi-realistic case studies. You will complete each project in a team. We expect team members to participate equally in the projects and will conduct team peer evaluations periodically.

On-line materials. Most of the course materials will be available electronically via the course website. Some of the course texts have their own web sites. These are:

- *Concurrency: State Models and Java Programs:* http://www.doc.ic.ac.uk/~jnm/book/
- *Software Abstractions:* http://softwareabstractions.org/
- *Logic in Computer Science: Modelling and Reasoning about Systems*: http://www.cambridge.org/uk/computerscience/huth

Another useful source of on-line materials is the Open Learning Initiative, which has a good course on logic and proof (free to CMU students). It can be found at http://oli.cmu.edu/learn-with-oli/see-our-free-open-courses/.

PhD Option. Students taking the course for PhD credit (17-751) will be required to complete a course project.

Exams. There will be a take-home midterm exam, **handed out after class Wednesday, October 10 and due at noon on Friday, October 12.** The final examination will be 3 hours in length from 9:00 a.m.-noon on **Monday, December 10**. Both exams will be open-book.

Grading. The course grade will be determined as a combination of five factors: homework assignments (30%), projects (30%), midterm exam (15%), and final exam (25%). Final grades may be adjusted up or down based on the instructor's judgment, and taking into account factors like evidence of effort, peer evaluations, extra credit problems, quizzes, and participation in class.

Cooperation Policy. We encourage you to discuss your homework with other students, but *the final write-up must be your own work. It is not ok to obtain an electronic or physical copy of any other student's homework and use this as the basis for your own*. If you work out ideas with someone on a whiteboard, you should erase the whiteboard before recreating your own homework. Note that when copying occurs both parties are in part to blame -- even if one person copies from another. If you have any questions about what is appropriate, please ask the instructor or one of the teaching assistants. Also, see the University Policy on Academic Integrity: http://www.cmu.edu/policies/student-and-student-life/academic-integrity.html.

Schedule

#	Date	Topic	Subtopic	Reading	Due
1	M 08/27	Introduction	What is a model?		
2	W 08/29	Foundations	Propositional & Predicate Logic	**Ch 1-3**; N+15	HW 1
3	W 09/05*		Proof Techniques	**Ch 4, 5**	HW 2
4	M 09/10		Sets, Relations, Functions	**Ch 6**	HW 3
5	W 09/12		Sequences & Induction	**Ch 7**	
6	M 09/17	State Machines	State Machines 1	**Ch 8**	HW 4
7	W 09/19		State Machines 2	**Ch 9**	
8	M 09/24		FSP & LTSA	MK06 Ch 1, 2.1,	HW 5
9	W 09/26		Reasoning about State Machines	**Ch 10**	
10	M 10/01	Structural Modeling	Object Models		HW 6
11	W 10/03		Alloy		
12	M 10/08		Modeling State Dynamics		HW 7
13	W 10/10		Refinement		Midterm– due 10/12 at noon
14	M 10/15		Advanced Structural Modeling		HW 8
15	W 10/17		Project 1 Lab	Project 1	
16	M 10/22	Concurrency	Introduction to Concurrency	AS83, MK06 Ch 1-3	HW 9
17	W 10/24		Concurrency Modeling Techniques	MK06 Ch 4-5	Project 1
18	M 10/29		Reasoning about Concurrency	MK06 Ch 6-7	HW 10
19	W 10/31		LTL	Kat96	
20	M 11/05		LTL in FSP	MK06 Ch 14	HW 11
21	W 11/07		Project 2 Lab	Project 2	
22	M 11/12	FM in Pr1actice	Probabilistic Models 1	S+14	HW 12
23	W 11/14		Probabilistic Models 2		Project 2
24	M 11/19		FM in the Real World	BLR11, Cal+13	HW 13
25	M 11/26*		Introduction to Petri Nets	Pet77	
26	W 11/28		Reasoning about Petri Nets	Jen91 Section 1	HW 14
27	M 12/03		UML	RCB99, Ha87 Sections 1-3	
28	W 12/05		Review	Skim SF03	HW 15
29	M 12/11		Final Examination (9:00-noon)		

Bold refers to GWC10

*Marks classes that follow a holiday

Important Dates

Project	Assigned	Lab	Due	Topic
Project 1	10/10	10/17	10/24	Structural Modeling
Project 2	10/31	11/07	11/14	Concurrency
Midterm	10/10	----	10/12	----
Final Exam	12/10	----	----	----

References

AS83 Concepts and Notations for Concurrent Programming, Andrews and Schneider. Computing Surveys, Vol. 15, No. 1, March 1983.

BLR11 A Decade of Software Model Checking with SLAM, T. Ball, V. Levin, S. K. Rajamani, Communications of the ACM, Vol. 54. No. 7, 2011, Pages 68-76.

Cal+13 Radu Calinescu, Kenneth Johnson, Yasmin Rafiq, Simos Gerasimou, Gabriel Costa Silva, Stanimir N. Pehlivanov: Continual Verification of Non-Functional Properties in Cloud-Based Systems. NiM-ALP@MoDELS 2013: 1-5.

DG90 A Formal Specification of an Oscilloscope, N. Delisle and D. Garlan, *IEEE Software*, September 1990

SG03 The Aura Software Architecture: an Infrastructure for Ubiquitous Computing, João Pedro Sousa, David Garlan, CMU-CS-03-183.

GWC10 Models of Software Systems, D. Garlan, J. and Wing, O. Celiku. Draft of 2010.

Har87 Statecharts: a visual formalism for complex systems. D. Harel. Science of Computer Programming, 8:231-274, 1987.

Jen91 Coloured Petri Nets: A High Level Language for System Design and Analysis. K. Jensen. In High-level Petri Nets: Theory and Application. K. Jensen and G. Rozenberg (eds.) Springer-Verlag, 1991.

Kat96 Temporal Logic Draft version of chapter from book in preparation. 1996.

MGY08 Comparison of Software Specification Methods Using a Case Study, by M. Yusufu and G. Yusufu, 2008 International Conference on Computer Science and Software Engineering, IEEE Computer Society, pp. 784-787.

MK06 Concurrency: State Models and Java Programs, Second Edition. J. Magee and J. Kramer. Wiley, 2006

N+15 How Amazon Web Services Uses Formal Methods. Newcombe, Rath, Zhang, Munteanu, Brooker, and Deardeuff, Communications of the ACM, April 2015, Vol. 58, No. 4, pp. 66-73.

Pet77 Petri Nets. J. L. Peterson. ACM Computing Surveys, Sept 1977.

RCB99 UML Walkthrough. J. Rumbaugh, I. Jacobson, and G. Booch. In The Unified Modeling Language Reference Manual. Addison Wesley, 1999, pp. 25-39.

SG03 The Aura Software Architecture: an Infrastructure for Ubiquitous Computing. João Pedro Sousa, David Garlan. August 2003. CMU-CS-03-183

Supplemental References

Abr07 Formal Methods: Theory Becoming Practice, by J.-R. Abrial. Journal of Universal Computer Science, Vol. 13, No. 5, (2007) 519-628

BBF01 Systems and Software Verification: Model Checking Techniques and Tools. B. Berard, M. Bidoit, and A. Finkel. Springer Verlag, 2001.

CW96 Formal Methods: State of the Art and Future Directions, Edmund M. Clarke and Jeannette M. Wing, report by the Working Group on Formal Methods for the ACM Workshop on Strategic Directions in Computing Research, ACM Computing Surveys, vol. 28, no. 4, December 1996, pp. 626-643. Also CMU-CS-96-178.

GS95 A Logical Approach to Discrete Math. D. Gries and F.B. Schneider. Springer-Verlag, 1993.

Hoa85 Communicating Sequential Processes. C.A.R. Hoare. Prentice-Hall International, 1985.

HR04 Logic in Computer Science: Modelling and Reasoning about Systems, M. Huth and M. Ryan, Second Edition, Cambridge University Press, 2004.

Jac97b Imperial College Alloy Tutorial. www.doc.ic.ac.uk/project/examples/2007/271j/suprema_on_alloy/Web/tutorial1_1.php

Jac02 Alloy: A Lightweight Object Modelling Notation, Daniel Jackson, ACM Transactions on Software Engineering and Methodology, Vol. 11, No. 2, April 2002, pp. 256-290.

MP91 The Temporal Logic of Reactive and Concurrent Systems Specification. Z. Manna and A. Pnueli. Springer-Verlag, 1991. Covers linear temporal logic. The relevant sections are 3.0-3.4.

Sch00 Concurrent and Real-time Systems: The CSP Approach. Steve Schneider. Wiley, 2000.

WL88 Software Engineering Mathematics. J. Woodcock and M. Loomis, Addison-Wesley 1988.