1. Course Information

Staff

Faculty

Butler Lampson	NE43-535	547-9580 x3-6004	blampson@microsoft.com
Martin Rinard	NE43-620A	x8-6922	rinard@lcs.mit.edu

Teaching Assistant

Mandana Vaziri	NE43-369	x3-6097	vaziri@theory.lcs.mit.edu

Course Secretary

Alicia Briceland	NE43-620	x3-9620	aliciab@lcs.mit.edu

Office Hours

Messrs. Lampson and Rinard will arrange individual appointments. Mandana Vaziri will hold scheduled office hours outside of NE43-365. In addition to holding regularly scheduled office hours, the TA will also be available by appointment.

Lectures

Lectures are held on Mondays and Wednesdays from 1:00 to 2:30PM in room 37-212. Messrs. Lampson and Rinard will split the lectures.

Handouts

The source material for this course is an extensive set of handouts. There are about 400 pages of topic handouts that take the place of a textbook; you will need to study this material to do well in the course. Seven research papers supplement the topic handouts. In addition there are 5 problem sets, and the project described below. Solutions for each problem set will be available shortly after the due date.

There is a course Web page, at URL ftp://theory.lcs.mit.edu/pub/classes/6.826/www/6.826-top.html. Last year's handouts can be found from this page. Current handouts will be placed on the Web as they are produced, as Postscript (.ps), Acrobat (.pdf), and Word (.rtf) files.

Current handouts will generally be available in lecture. If you miss any in lecture, you can obtain them afterwards from the course secretary. She keeps them in a file cabinet outside her office.

Problem sets

There is a problem set approximately once a week for the first half of the course. Problem sets are handed out on Wednesdays and are due in class the following Wednesday. They normally cover the material discussed in class during the week they are handed out. Delayed submission of the solutions will be penalized, and no solutions will be accepted after Thursday 5:00PM.

Students in the class will be asked to help grade the problem sets. Each week a team of students will work with the TA to grade the week's problems. This takes about 3-4 hours. Each student will probably only have to do it once during the term.

We will try to return the graded problem sets, with solutions, within a week after their due date.

Policy on collaboration

We encourage discussion of the issues in the lectures, readings, and problem sets. However, if you collaborate on problem sets, you must tell us who your collaborators are. And in any case, you must write up all solutions on your own.

Project

During the last half of the course there is a project in which students will work in groups of three or so to apply the methods of the course to their own research projects. Each group will pick a real system, preferably one that some member of the group is actually working on but possibly one from a published paper or from someone else's research, and write:

A specification for it.

A high-level implementation that captures the novel or tricky aspects of the actual implementation.

The abstraction function and key invariants for the correctness of the implementation. This is not optional; if you can't write these things down, you don't understand what you are doing.

Depending on the difficulty of the specification and implementation, the group may also write a correctness proof for the implementation.

In general, the specification and implementation should be written in Spec. But you can also use IOA, a formal language based on I/O automata, which is a mathematical model used for describing components of distributed systems. The IOA system comes with a parser and a static semantic checker that are ready for you to use. It also comes with verification tools that are currently being connected to the system. For a more research-oriented project, you can try to use some of these tools as well. Consult the TA if you would like to explore this possibility.

Projects may range in style from fairly formal, like the handout on consensus, in which the 'real is a simple one, to fairly informal (at least by the standards of this course), like the section on copying file systems in handout 7. These two handouts, along with the ones on naming, sequential transactions, concurrent transactions, and caching, are examples of the appropriate size and possible styles of a project.

The result of the project should be a write-up, in the style of one of these handouts. During the last two weeks of the course, each group will give a 25-minute presentation of its results. We have allocated four class periods for these presentations, which means that there will be twelve or fewer groups.

The projects will have five milestones. The purpose of these milestones is not to assign grades, but to make it possible for the instructors to keep track of how the projects are going and give everyone the best possible chance of a successful project

1. We will form the groups around March 1, to give most of the people that will drop the course a chance to do so.
2. Each group will write up a 2-3 page project proposal, present it to one of the instructors around spring break, and get feedback about how appropriate the project is and suggestions on how to carry it out. Any project that seems to be seriously off the rails will have a second proposal meeting a week later.
3. Each group will submit a 5-10 page interim report in the middle of the project period.
4. Each group will give a presentation to the class.
5. Each group will submit a final report, which is due on the last day allowed by MIT regulations. Of course you are free to submit it early.

Half the groups will be ‘early’ ones that give their presentations on April 28 or May 3; the other half will be ‘late’ ones that give their presentations on May 10 or May 12. The due dates of proposals and interim reports will be spread out over two weeks in the same way.

Grades

There are no exams. Grades are based 30% on the problem sets, 50% on the project, and 20% on class participation and quality and promptness of grading.

Course mailing list

A mailing list for course announcements—6826@mit.edu—has been set up to include all students and the TA. If you do not receive any email from this mailing list within the first week, check with the TA. Two additional mailing lists have also been provided: 6826-staff@mit.edu allows students to send email to the entire 6.826 staff, and 6826-disc@mit.edu is an unofficial forum for discussion among students.

Course Schedule

Date	No	By	HO	Topic	PS out	PS due
Wed., Feb. 3	1	L		**Overview**. The Spec language. State machine semantics. Examples of specifications and implementations.	1	
			1	Course information		
			2	Background		
			3	Introduction to Spec		
			4	Spec reference manual		
			5	Examples of specs and implementations		
Mon., Feb. 8	2	L		**Specification and implementation** for sequential programs. Correctness notions and proofs. Proof methods: abstraction functions and invariants.		
			6	Abstraction functions		
Wed., Feb. 10	3	L		**File systems 1**: Disks, simple sequential file system, caching, logs for crash recovery.	2	1
			7	Disks and file systems		
Tues., Feb. 16	4	L		**File systems 2**: Copying file system.		
Wed., Feb. 17	5	L		**Proof methods**: History and prophecy variables; abstraction relations.	3	2
			8	History variables		
Mon., Feb. 22	6	R		**Semantics and proofs**: Formal sequential semantics of Spec.		
			9	Atomic semantics of Spec		
Wed., Feb. 24	7	R		**Performance**: How to get it, how to analyze it.	4	3
			10	Performance		
			11	Paper: Michael Schroeder and Michael Burrows, Performance of Firefly RPC, *ACM Transactions on Computer Systems* **8**, 1, February 1990, pp 1-17.		
Mon., Mar. 1	8	L		**Naming**: Specs, variations, and examples of hierarchical naming.	Form groups	
			12	Naming		
			13	Paper: David Gifford et al, Semantic file systems, *Proc.13th ACM Symposium on Operating System Principles*, October 1991, pp 16-25.		
Wed., Mar. 3	9	L		**Concurrency 1**: Practical concurrency, easy and hard. Easy concurrency using locks and condition variables. Problems with it: scheduling, deadlock.	5	4
			14	Practical concurrency		

Date	No	By	HO	Topic	PS out	PS due
			15	Concurrent disks		
			16	Paper: Andrew Birrell, An Introduction to Programming with Threads, Digital Systems Research Center Report 35, January 1989.		
Mon., Mar. 8	10	L		**Concurrency 2:** Concurrency in Spec: threads and non-atomic semantics. Big atomic actions. Safety and liveness. Examples of concurrency.		
			17	Formal concurrency		
Wed., Mar. 10	11	R		**Concurrency 3:** More examples. Proving correctness of concurrent programs.		5
Mon., Mar. 15	12	R		**Concurrency 4:** Examples of correctness proofs		
Wed., Mar. 17	13	L		**Distributed consensus** in the presence of faults. Paxos algorithm for asynchronous consensus.		Early proposals
			18	Consensus		
Mar. 20-28				Spring Vacation		
Mon., Mar. 29	14	L		**Sequential transactions** with caching.		
			19	Sequential transactions		
Wed., Mar.31	15	L		**Concurrent transactions**: Specs for serializability. Ways to implement the specs.		Late proposals
			20	Concurrent transactions		
Mon., Apr. 5	16	R		**Introduction to distributed systems**: Characteristics of distributed systems. ISO Reference Model: physical, data link, and network layers. Design principles. **Networks 1**: Links. Point-to-point and broadcast networks.		
			21	Distributed systems		
			22	Paper: Michael Schroeder et al, Autonet: A high-speed, self-configuring local area network, *IEEE Journal on Selected Areas in Communications* **9**, 8, October 1991, pp 1318-1335.		
			23	Networks: Links and switches		
Wed., Apr. 7	17	R		**Networks 2**: Links cont'd: Ethernet. Token Rings. Switches. Implementing switches. Routing. Learning topologies and establishing routes.		
Mon., Apr. 12	18	L		**Networks 3**: Network objects and remote procedure call (RPC).		
			24	Network objects		

Date	No	By	HO	Topic	PS out	PS due
			25	Paper: Andrew Birrell et al., Network objects, *Proc.14th ACM Symposium on Operating Systems Principles*, Asheville, NC, December 1993.		
Wed., Apr. 14	19	L		**Networks 4**: Reliable messages. 3-way handshake and clock implementations. TCP as a form of reliable messages.		Early interim reports
			26	Paper: Butler Lampson, Reliable messages and connection establishment. In *Distributed Systems*, ed. S. Mullender, Addison-Wesley, 1993, pp 251-281.		
Mon., Apr. 19				Patriot's Day, no class.		
Wed., Apr. 21	20	R		**Distributed transactions**: Commit as a consensus problem. Two-phase commit. Optimizations.		Late interim reports
			27	Distributed transactions		
Mon., Apr. 26	21	L		**Replication and availability**: Implementing replicated state machines using consensus. Applications to replicated storage.		
			28	Replication		
			29	Paper: Jim Gray and Andreas Reuter, Fault tolerance, in *Transaction Processing: Concepts and Techniques*, Morgan Kaufmann, 1993, pp 93-156.		
Wed., Apr. 28	22			**Early project presentations**		
Mon., May 3	23			**Early project presentations**		
Wed., May 5	24	V		**Caching**: Maintaining coherent memory. Broadcast (snoopy) and directory protocols. Examples: multiprocessors, distributed shared memory, distributed file systems.		
			30	Concurrent caching		
Mon., May 10	25			**Late project presentations**		
Wed., May 12	26			**Late project presentations**		
Fri., May 14				**Final reports due**		
May 17-21				Finals week		