#  Introduction to Cloud Computing – July 4th, 2011

## What is cloud computing to you? (Answers from Class)

* Corporate buzz word
* Distributed system
* A computer on a network
* Renting computing resources
* Elastic capability
* Virtualization application
* Scalable

Many people draw comparisons between cloud computing and **grid computing**[1]

### The major differences [2]:

* Higher inter-node bandwidth in cloud computing
	+ Critical for distribution/synchronization of large quantities of data
* Virtualization
	+ Virtualization of hardware in CC
	+ Virtualization of data in CC
		- User-facing representation of data is independent of data storage implementation
		- E.g. Amazon stores stuff over multiple nodes, you do not care how it stores it, just that it is available

Similarities between cloud computing and grid computing

* Increased reliability of infrastructure, hardware, and data
* Increased flexibility
* Reduces cost

**Utility computing**[3] comprises charging for the use of packaged resources

### Does grid computing + utility computing = cloud computing?

 Yes and no (check slides - page 6 ) -- the answer depends on the definition of cloud computing, which is still under debate [5]

There is a (relatively) universally accepted definition of cloud computing by NIST (see the slide) [6] [11]

### What is a snapshot of a machine?

* Historical point of time, makes a copy of installed services, programs

### Why do we take a snapshot of a system?

* Allows administrator to easily mass produce a resource of a specific configuration
* Provides a backup of a "healthy" state/template to build other configurations from

### Additional notes on the NIST-def'n of cloud computing:

* On-demand self-service[7]
	+ Must automate the delegation of resources - this is critical to self-service since humans have a higher probability of committing an error
* Digital rights management is better than automation.
	+ We have technology that force legality for digital rights.
	+ Other areas (e.g law) has human that govern the rules (e.g judges)
* Ubiquitous network access (Application for multiple devices)
	+ Why support heterogeneous platforms:
		- Greater market penetration (get more people can access it)
		- Separation of concerns -- must isolate interface (for various devices) from the business logic
	+ Must be mindful of performance capabilities of different types of clients
		- E.g. phones has less bandwidth than laptops
* Resource pooling [8]
	+ Resources are shared by the users
		- Mutli-tenancy
	+ How do we partition data to isolate each organization's data from the others?
		- Table partitioning
	+ Has implications for performance
	+ Has implications for scalability:
		- Easy to add (set null to everyone)
		- Not easy to remove/reconfigure data
	+ Security is accomplished by the service provider
		- Service provider has access to data
* Rapid elasticity
	+ Reduce resources to:
		- Save power (can turn off unused resources)
		- Free up resources for other users
	+ However there are difficulties in achieving this elasticity:
		- Incur overhead when increasing/reducing available resources
	+ Should remove state from computational resources (e.g. offset state to a data store so resources can be turned off without loss of data)
	+ Appearance of “unlimited” resource - but can crash from time to time.
	+ Resources are shared, you can scale up while someone scales down
* Measured service
	+ Typically do not measure money
* Access
	+ No universally well-defined protocol, but all are based on the foundation of HTTP
* Virtualization (see notes)

### Software-as-a-Service (SaaS):

* Service determined by the provider
	+ User does not decide which version to use

E.g You cannot switch to previous version of Google Mail/Docs

* SaaS applications are "net native" applications[10]
	+ Disadvantages:
		- Since provider is in charge of service and data, very tight coupling between the user and the Cloud Service Provider (CSP)
		- Security is heavily controlled by the CSP (can be both good and bad). User might need to add own security on top of the one provided.
		- Bandwidth can be bad
	+ Advantages
		- Security: If there is a fix to a bug fixes are passed to everyone instantly.

### Platform-as-a-Service (PaaS)

* User creates an application on top of provided platform/environment governed by set of provider-specified rules
	+ Main example of a rule is language specification
	+ Service Provider performs resource management

### Infrastructure-as-a-Service (Iaas)

* Provider gives users access to computing resources for processing/storage/whatever
* There is an illusion that the resources are entirely under your control
	+ The Service Provider still has control and access to the resources

### Everything-as-a-Service (XaaS)

* Provide capability to write multi-tiered application (i.e. applications on applications)

# Questions

**Give an example of a SPI Service that is not flexible. Why is it not flexible?**

Facebook. There is only one view of the Facebook homepage. When facebook decides to change the view or add/remove features you are unable to revert back to the old one. All your interactions are going through their current design and view.

**Why would you like to exclude human interaction from On-demand self-service?**

Any human process introduces change of errors. By excluding the human interaction the fault rate decreases. Also human interactions are slower than automated processes hence improvement both on speed and quality.

# References:

[1] <http://en.wikipedia.org/wiki/Grid_computing>

[2] Advanced Topics in Computer Systems: Cloud Computing and Management – 2011(Raouf Boutaba)

[3] <http://en.wikipedia.org/wiki/Utility_computing>

[4] Cloud computing and grid computing 360-degree compared I. Foster et al. 2008

[5] “Twenty Experts Define Cloud Computing”, SYS-CON Media Inc 2008.

[6] <http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc>

[7] Draft NIST Working Definition of Cloud Computing – 2009, Peter Mell and Tim Grance

[8] <http://msdn.microsoft.com/en-us/library/aa479086.aspx>

[9] Visualizing the Boundaries of Control in the Cloud. Dec 2009.

http://kscottmorrison.com/2009/12/01/visualizing-the-boundaries-of-control-in-the-cloud/

[10] <http://www.crn.com/news/applications-os/18831096/how-net-native-software-as-services-differs-from-packaged-apps.htm>

[11] <http://csrc.nist.gov/publications/drafts/800-145/Draft-SP-800-145_cloud-definition.pdf>