15-319/15619: CLOUD COMPUTING

COURSE DESCRIPTION & SYLLABUS

CARNEGIE MELLON UNIVERSITY
SPRING 2016

1. OVERVIEW

Title: Cloud Computing
Units: 15-319 is 9 units and 15-619 is 12 units.
Pre-requisites for undergraduate students: A "C" or better in 15-213.
Pre-requisites for graduate students: Knowledge of computer systems, programming and debugging, with a strong competency in at least one language (such as Java/Python), and the ability to pick up other languages as needed.

OLI Course: http://oli.cmu.edu **(**accessed through https://blackboard.andrew.cmu.edu**)**
The Project Zone: https://TheProject.Zone
Piazza: http://piazza.com/cmu/spring2016/1531915619/home

Recitation:

1. **Tuesday, 8:00 AM – 8:50 AM, GHC 4307 (Videotaped)**
2. **Thursday, 4:30 PM – 5:20 PM, GHC 4307 (First few weeks and when needed)**

Teaching Staff:

Prof. Majd F. Sakr
msakr@cs.cmu.edu
GHC 7006, +1-412-268-1161
Office hours: Tuesday, 3-4pm (Pittsburgh)

TAs in Pittsburgh typically hold office hours in GHC 5th Floor Teaching Commons. The TA office hours are posted on Piazza:

- Jinhong Chen < jinhongc@andrew.cmu.edu>
- Chaskiel Grundman < cg2v@andrew.cmu.edu>
- Xingchi Jin < xingchij@andrew.cmu.edu>
- Mrigesh Kalvani <mkalvai@andrew.cmu.edu>
- Wei Luo <weiluo@andrew.cmu.edu>
- Mohammed Suhail Rehman <suhailr@andrew.cmu.edu>
- Lei Sun <leis1@andrew.cmu.edu>
- Chrysanthi Vandera <cvandera@andrew.cmu.edu>
- Yang Wang < yangw3@andrew.cmu.edu>
- Yang Wu <yangwu@andrew.cmu.edu>
- Ruirui Xiang <ruiruix@andrew.cmu.edu>
- Di Xiao <dxiao1@andrew.cmu.edu>
- Kevin Xu <klx@andrew.cmu.edu>
- Mengyu Yang <mengyuy@andrew.cmu.edu>
- Liruoyang Yu <liruoyay@andrew.cmu.edu>
- Yiming Zang <yzang@andrew.cmu.edu>
- Yang Zhang <yangz4@andrew.cmu.edu>
- Ying Zhang <yingzha3@andrew.cmu.edu>

2. COURSE DESCRIPTION

This on-line course gives students an overview of the field of Cloud Computing, its enabling technologies, main building blocks, and hands-on experience through projects utilizing public cloud infrastructures (Amazon Web Services (AWS) and Microsoft Azure). Cloud computing services are being adopted widely across a variety of organizations and in many domains. Simply, cloud computing is the delivery of computing as a service over a network, whereby distributed resources are rented, rather than owned, by an end user as a utility.

The course will introduce this domain and cover the topics of cloud infrastructures, virtualization, software defined networks and storage, cloud storage, and programming models. As an introduction, we will discuss the motivating factors, benefits and challenges of the cloud, as well as service models, service level agreements (SLAs), security, example cloud service providers and use cases. Modern data centers enable many of the economic and technological benefits of the cloud paradigm; hence, we will describe several concepts behind data center design and management and software deployment. Next, we will focus on virtualization as a key cloud technique for offering software, computation and storage services. We will study how CPU, memory and I/O resources are virtualized, with examples from Xen and VMWare, and present real use cases such as Amazon EC2. Within the same theme of virtualization, students will also be introduced to Software Defined Networks and Storage (SDN and SDS). Subsequently, students will learn about different cloud storage concepts including data distribution, durability, consistency and redundancy. We will discuss distributed file systems, NoSQL databases and object storage. HDFS, CephFS, HBASE, MongoDB, Cassandra, DynamoDB, S3, Swift and Ceph Object Gateway will be presented as case studies. Finally, students will understand the details of the MapReduce programming model and gain a broad overview of the Spark, GraphLab programming models as well as message queues and stream processing.

Students will work with Amazon Web Services and Microsoft Azure, use them to rent and provision compute resources and then program and deploy applications that run on these resources. Students will develop and evaluate scaling and load balancing solutions. In addition, students will work with cloud storage systems and learn to develop different applications in several programming paradigms. 15-619 students will have to complete an extra project which entails designing and implementing a complete web-service solution for querying big data. For the extra project, the student teams are evaluated based on the cost and performance of their web service.

3. COURSE OBJECTIVES

In this on-line course we plan to give students an overview of the field of Cloud Computing, and an in-depth study into its enabling technologies and main building blocks. Students will gain hands-on experience solving relevant problems through projects that will utilize existing public cloud tools. It is our objective that students will develop the skills needed to become a practitioner or carry out research projects in this domain. Specifically, the course has the following objectives:

Students will learn

1) the fundamental ideas behind Cloud Computing, the evolution of the paradigm, its applicability; benefits, as well as current and future challenges;
2) the basic ideas and principles in data center design; cloud management techniques and cloud software deployment considerations;
3) different CPU, memory and I/O virtualization techniques that serve in offering software, computation and storage services on the cloud; Software Defined Networks (SDN) and Software Defined Storage (SDS);
4) cloud storage technologies and relevant distributed file systems, NoSQL databases and object storage;
5) the variety of programming models and develop working experience in several of them.

4. LEARNING OUTCOMES

The primary learning outcomes of this course are five-fold. Students will be able to:

1) Explain the ***core concepts*** of the cloud computing paradigm: how and why this paradigm shift came about, the characteristics, advantages and challenges brought about by the various models and services in cloud computing.
2) Apply fundamental concepts in ***cloud infrastructures*** to understand the tradeoffs in power, efficiency and cost, and then study how to leverage and manage single and multiple datacenters to build and deploy cloud applications that are resilient, elastic and cost-efficient.
3) Discuss ***system, network and storage virtualization*** and outline their role in enabling the cloud computing system model.
4) Illustrate the fundamental concepts of ***cloud storage*** and demonstrate their use in storage systems such as Amazon S3 and HDFS.
5) Analyze various ***cloud programming models*** and apply them to solve problems on the cloud.

4.1. BASIC CONCEPTS

This module will provide a broad overview of cloud computing, its history, technology overview, benefits, risks and the economic motivation for it. Upon completion of this module, students will be able to:

4.1.1. Explain the concept of "cloud computing".
4.1.2. Briefly recall the recent history of cloud computing, illustrating its motivation and evolution.
4.1.3. List some of the enabling technologies in cloud computing and discuss their significance.
4.1.4. Discuss some of the advantages and disadvantages of the cloud paradigm.
4.1.5. Articulate the economic benefits as well as issues/risks of the cloud paradigm for businesses as well as cloud providers.
4.1.6. Associate the various layers in the cloud building blocks and differentiate cloud service models.
4.1.7. Define SLAs and SLOs and illustrate their importance in Cloud Computing.
4.1.8. Enumerate and explain various threats in cloud security.
4.1.9. List some of the common cloud providers and their associated cloud stacks and recall popular cloud use case scenarios.

4.2. CLOUD INFRASTRUCTURE

This module will provide a historical overview of data centers, along with design considerations. Students will learn to apply methods to evaluate data centers, cloud management techniques and software deployment considerations. Upon completion of this module, students will be able to:

4.2.1. Describe the evolution of data centers and outline the architecture of a modern data center.
4.2.2. Indicate design considerations and discuss their impact.
4.2.3. Demonstrate the ability to calculate various power requirements of a data center.
4.2.4. Recall challenges and requirements for a cloud-centric data center and how they differ from large, single-entity warehouse-scale computers.
4.2.5. Explain the cloud software stack and the role of each layer within it.
4.2.6. Identify the need for and techniques behind automation and orchestration of resources, as well as key scheduling considerations in the cloud.
4.2.7. Evaluate programming, deployment and failure considerations when programming the cloud.
4.2.8. Understand the implications of building a multi-tier cloud application to achieve resiliency and elasticity, and the latency implications of such applications.

4.2.9. Recall various cloud pricing models and their applicability to various business use cases.
4.2.10. Recall and describe cloud management techniques such as middleware, resource provisioning, metering, and orchestration.
4.2.11. Describe and evaluate different cloud software deployment considerations such as scaling strategies, load balancing, fault tolerance, accounting for tail latencies and optimizing for cost.

4.3. CLOUD RESOURCE MANAGEMENT

Students will learn how virtualization can allow software and hardware images (e.g., virtual machines) to run side-by-side on a single cloud data center while provided security, resource and failure isolations. They will understand how virtualization enables clouds to offer software, computation, and storage as services as well as attain agility and elasticity properties. We will discuss resource virtualization in detail and present multiple examples from Xen and VMware. Finally, we will present a real use case such as Amazon EC2. After finishing this unit students will be able to:

4.3.1. Identify major reasons for why virtualization is useful, especially on the cloud.
4.3.2. Explain different isolation types such as fault, resource, and security isolations provided by virtualization and utilized by the cloud.
4.3.3. Indicate how system complexity can be managed in terms of levels of abstractions and well-defined interfaces, and their applicability to virtualization and the cloud.
4.3.4. Define resource sharing as provided by virtualization and discuss how it can be offered in space and time via physical and logical partitioning.
4.3.5. Define virtualization and identify different virtual machine types such as process and system virtual machines.
4.3.6. Identify conditions for virtualizing CPUs, recognize the difference between full virtualization and paravirtualization, explain emulation as a major technique for CPU virtualization, and examine virtual CPU scheduling in Xen.
4.3.7. Outline the difference between classical OS virtual memory and system memory virtualization, explain the multiple levels of page mapping as imposed by memory virtualization, define memory over-commitment and illustrate VMWare memory ballooning as a reclamation technique for memory over-committed virtualized systems.
4.3.8. Explain how CPU and I/O devices can communicate with and without virtualization, identify the three main interfaces, system call, device driver and operation level at which I/O virtualization can be carried, and apply I/O virtualization to Xen.
4.3.9. Outline recent developments in software defined networking and software defined storage from the cloud computing perspective.

4.4. CLOUD STORAGE

This module will provide a broad overview of storage technologies and concepts of cloud storage. It will also provide a detailed study of Amazon S3, EBS and distributed file systems and databases. Students will be able to:

4.4.1. Describe the overall organization of data and storage.
4.4.2. List the various types of data within the data taxonomy and classify different data types within the data taxonomy.
4.4.3. Identify the problems of scale and management in big data. Discuss various storage abstractions.
4.4.4. Compare and contrast different types of file systems and discuss their design considerations. Compare and contrast Hadoop Distributed File System (HDFS) with Ceph File System (CephFS).
4.4.5. Compare and contrast different types of databases and discuss their design tradeoffs.

4.4.6. Discuss the concepts of cloud object storage. Enumerate the different types of block devices used in data storage.

4.5. PROGRAMMING MODELS

Students will be given an overview on a variety of cloud-based programming models. Students will understand the benefits and limitations of each so that they can assess applicability based on the problem domain. Students will gain working experience in one of these programming models. Upon completion of this module students will be able to:

4.5.1. Explain the fundamental aspects of parallel and distributed programming models.
4.5.2. Recall and contrast different cloud programming models (MapReduce, Spark, GraphLab and Spark Streaming).
4.5.3. Explain the main execution flow, scheduling and fault tolerance concepts in the MapReduce programming model.

5. COURSE ORGANIZATION

Your participation in the course will involve several forms of activity:

1. Reading the online coursework content for each unit on OLI.
2. Completing the unscored inline activities for each unit (Review activities on OLI).
3. Completing the graded checkpoint quizzes after each unit.
4. Complete projects which are performed using AWS and submitted through TheProject.Zone.
5. AssessMe AssessMents, unscored short quizzes to unlock subsequent project sections.
6. Complete a team project on building a complete web service.

Students should regularly check OLI to see when new content, projects or checkpoint quizzes are made available. Projects and Checkpoint quizzes must be completed by the due dates posted on OLI.

6. GETTING HELP

Students are encouraged to ask questions about content and projects through **Piazza**, where an online class portal has been created for this course. The course link for Piazza is:

http://piazza.com/cmu/spring2016/1531915619/home.

There is a weekly recitation (either at 8AM on Tuesdays in GHC 4307 or 4:30PM on Thursdays at GHC 4307) in Pittsburgh. The teaching staff will be on hand to highlight and discuss any major questions that have been posted to Piazza or by email. For urgent communication with the teaching staff, it is best to send an email.

We will use the course website as the basic portal for the class. The course content is entirely on OLI. The project write-ups, submission, scoreboard and grades are on TheProject.Zone. The checkpoint quizzes are on OLI. OLI can be reached through Blackboard. Announcements, discussions and questions are posted on Piazza.

7. POLICIES

WORKING ALONE ON PROJECTS

Projects that are assigned to single students should be performed individually.

HANDING IN PROJECTS

All assignments/projects are due at 11:59 PM EST (one minute before midnight) on the due dates specified on OLI or TheProject.Zone. All hand-ins are electronic, and use the OLI Checkpoint system and TheProject.Zone.

APPEALING GRADES

After each project phase is graded, you have seven calendar days to appeal your grade. All your appeals should be provided by email to Prof. Sakr.

8. ASSESSMENT

Inline activities ("Learn by Doing" and "Did I Get This"), which are available in most pages in the OLI course, are simple, non-graded activities to assess your comprehension of the material as you read through the course material. You are advised to complete all of the inline activities before proceeding through to the next page or module. If you missed many of the activities, it is recommended that you review the material again.

There are five units consisting of modules of content on OLI, each week has a Checkpoint Quiz that you must complete before the deadline posted on OLI. Each weekly Checkpoint Quiz will be worth ~2% of your total grade. It is your responsibility to ensure that the quiz is submitted prior to the deadline. You will have only a single attempt to complete each Checkpoint Quiz on OLI.

This course includes four individual projects. Each project consists of several project modules. Every week, a project module has to be completed based on the deadlines posted on TheProject.Zone. The write-up required to complete each project module is available on TheProject.Zone. Each module has a submission process that is specific to the project module that is due. It is the students' responsibility to make sure that all project work is completed and that the project module is submitted prior to the deadline. Students typically have multiple attempts to submit the project module on TheProject.Zone.

15-619 students have to complete a team-based multi-week project in parallel to the weekly Project Modules.

Type	Number	Weight
Content Checkpoint Quizzes (equal weight)	12	20%
Projects (equal weight)	4 (15-319) 5 (15-619)	80%
Total Grade		**100%**

9. CHEATING

We urge each student to carefully read the university policy on academic integrity, which outlines the policy on cheating, plagiarism or unauthorized assistance. It is the responsibility of each student to produce her/his own original academic work. Collaboration or assistance on academic work to be graded is not permitted unless explicitly authorized by the course instructor. Each unit checkpoint quiz or project module submitted must be the sole work of the student turning it in. Student work on AWS is logged, submitted work 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 any file. Copying any code from the internet (stackoverflow.com or github or others).
2. Copying answers to any checkpoint quiz from another individual, published or unpublished written sources, and electronic sources.
3. Collaborating with another student or another individual on checkpoint quizzes or project modules.
4. Sharing written work, looking at, copying, or supplying work from another individual, published or unpublished written sources, and electronic sources.
5. Collaboration in team projects is strictly limited to the members of the team.

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 code.

Cheating in projects will also be strictly monitored and penalized. Be aware of what constitutes cheating (and what does not) while interacting with students. You cannot share or use written code, and other electronic files from students. 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 simply 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. SYLLABUS

The course will be structured into the following units:

Unit #	Title	Modules and Description
1	**Introduction**	Definition and evolution of Cloud Computing Enabling Technologies, Service and Deployment Models Popular Cloud Stacks and Use Cases Benefits, Risks, and Challenges of Cloud Computing Economic Models and SLAs Topics in Cloud Security
2	**Cloud Infrastructure**	Historical Perspective of Data Centers Datacenter Components: IT Equipment and Facilities Design Considerations: Requirements, Power, Efficiency, & Redundancy Power Calculations, PUE and Challenges in Cloud Data Centers Cloud Management and Cloud Software Deployment Considerations
3	**Virtualization**	Virtualization (CPU, Memory, I/O) Case Study: Amazon EC2 Software Defined Networks (SDN) Software Defined Storage (SDS)
4	**Cloud Storage**	Introduction to Storage Systems Cloud Storage Concepts Distributed File Systems (HDFS, Ceph FS) Cloud Databases (HBase, MongoDB, Cassandra, DynamoDB) Cloud Object Storage (Amazon S3, OpenStack Swift, Ceph)
6	**Programming Models**	Distributed Programming for the Cloud Data-Parallel Analytics with Hadoop MapReduce (YARN)

	Iterative Data-Parallel Analytics with Apache Spark Graph-Parallel Analytics with GraphLab 2.0 (PowerGraph)

11. PROJECTS

The programming projects in this course will be geared towards providing hands-on experience with various cloud technologies. Students will learn to develop all projects using various public cloud services (primarily Amazon Web Services (AWS) and some work on Microsoft Azure). Students will be given a budget for cloud resources for each project and are expected to work within the budget otherwise, they risk being penalized.

11.1. PROJECT 1: BIG DATA ANALYSIS

Students will work with Amazon AWS and provision their first compute resources. Students will setup AWS accounts, work with provisioning management software and launch instances on Amazon EC2. Students will learn the benefits and tradeoffs of running programs in parallel, using AWS EMR, versus sequential on a large dataset. Students will have to solve a problem using resources provisioned in Amazon within particular cost constraints.

11.2. PROJECT 2: CLOUD ELASTICITY

In this project, students will learn about cloud elasticity. Students will be first tasked with developing their own elastic services for a dynamically changing load scenario using AWS APIs. Students will then work with the Elastic Load Balancing and Auto Scaling services at AWS to mitigate varying loads on the server. Students will also build their own load balancer and evaluate different caching mechanisms. This project helps students explore critical cloud concepts like rapid on-demand scalability, fault detection and tolerance, and performing cost benefit analyses to maximize profit.

11.3. PROJECT 3: CLOUD STORAGE

Using AWS resources, students will work on cloud storage technologies to evaluate their capabilities and limitations, each week with a new workload and a storage system. Students begin by exploring the limitations of traditional filesystems, and then compare them to relational databases (MySQL) and NoSQL databases (HBase). Next, students explore sharding and replication of a simple key-value store. Consistency problems in geo-replicated key-value stores are dealt with. Students will build a social network timeline using heterogeneous back-end storage systems. Projects will cover several storage systems, including low-latency KV stores, NoSQL databases, and in-memory databases (examples include Apache HBase, Amazon RDS, DynamoDB and others). Finally, students will perform Data Warehousing tasks using Online Line Analytics Processing (OLAP) systems such as AWS Redshift and Impala.

11.4. PROJECT 4: PROGRAMMING MODELS

In this project, students will work on developing applications using the MapReduce, Spark, GraphLab and Samza programming models. Students will write their own MapReduce code using Apache Hadoop and provision instances on Amazon EC2 to run them in order to build their own input text predictor, similar to Google Instant. Students will build the input text predictor from a large text corpus by generating a list of n-grams, building a statistical language model using the n-grams, and creating a user interface. Students will also be introduced to iterative programming models by implementing the PageRank algorithm on Apache Spark and GraphLab. Finally, students will learn to deal with streaming data to perform real-time processing of multiple data streams.

11.5. 15-619 TEAM PROJECT: TWITTER ANALYTICS WEB SERVICE

Students will work in teams to design and implement a complete web-service that uses the REST interface to respond to queries that require running an analytics job on a large (1TB) twitter data set which is stored in a database (MySQL, HBASE, etc.). In this team project, student teams are expected to use different tools and services to achieve build a performing web-service that meets the requirements. The students' web-services are evaluated through a load generator for a fixed time period (several hours) by measuring the cost of AWS cloud resources used and their system's performance (throughput). There is an upper bound on the budget which could cause students to be disqualified. Students are evaluated based on how their service performs compared to a baseline.

12. SCHEDULE

The tentative schedule is as follows (specific deadlines are posted on OLI and TheProject.Zone):

Week	Monday	OLI Content	Projects	15-619 Project	Quizzes
1	1/11/2016	Unit 1, Module 1	Primers/P0 (Jan 17)		
2	1/18/2016	Unit 1, Module 2	P1.1 (Jan 24)		Q1 (Jan 22)
3	1/25/2016	Unit 2, Module 3, 4	P1.2 (Jan 31)		Q2 (Jan 29)
4	2/1/2016	Unit 2, Module 5, 6	P2.1 (Feb 7)		Q3 (Feb 5)
5	2/8/2016	Unit 3, Module 7, 8, 9	P2.2 (Feb 14)		Q4 (Feb 12)
6	2/15/2016	Unit 3, Module 10, 11, 12	P2.3 (Feb 21)		Q5 (Feb 19)
7	2/22/2016	Unit 3, Module 13	P3.1 (Feb 28)	Project Out (Feb 24)	Q6 (Feb 26)
8	2/29/2016	Unit 4, Module 14	P3.2 (Mar 6)		Q7 (Mar 3)
9	3/7/2016	Spring Break			
10	3/14/2016	Unit 4, Module 15	P3.3 (Mar 20)	Phase 1 Due (Mar 16)	Q8 (Mar 18)
11	3/21/2016	Unit 4, Module 16, 17	P3.4 (Mar 27)		Q9 (Mar 25)
12	3/28/2016	Unit 5, Module 18	P3.5 (Apr 3)	Phase 2 Due (Mar 30)	Q10 (Apr 1)
13	4/4/2016	Unit 5, Module 19	P4.1 (Apr 10)		Q11 (Apr 8)
14	4/11/2016	Unit 5, Module 20, 21	P4.2 (Apr 17)	Phase 3 Due (Apr 13)	
15	4/18/2016	Unit 5, Module 20, 21	P4.3 (Apr 24)		Q12 (Apr 22)