

# ENGR E-516: Engineering Cloud Computing

## Introduction to Cloud Computing

**Prateek Sharma**

Week 1

## What is this course about?

From last time..

This course is about how the largest computing systems are built and run, and how you can use them effectively.

No Pressure  
Pressure Sensitive

## Course Overview

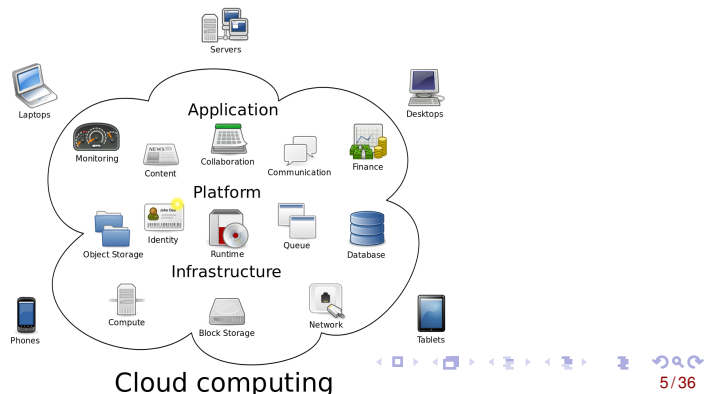
- 1 Fundamentals of operating systems, computer networks, and distributed systems
- 2 Cloud applications [network services, data processing]
- 3 Virtualization
- 4 Distributed management of computing resources
- 5 Current and future cloud ecosystems and services
- 6 Distributed data storage: costs, caching, and consistency

# Introduction to Cloud Computing

# What is Cloud Computing?

Multiple perspectives

- 1 A way for end-devices to access computational and storage resources over the internet
- 2 Computing as a utility
- 3 Cloud  $\equiv$  infinite, elastic computer
- 4 Programming platform for complex applications
  - A way to deploy applications as distributed networked services



Why the word “cloud?”

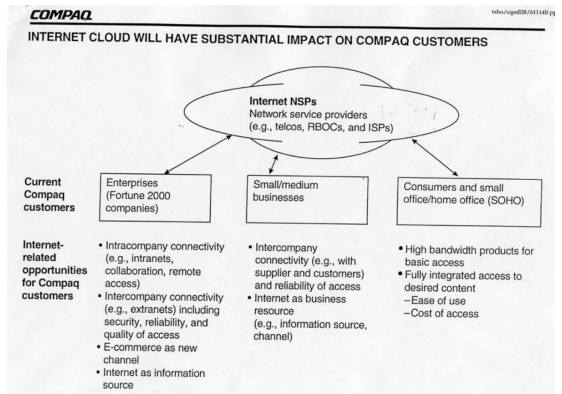
Sun microsystems (mid 90s): The network is the computer

# Cloud Computing Origins

- Etymology: Networks often represented as “clouds” in architecture diagrams
- Compaq 1996 memo: “Internet Solutions Division Strategy For Cloud Computing”

Why the word “cloud?”

Sun microsystems (mid 90s): The network is the computer



# Utility Computing

Imagine having to live without standardized utilities.

Can computing be a utility just like electricity, water, etc.?

- Where's the electricity coming from?
- What are the sources of generation?
- Can ignore complex interaction between powerplants, grid, and utility companies.
- Can ignore transformers, phases, ...
- From the consumer perspective, just a flick of the switch.
- Appliances "dont care".





## What being a utility entails

- Standardization of client endpoints [plugs]
- Resource provider and consumer agree on resources provided [voltage, frequency, max current draw]

Other issues:

- Utilities must match supply and demand
- Billing
- Time of day pricing [lower during the night]
- Demand-response pricing [higher prices during peak-loads to discourage excess use]
- Users/appliances do not worry about any of these things!

## What this course is about

- How cloud platforms and services work
- How they can be used effectively

User centric view:

- How to build and deploy applications on the cloud
- How to improve performance, cost

## Fungibility of Computing Resources

- Fungible goods can be easily interchanged
- Example: gasoline/petrol
- Energy → Computing → Applications → \$\$\$

Can computing be made fungible?

### Spoiler

Yes, mostly, using different abstractions.

# Elastic Computing

- Can view the cloud as “one giant computer”
- That can meet different-sized computing demands
  - Use 1 CPU for 1 year,  $10^4$  CPUs for 1 hour
- “On-demand”: Near-instantaneous request-use-discard process
- Utility computing → Pay for what you use
- Different sized computing resources can be leased at various timescales
  - “Rent” a server for a minute, hour, day, year, ...

# Provisioning of resources

“Pre-cloud”:

- 1 Build application
- 2 Request hardware (servers, disks) to run application
  - Go through enterprise, university purchasing and procurement
  - Can take several days, even months
- 3 Deploy application
- 4 Application becomes popular, needs more servers. Goto step 2

Cloud provisioning:

- `server = request_instance(size, number)`
- ...wait 30 seconds
- `ssh(server) // deploy application`

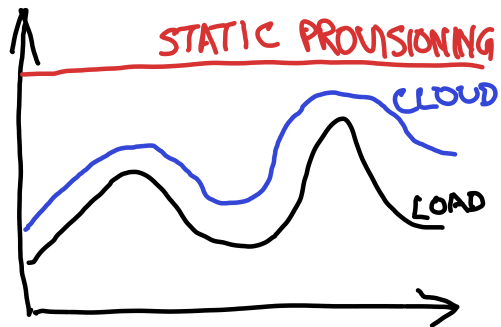
## Analogy

Building a house vs. staying in a hotel

# Elastic Provisioning

Elasticity enables many use-cases:

- Popular applications can “scale up” to meet user demands
- Many applications see fluctuating load
  - More people check email during day vs. night
- Cloud computing enables *dynamic* allocation of resources



- Consider a service with multiple users, such as Canvas
- Multiple software components and tiers:
  - Web-server for serving pages on `iu.canvas.edu`
  - Application-tier (ex: compute grades based on scores)
  - Database to store student data
- Different components run on different servers
- Servers have a limit on the number of operations/users they can support
  - Example: A single web-server may be limited to 10 requests/second

## Clouds as application platforms

- We all “use” the cloud through cloud-based applications
- Email [gmail, . . .]
- Data storage [dropbox, box, . . .]
- Various web-services like Slack, Uber, . . .

*What does it mean for an application to be “on the cloud”?*



## Local vs. Cloud Application : Email

Local email client such as Thunderbird:

- Retrieve messages from mail server periodically
- Store messages on local disk
- User interface
- Search, spam-filtering of messages
- Sending emails via smtp

Web-based email client (gmail):

- UI is operated through browser (HTML)
- No local storage of messages [Good and bad thing. 2TB of messages]
- All operations (view, delete, search, etc.) performed remotely
- Can't operate offline: Must be connected to gmail for all operations



## Cloud application: Someone else's problem

..Until it isnt.

### Advantages:

- Can access from anywhere
- Software updates and security patches
- Spam optimizations can be globally deployed

### Down-sides:

- Control over UI: What if UI changes and you don't like it?
- Privacy : full control over code if local.

## Network-centric computing

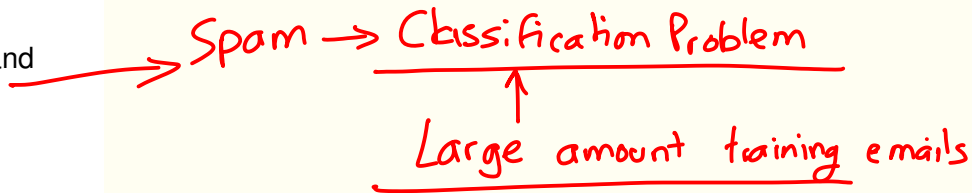
- Use remote computing resources
- Servers located in data centers
- Possible due to advancements in networking
- Access remote services with thin clients (e.g., chromebooks)
- Standardized interfaces for data transfer (WWW, HTTP)

Third perspective of cloud computing

## Cloud Advantages Recap

- Utility
- Pay only for what you use [mailbox size]
- Maintenance and security ensured by service providers
- Economy of scale: increases efficiency due to specialization and centralization [spam]
- Resource multiplexing [not everyone uses 2TB] lowers costs
- Redundancy: can store data on multiple locations

\$1/month per GB

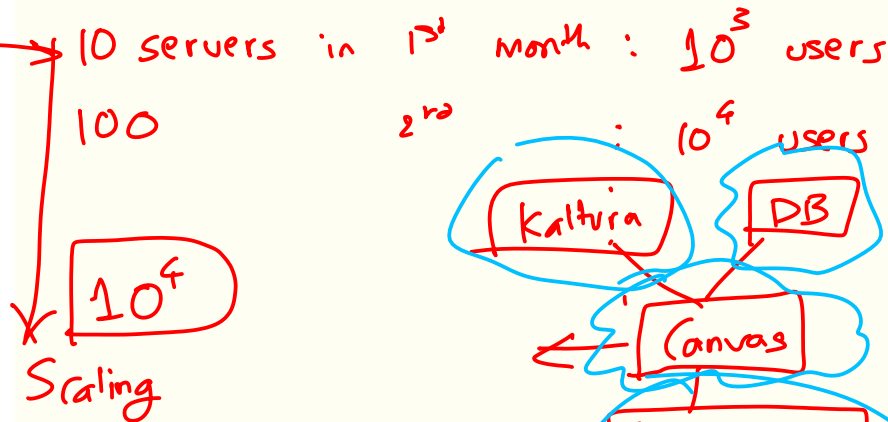


# Cloud Platforms

# Cloud uses

- Large enterprises
- Software startups: easy to deploy and scale.
  - Don't have to worry about buying and maintaining hardware
  - Uber: \$3Billion cloud bill per year
  - Netflix: famous as an early user of Amazon AWS *cloud*
- Scientific computing: NSF Chameleon Cloud
- Data analytics and machine learning

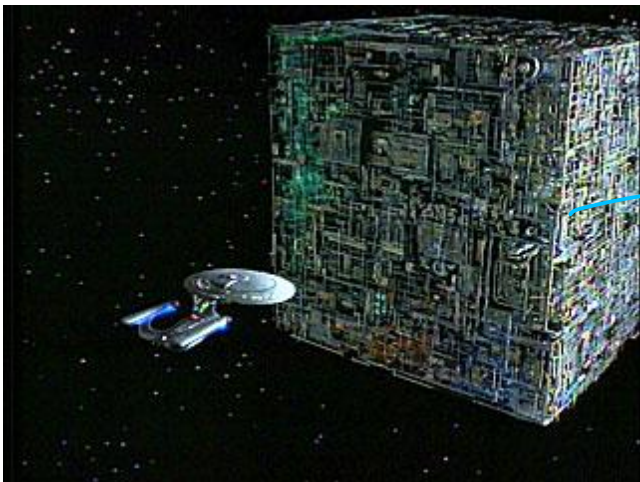
≈ "All" applications will be on the cloud, if not already.



Network effects

## Resistance Is Futile

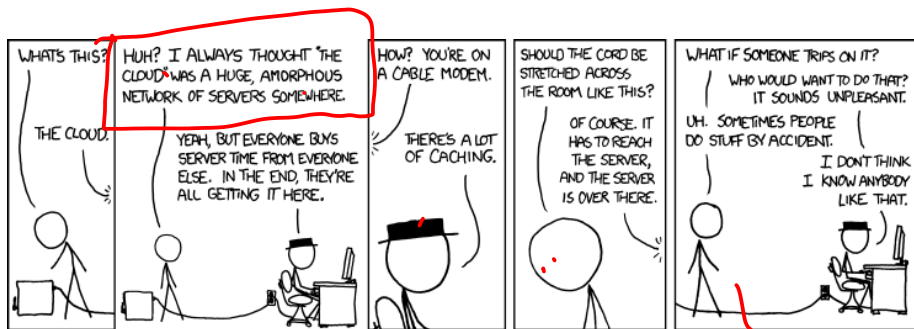
- \$200B industry
- Key reason for boom in Deep Learning and AI
- The cloud will assimilate all computing applications



→ "BORG" ≡ CLOUD

## There's an xkcd for that

Lets look what is inside that Borg Cube







## Modern Cloud Computing (Post 2006)

Folk-lore of how AWS began:

- Excess server capacity during slow shopping periods
- Use supply-side elasticity to offer unused resources as a service
- S3 (Simple Storage Service) and EC2 (Elastic Compute Cloud) in 2006

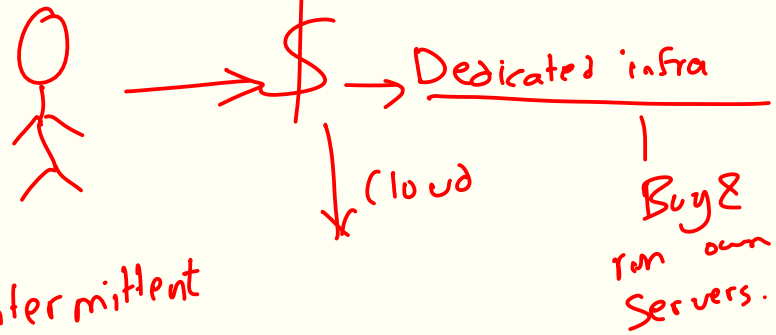
### Eric Schmidt in 2006

“It starts with the premise that the data services and architecture should be on servers. We call it cloud computing—they should be in a “cloud” somewhere.”

→ Peak provisioning —  $10^5$  servers  
(highest load/traffic)  $\approx$  Black Friday.  
Regular:  $1/2 \times 10^5$  servers needed

# Cloud Data Centers

- Warehouse sized buildings housing thousands of servers
- 2x more efficient than conventional data centers



Intermittent

Buy & run own Servers.

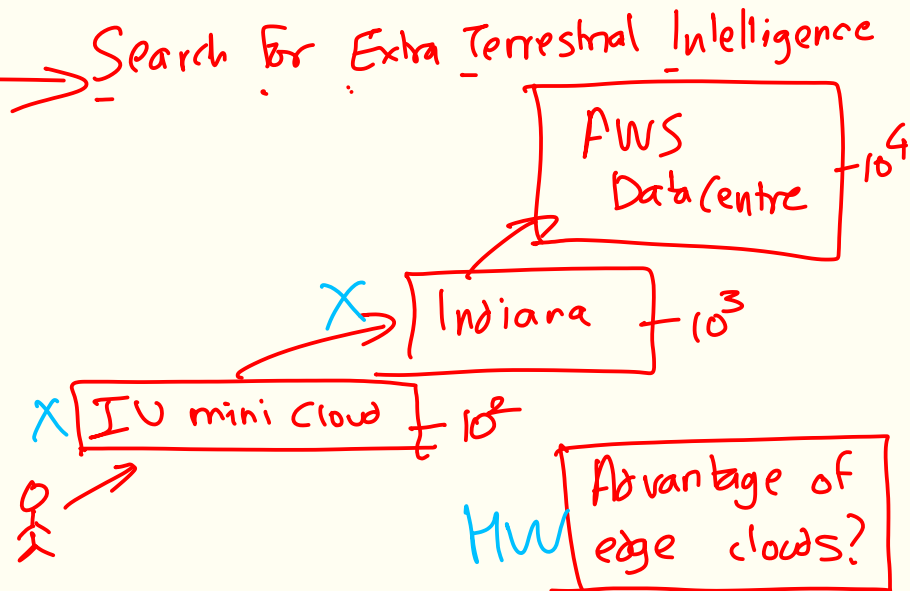
# Centralization

- Data processing and storage can be done more efficiently centrally in large data centers

## Alternative: Decentralized Architectures

- Example: SETI@Home in the 1990's used a vast network of volunteer desktop computers for highly parallel computation

↳ "Edge Clouds"



## Clouds enable specialization

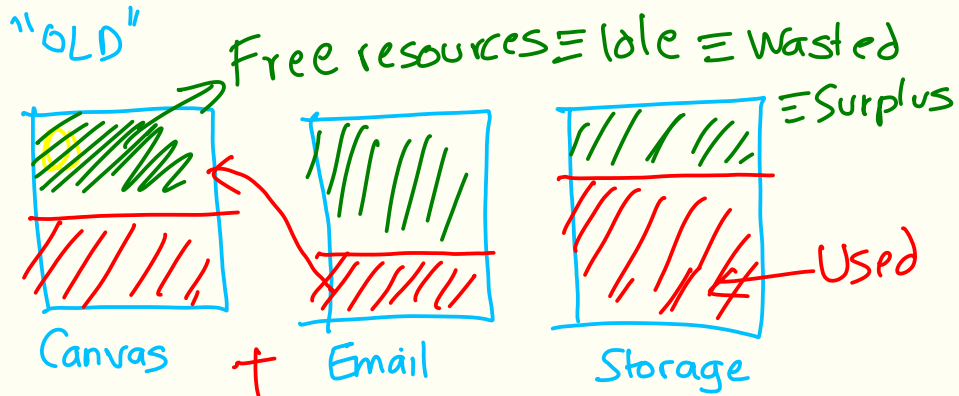
- Cloud Computing also reduces redundant software and hardware setups
- Running your own computing and storage cluster is *hard*
- Setting up and maintaining complex software is also painful



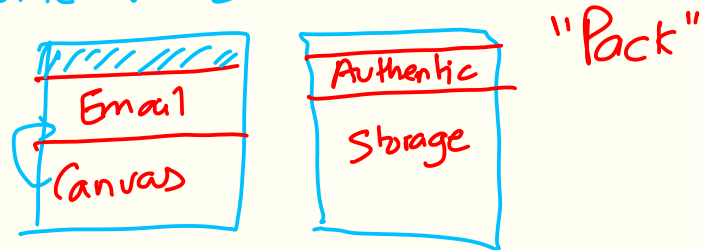
## Why are cloud platforms efficient?

- Resource pooling and multiplexing: multiple agents/tenants can share a large pool of resources, instead of strict division
- Large, optimized data centers that house all the hardware
- Higher hardware utilization (related to power proportionality)
- Virtualization
- "Software-defined" everything: networking, storage, data-centers

↓  
Illusion applications running on "virtual" hardware



① Load/traffic varies over time & is different



## Data center Challenges

### Hardware challenges:

- Powering, running, and cooling  $10^5$  servers
- Reliability and redundancy
- Low energy consumption
- Location: must be close enough to end-users

### Cluster management software:

- Run multiple apps and services
- Software for multiplexing, elasticity, ...
- Manage resources across all servers
- Where (which server) to run an application on?

Backup

power / cooling / network

2 Power stations,

4 Transformers,

UPS ("Batteries")

Each server: 2 power supplies

Low temperature locations (Siberia)

## Economies of Scale

- Larger data centers more efficient
- Large scale optimizations at different levels possible
- Better energy efficiency
- Administrative overhead
- Custom hardware [servers] possible. Remove unneeded components. Add more redundancy
- Easier to scale with larger clusters



# Cloud service abstractions

Utility Computing: Standard Interface "Plug"

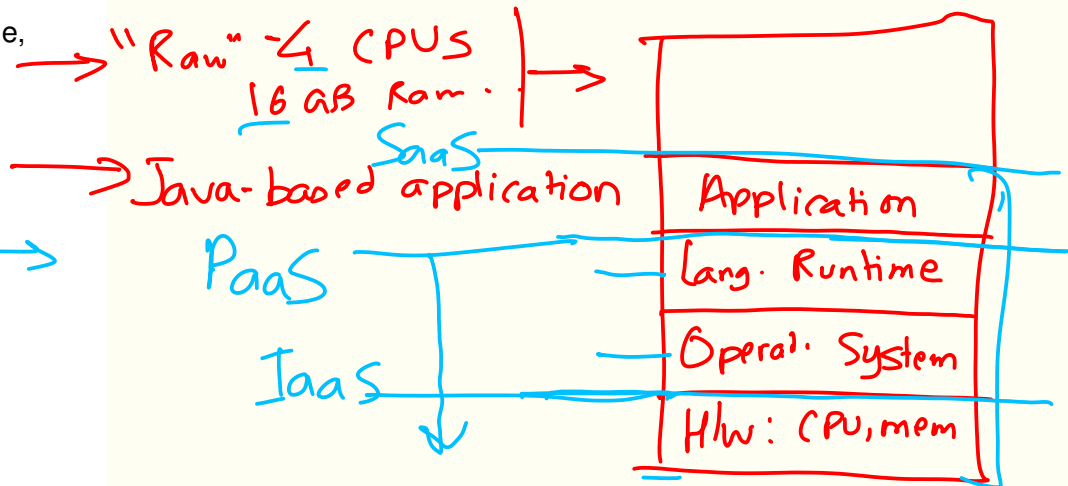
Cloud platforms offer computing resources at different levels of abstraction:

- **Infrastructure as a service.** Provide basic computing, storage, networking resources, typically virtualized.
- **Platform as a service.** Limited application deployment and programming API.
- **Software as a service.** Specific application such as email.

Tradeoffs:

- Programmer convenience vs. flexibility
- Higher abstraction services may have restrictions
- More management required with lower abstraction (IaaS)

Fundamental



## Cloud services

- 100s of different services for computing, storage, networking, etc.
- Examples: Databases as a service, email-relays, DNS servers, message queues,...
- Many services added **each week**
- Autoscaling , Deployment , Analytics , Distributed data processing, machine learning , Stream processing , Databases and data storage , Workflow automation , Network services: smtp, dns, etc.

Infrastructure level:

- Availability zones
- Geographical regions
- Network IP addresses

*Redundancy as a Service*

# Cloud Costs

- Due to economies of scale, cloud resources often cheaper
  - Energy efficiency of cloud data centers: 90%
  - Energy efficiency of smaller data centers: 50%
- Google cloud: \$0.02 per hour per CPU
- Bare-metal cloud (Hetzner) : \$0.004 per hour per CPU
- Total cloud costs includes storage and networking costs
- Cost of other services: load-balancers, elastic-mapreduce, ...

$$\frac{\text{Energy to run servers}}{\text{Total Data Center Consumption}}$$

↑  
Cooling, transmission losses, ...

## Cloud vs. local deployment



Cost considerations of moving applications to the cloud:

- Investment aka capital expenses: building data center, buying servers, ...
- Operating expenses: energy costs for running servers, cooling, and maintenance.

→ 1 time / fixed

→ Per hour ~ based on use.

## Potential Pitfalls

- Lock in 
- Privacy and security 
- Costs 