15-440: Distributed Systems
Syllabus

School of Computer Science
Carnegie Mellon University, Qatar
Fall 2012

1 Overview

Title: Distributed Systems
Units: 12 units
Pre-requisites: A grade of "C" or better in 15-213, Introduction to Computer Systems
Lectures: Monday and Wednesday, 2:30 – 3:50 PM, Room 2049
Recitation: Thursday, Time: TBA, Room: TBA
Webpage: http://www.qatar.cmu.edu/~msakr/15440-f12/

Description:

15-440 is an introductory course in distributed systems. The emphasis will be on the techniques for creating functional, usable, and high-performance distributed systems. To make the issues more concrete, the class includes several multi-week projects requiring significant design and implementation.

The goals of this course are twofold: First, students will gain an understanding of the principles and techniques behind the design of distributed systems, such as locking, concurrency, scheduling, and communication across networks. Second, students will gain practical experience in designing, implementing, and debugging real distributed systems.

The major themes this course will teach include process distribution and communication, data distribution, scheduling, concurrency, resource sharing, synchronization, naming, abstraction and modularity, failure handling, protection from accidental and malicious harm, distributed programming models, distributed file systems, virtualization, and the use of instrumentation, monitoring and debugging tools in problem solving. As the creation and management of software systems is a fundamental goal of any undergraduate systems course, students will design, implement, and debug large programming projects. Students will learn the design and implementation of today's popular distributed system paradigms, such as Google File System and MapReduce.

Instructors:

Majd F. Sakr
msakr@qatar.cmu.edu, M1016, 4454-8625, Office hours: Tuesday, 3:00 – 4:00 PM

Mohammad Hammoud
mhhammou@qatar.cmu.edu, M1013, 4454-8506, Office Hours: Thursday, 11AM – 12PM

2 Objectives

Distributed Systems combine the computational power of multiple computers to solve complex problems. The individual computers in a distributed system are typically spread over wide geographies, and possess heterogeneous processor and operating system architectures. Hence, an important challenge in distributed systems is to design system models, algorithms and protocols that allow computers to communicate and coordinate their actions to solve a problem.

Our aim in this course is to introduce you to the area of distributed systems. Y*ou will examine and analyze how a set of connected computers can form a functional, usable and high-performance distributed system.*

The course has three goals:

- To learn the principles, architectures, algorithms and programming models used in distributed systems.
- To examine state-of-the-art distributed systems, such as Google File System.
- To design and implement sample distributed systems.

Through these objectives, the course will transform your computational thinking from designing applications for a single computer system, towards that of distributed systems.

3 Learning Outcomes

The primary learning outcome of the course is two-fold:

1. *Students will identify the core concepts of distributed systems: the way in which several machines orchestrate to **correctly** solve problems in an **efficient**, **reliable** and **scalable** way.*
2. *Students will examine how existing systems have **applied the concepts** of distributed systems in designing large systems, and will additionally apply these concepts to develop sample systems.*

3.1 Understanding the Core Concepts of Distributed Systems:

Students will learn the core concepts underlying distributed systems designs. They will understand the system constraints, trade-offs and techniques in distributed systems to best serve the computing needs for different types of data and applications. Specifically, they will learn the following concepts:

- *Access and location transparency*
- *Parallelization of tasks*
- *Fault-tolerance*
- *Security*

3.1.1 Access and location transparency

Hiding the details of machines and exposing the capabilities is one of the first steps to design distributed systems that scale and penetrate economies and masses to utilize their power. For example, in the Internet, which is a successful distributed system, a simple browser interface will allow you to explore information scattered over wide-geographies. In this course, students will examine how to **abstract** locations, replication,

sharing and failure of resources that may reside at different physical places.

Students will learn the following topics:

- **Processes and Communication:** Students will explain and contrast the communication mechanisms between different processes and systems.
- **Naming:** Students will identify why entities and resources in distributed systems should be named, and examine the naming conventions and name resolution mechanisms.

3.1.2 Parallelization of tasks

Traditional algorithms that work on a single processor are inefficient – or even fail to work – in a system where multiple machines are working in parallel. In distributed systems, problems/jobs can be solved using parallelization. Generally a job is split into multiple tasks, and each task is executed concurrently with other tasks on a different machine. The tasks may access common resources, such as the data contained in a single file. As such, two main challenges emerge. The first challenge is to ensure that concurrently running tasks coordinate and synchronize to achieve a common goal. The second challenge is to replicate and place resources among multiple computers such that concurrently running tasks can access resources efficiently.

Students will study the following topics:

- **Concurrency and Synchronization:** Students will identify issues on how to coordinate and synchronize multiple tasks in a distributed system.
- **Consistency and Replication:** Students will identify how replication of resources improves performance and scalability in distributed systems, and examine algorithms that maintain consistent copies of replicas.

3.1.3 Fault-tolerance

In a distributed system with many computers, a failure of a single or a part of the computer is very likely. If such a system failure is not avoided or recovered from, the whole system might halt, resulting in a fragile and unpredictable system. *Students will identify the issues dealing with avoiding and recovering from failures*, a concept referred to as fault-tolerance, in distributed systems.

3.1.4 Security

In distributed systems, computers that solve your problem may not be under your administrative control; you do not own – sometimes, even know – where your program is running on a big connected set of computers. This makes a distributed system vulnerable to security and privacy related issues. *Students will learn the common security issues in distributed systems and mechanisms to secure the system*.

3.2 Practical Application of the State-of-the-Art Distributed Systems:

Students will also learn how to apply principles of distributed systems in a real-world setting. Specifically, they will learn the following topics:

- **Programming Models:** Students will learn the programming models that are commonly adopted in distributed systems. These programming models allow developers to easily program jobs that can be solved in distributed systems, while ensuring correctness and efficiency.

- **Distributed File Systems:** Students will learn about a file being split and stored anywhere in the distributed system, yet can be accessed transparently as a local file. They will examine how to apply distributed system principles in ensuring transparency, consistency and fault tolerance in distributed file systems.

- **Virtualization:** Students will learn the concept of *system* virtualization, where a state of a computer is abstracted from the underlying hardware. This allows transporting a virtual computer from one machine to a completely different machine, which may be physically present on other end of the world.

4 Textbooks

The primary textbooks for this course are:

- Andrew S. Tannenbaum and Maarten Van Steen, "***Distributed Systems: Principles and Paradigms"***, Second Edition, Pearson, 2007.
- George Coulouris, Jean Dollimore, Tim Kindberg, and Gordon Blair, "***Distributed Systems: Concepts and Design"****, Fifth Edition, Addison Wesley, 2011.*
- James E. Smith, and Ravi Nair, "***Virtual Machines: Versatile Platforms for Systems and Processes****", First Edition, Morgan Kaufmann, 2005.*

In addition, we recommend the following text books:

- Randal E. Bryant and David R. O'Hallaron, "***Computer Systems: A Programmer's Perspective****", Prentice Hall, 2003.*
- Tom White, "***Hadoop: The Definitive Guide****", Second Edition, O'Reilly Media, 2010.*

We have several reference books in the library covering most of the topics of the course. We will also be reading tutorials, journals and conference publications on the subject.

5 Course Organization

Students' participation in the course will involve five forms of activities:

- Attending and participating in the lectures and recitations
- Assignments (including writing and reading assignments)
- Projects
- Exams and quizzes
- In-class discussions

6 Getting Help

For urgent communication with the teaching staff, it is best to send an email (preferred) or phone. If you want to talk to an instructor in person, remember that our posted office hours are merely nominal times when we guarantee that we will be in our offices. You are always

welcome to visit us outside of our office hours if you need help or want to talk about the course.

We ask that you follow a few simple guidelines. Prof. Sakr and Dr. Hammoud normally work with their office doors open. Whenever their office doors are open, they welcome visits from students. However, if their office doors are closed, this means they are busy with meetings or phone calls and should not be disturbed.

We will use the course web-page as the central repository for all information about the class. Using the web-page, you can:

1. Obtain copies of any handouts or assignments. This is especially useful if you miss class or you lose your copy.
2. Find links to any electronic data you need for your assignments.
3. Read clarifications and changes made to any assignments, schedules, or policies.
4. Provide healthy feedback about the course.

You can use a mailing list specific for the class (instructions on this mailing list will be provided in class and on the course webpage) to post messages, make queries about the course, specific projects, or exams. The messages on this mailing list will be distributed to all the students and the instructors of the course.

7 Policies

Working Alone on Assignments/Projects

Assignments/projects that are assigned to single students should be performed individually. Some projects may be group projects, in which case it will be notified earlier.

Working on Group Assignments/Projects

Some assignments/projects are collectively performed by a group of students. A student can be in only one group. The maximum and minimum number of students in a group will be announced earlier. On such assignments/projects, you can collaborate only among your team members.

Handing in Assignments/Projects

All assignments/projects are due at 11:59 PM (one minute before midnight) on the specified due date. All hand-ins are electronic using CMU Autolab. Instructions on using Autolab will be provided in class and on the course webpage.

Making up Exams, Assignments and Projects

Missed exams, assignments and projects can be made up on a case by case basis, but only if you make prior arrangements with Prof. Sakr. However, you should have a good reason for doing so. You need a written consent from Prof. Sakr for making up exams, assignments or projects. It is your responsibility to get your projects done on time. Be sure to work far enough in advance to avoid unexpected problems, such as illness, unreliable or overloaded computer systems, etc.

Appealing Grades

After each assignment, exam and/or project is graded, you have seven calendar days to appeal your grade. All your appeals should be provided in writing. If you are still not satisfied, please come and visit Prof. Sakr. If you have questions about an exam grade, please visit Prof. Sakr directly.

8 Assessment

Final Grade Assignment and Assessment methods

Each student will receive a numeric score for the course, based on a weighted average of the following:

1. **Projects:** The projects will count a combined total of 45% of your score. There are 4 projects throughout the course. The first three projects are worth 10% each and the last is worth 15%. The last two projects (combined) will involve a presentation and a paper. Take into account that small differences in scores can make the difference between two letter grades.

You are encouraged to submit the projects on time. For all projects except the final one, the following rules apply. If you submit one day late, there will be deducted 25% of the project score as a penalty. If you are two days late, 50% will be deducted. The project will not be graded (and you will receive a zero score) if you are more than two days late. However, there is a **grace-days quota** for projects; you are given **3 grace days** for all projects (except the final project). You can use the grace days as needed. For example, you can submit your project 1, three days late and still not get any penalty. Your penalty starts from 4^{th} day after the deadline if you use your grace days. However, since you have used up all your grace days from your quota, you do not have any grace days for other projects. Plan how to utilize your grace day quota judiciously. For a team project, we deduct one grace day from each student if the team submits the project one day late. Hence, make sure that everyone in your team has 'x' grace days left if you want to submit the project 'x' days late.

Note that the final project is unique. You cannot use grace days for it. There will not be any penalty system for this project either. If you are one day late in submitting the final project, it will not be graded (and you will receive a zero score).

2. **Exams:** There will be two in-class exams – mid-term and final – that count for 25% of the grade. Mid-term will count for 10%, and final for 15% of the overall grade.

3. **Problem Solving Assignments:** There will be 4 written assignments that will test students on problem analysis and solving skills. These assignments will carry an overall score of 10% of your total score.

4. **Quizzes:** There will be 10 quizzes in the class or the recitation, which will account for 15% of your grade. There will be a quiz per topic that tests your understanding in the topic covered.

5. **Class/Recitation Participation and Attendance:** Your attendance and participation in the different discussions held in the class and the recitation will account towards 5% of your final grade.

Type	#	Weight
Projects	4	45%
Exams	2	25%
Problem Solving Assignments	4	10%
Quizzes	10	15%
Class/Recitation Participation and Attendance	43	5%

Grades for the course will be determined by absolute standards. The total score will be plotted as a histogram. Cutoff points are determined by examining the quality of work by students on the borderlines. Individual cases, especially those near the cutoff points may be adjusted upward or downward based on factors such as attendance, class participation, improvement observed throughout the course, exam performance, and special circumstances.

9 Cheating

Each project must be the sole work of the student turning it in, except for possible group projects. Projects will be closely monitored by automatic cheat checkers, and students may be asked to explain any suspicious similarities with any piece of code available. The following are guidelines on what collaboration is authorized and what is not:

What is cheating?

1. Sharing code or other electronic files: either by copying, retyping, looking at, or supplying a copy of a file.
2. Sharing written assignments: Looking at, copying, or supplying an assignment.

What is NOT cheating?

1. Clarifying ambiguities or vague points in class handouts.
2. Helping others use the computer systems, networks, compilers, debuggers, profilers, or other system facilities.
3. Helping others with high-level design issues.
4. Helping others debug their codes.

Cheating in group projects will also be strictly monitored and penalized (similar to cheating in individual exams, assignments or projects). Be aware of what constitutes cheating (and what does not) while interacting with students in other groups; same rules of cheating as above apply when collaborating between two or more groups. You cannot share or use written assignments, code, and other electronic files from students in other groups. If you are unsure, ask the teaching staff.

Be sure to store your work in protected directories. The penalty for cheating is severe, and might jeopardize your career – cheating is not worth the trouble. By cheating in the course, you are cheating yourself; the worst outcome of cheating is missing an opportunity to learn. In addition, you will be removed from the course with a failing grade. We also place a record of the incident in the student's permanent record.

10 Class Schedule

Table 1 shows the tentative schedule for the class. The schedule also indicates the project activities. Any changes will be announced on the class distribution list. An updated schedule will be maintained on the class webpage.

Week	Session	Date	Topic	Teaching Method	Reading list	Projects	Prob. Solving Assignment
1	1	3 Sep	Adiministrivia and Introduction	Lecture	Syllabus	Start P1	
	2	5 Sep	Introduction to Distributed Systems	Lecture	C1, T1		Start PS1
	3	6 Sep	Case study: Java Socket programming, RMI	Recitation	Notes about Socket programming and RMI		
2	4	10 Sep	Distributed System Architecture, Introduction to Networking	Lecture	C.2.1, C2.2, C2.3 (except 2.3.3), C3.1, C3.2		
	5	12 Sep	Networking – Layering, Switching, Routing, Congestion Control	Lecture	C3.3, C3.4	Design Report P1	
	6	13 Sep	Design of P1	Recitation			
3	7	17 Sep	Inter-Process Communication – Socket, RPC, Message-passing and multi-cast	Lecture	T4.2 – T4.6		
	8	19 Sep	Naming – Flat, structured and attribute-based	Lecture	T5.1, T5.2, T5.3, T5.4 .1, T5.4.2		
	9	20 Sep	Case study: Google protocol buffers and publish-subscribe	Recitation			End PS1
4	10	24 Sep	Synchronization – Physical clocks, Logical clocks, Vector clocks	Lecture	T6.1, T6.2		Start PS2
	11	26 Sep	Synchronization – Mutual exclusion, election algorithms	Lecture	T6.3 – T6.6		
	12	27 Sep	Case study: Google Chubby	Recitation			
5	13	1 Oct	Synchronization – Transactions and concurrency control	Lecture	C16	End P1/Start P2	
	14	3 Oct	Consistency & Replication	Lecture	T7		
	15	4 Oct	Case study: Replication in GFS, Design of P2	Recitation			
6	16	8 Oct	Consistency & Replication	Lecture	T7	Design Report P2	
	17	10 Oct	Consistency & Replication	Lecture	T7		
	18	11 Oct	Design of P2	Recitation			End PS2
7	19	15 Oct	Fault Tolerance	Lecture	T8		
	20	17 Oct	Fault Tolerance	Lecture	T8		
	21	18 Oct	Programming on Reliable Communication	Recitation		End P2	
8	22	22 Oct	Fault Tolerance	Lecture	T8	Start P3	
	23	**24 Oct**	**Midterm**	**Exam1**			Start PS3
	24	25 Oct	Design of P3 & Developing MPI programs	Recitation			
9		28 Oct	Eid Al-Adha Break; No Classes				
10	25	5 Nov	Programming Models	Lecture	Notes on MPI, Shared memory, W1, W2, W6		
	26	7 Nov	Programming Models	Lecture	Notes on MPI, Shared memory, W1, W2, W6		
	27	8 Nov	Developing MPI programs	Recitation			

11	28	12 Nov	Programming Models	Lecture	Notes on MPI, Shared memory, W1, W2, W6		End PS3
	29	14 Nov	Distributed File Systems	Lecture	T11	End P3	Start PS4
	30	15 Nov	Developing MapReduce Programs				
12	31	19 Nov	Distributed File Systems	Lecture	T11	Start P4	
	32	21 Nov	Big Table: A Distributed Structured Storage System	Recorded Video Lecture	The BigTable paper		
	33	22 Nov	Design of P4 & Developing MapReduce Programs	Recitation			
13	34	26 Nov	Security	Lecture	T9		
	35	28 Nov	Security	Lecture	T9		
	36	29 Nov	Developing MapReduce programs	Recitation			
14	37	3 Dec	TBA	Guest Lecture			
	38	5 Dec	Virtualization	Lecture	S1, S8, S9		
	39	6 Dec	TBA	Recitation			
15	40	10 Dec	Virtualization	Lecture	S1, S8, S9		End PS4
	41	12 Dec	Virtualization	Lecture	S1, S8, S9		
	42	13 Dec	Presentation Session on P3 & P4	Presentations			
16	**43**	**17 Dec**	**Final**	**Exam2**		**End P4**	

Table 1: Tentative Time-Line of the Course.

Notations used in Table 1 are as given below:

- Assignments: **PS**=Problem Solving Assignments
- Reading list: **Cx(.y.z)** = Chapter x (Section y, subsection z) from Colouris textbook; similarly, **Tx(.y.z)** refers to chapters from Tannenbaum textbook, **Wx(.y.z)** refers to White textbook, and **Sx(.y.z)** refers to Smith textbook.
- Instructors: ***MFS***=*Majd Sakr,* ***MHH***=*Mohammad Hammoud* ***TBA***=*To Be Announced*