CS5412: SPRING 2014 CLOUD COMPUTING

Lecture 1

Ken Birman

Welcome to CS 5412...

A course dedicated to the technology behind cloud computing!

In my country of Khazackstan, many excellent hacker. We hack cloud, steal private stuff of whole world!

Cloud Computing: The Next New Thing

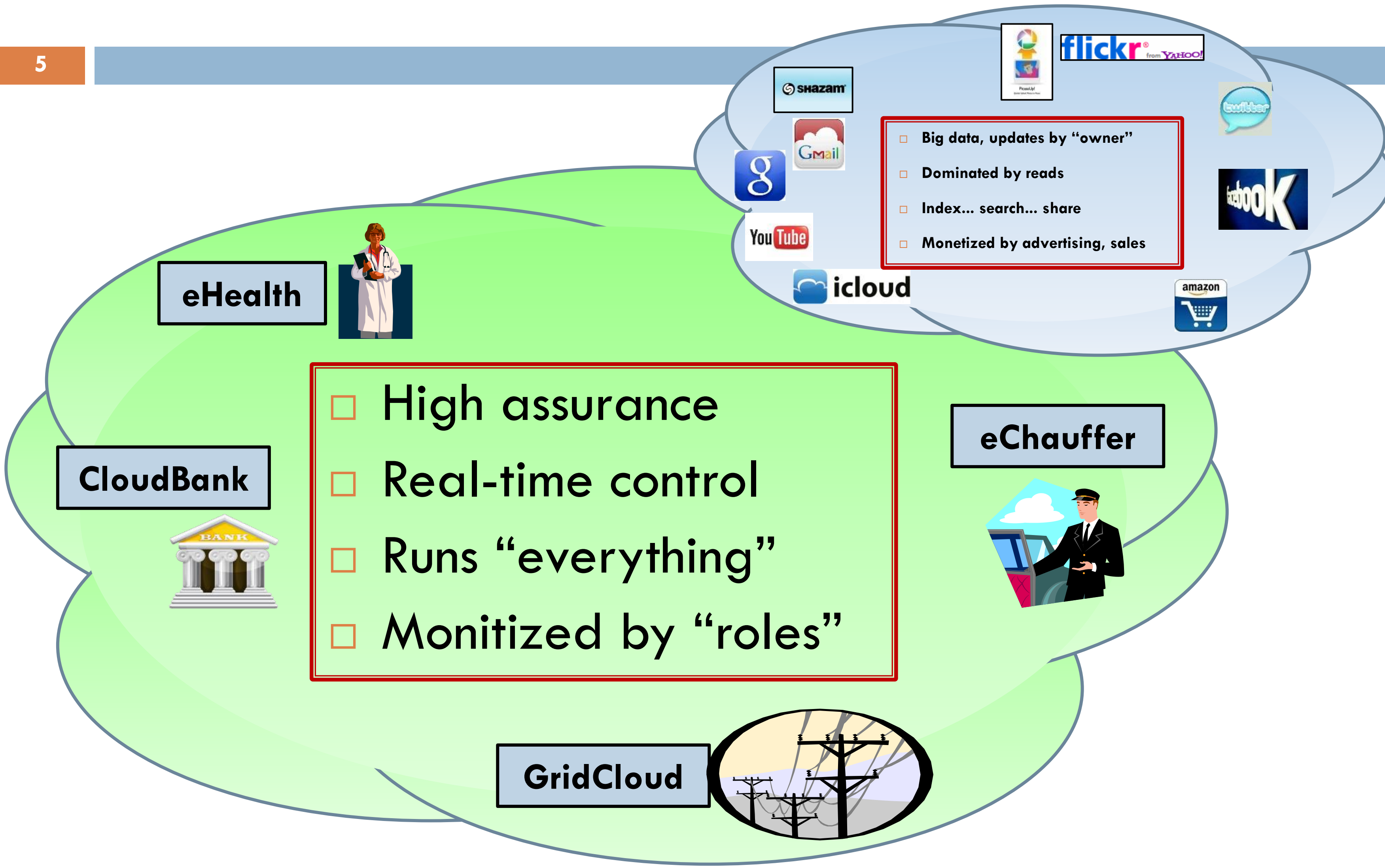
- A general term for the style of computing that supports web services, search, social networking
- Increasingly powerful and universal
- Enables a new kind of massively scaled, elastic app

- Our goal: understand the technology of the cloud, its limitations, and how to push beyond them
- Invent “highly assured cloud computing” options

Today's Cloud: Surprisingly limited

- Big data, updates by "owner"
- Dominated by reads
- Index… search… share
- Monetized by advertising, sales

Tomorrow's cloud?



Clouds are hosted by data centers

- Huge data centers, far larger than past systems
- Very automated: far from where developers work. Often close to where power is generated

  (ship bits… not watts)
- Packed for high efficiency. Each machine hosts many applications (usually in lightweight virtual machines to provide isolation)
- Scheduled to keep everything busy (but overloads hurt performance so we avoid them)

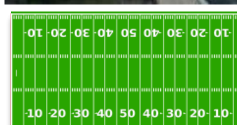
Clouds are cheaper… and winning…

Range in size from "edge" facilities to megascale.

Incredible economies of scale

Approximate costs for a small size center (1K servers) and a larger, 50K server center.

Technology	Cost in small-sized Data Center	Cost in Large Data Center	Cloud Advantage
Network	**$95 per Mbps/ month**	**$13 per Mbps/ month**	**7.1**
Storage	**$2.20 per GB/ month**	**$0.40 per GB/ month**	**5.7**
Administration	**~140 servers/ Administrator**	**>1000 Servers/ Administrator**	**7.1**



Each data center is
11.5 times
the size of a football field

Slide provided by Roger Barga, Head of Cloud Computing, Microsoft

Key benefits?

- Machines busier, earn more $'s for each $ investment
  - Hardware handled a whole truckload at a time
- Applications far more standardized
  - Automated management: few "sys admins" needed
  - Power consumed near generator: less wastage
  - Data center runs hot, wasting less on cooling
  - Can "rent" resources rather than owning them
- Supports new, extremely large-scale services
  - Elasticity to accomodate surging demands
  - Can accumulate and access massive amounts of data
    - But must read or process it in a massively parallel way
  - Enables overnight emergence of major companies, but scalability model does require new programming styles, and imposes new limits

Assurance properties

- Unfortunately, today's cloud
  - Has a limited security model focused on credit card transactions
  - Weakens consistency to achieve faster response times: the cloud is "inconsistent by design"
  - Pushes many aspects of failure handling to clients
- Model supported by the "CAP" and "FLP" theorems, which are cited by many application designers
- Instead, cloud favors "BASE"

Acronyms

- CAP: A theorem that says one can have just two from {Consistency, Availability, Partition Tolerance}
- FLP: A theorem that says it is impossible to guarantee "live" fault-tolerance in asynchronous systems (here, "live" ≡ certain to make progress)
- BASE: A cloud computing methodology that seeks "Basically available soft-state services with eventual consistency" and is popular in the outer layers (first tier) of the cloud. The opposite of ACID
- ACID: A database methodology: offers guaranted {Atomicity, Consistency, Isolation and Durability}.

CS5412: How to do better!

- Future cloud will need stronger guarantees than we see with today's cloud
  - How can we achieve those?
  - Are strong guarantees "scalable"?

- Betting that the cloud will win
  - Cheaper than other options…
  - … and the cheaper option usually wins!
  - But technology also advances over time, which helps!

Making the cloud highly assured

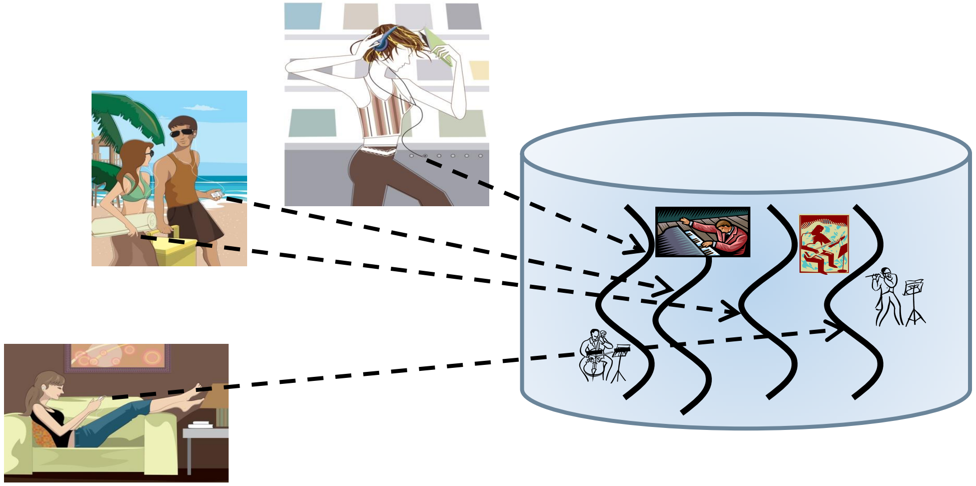
- Find ways to overcome limitations like FLP and CAP
- Define new assurance goals that might still be forms of security and consistency but are easier to achieve
- Only consider things that are real enough to be implemented and demonstrated to scale well and perform in a way that would compete with today's cloud platforms. A practical mindset.
- But use theoretical tools when theory helps with goals.

… And making it fast

- The cloud makes it easy to create "mashups"
  - Applications send data to each other, one system might "call upon" 10 or 100 others for help
  - Very powerful but also very inefficient in some ways
    - Example: Networks that become overloaded because of the same image or video being sent again and again!
- Getting the cloud to "scale" and perform well comes down to enabling productivity while also finding tricks to ensure super good performance
    - Example": store the image, ship a URL…

CS5412: Topics Covered

- We'll treat the cloud as having three main parts
  - The client side: Everything on your device
  - The Internet, as used by the cloud
  - Data centers, which themselves have a "tiered" structure
    - Like a dedicated and personal computer
    - Yet massively scaled with many moving parts
- Special theme: *high assurance*



The Old World and the New

- **Old world:** we replicated servers for speed and availability, but maintained consistency

- **New world:** *scalability* matters most of all
  - *Focus is on extremely rapid response times*
  - Amazon estimates that each millisecond of delay has a measurable impact on sales!

- **But** our premise is that we can have scalability and also have other guarantees that today's cloud lacks

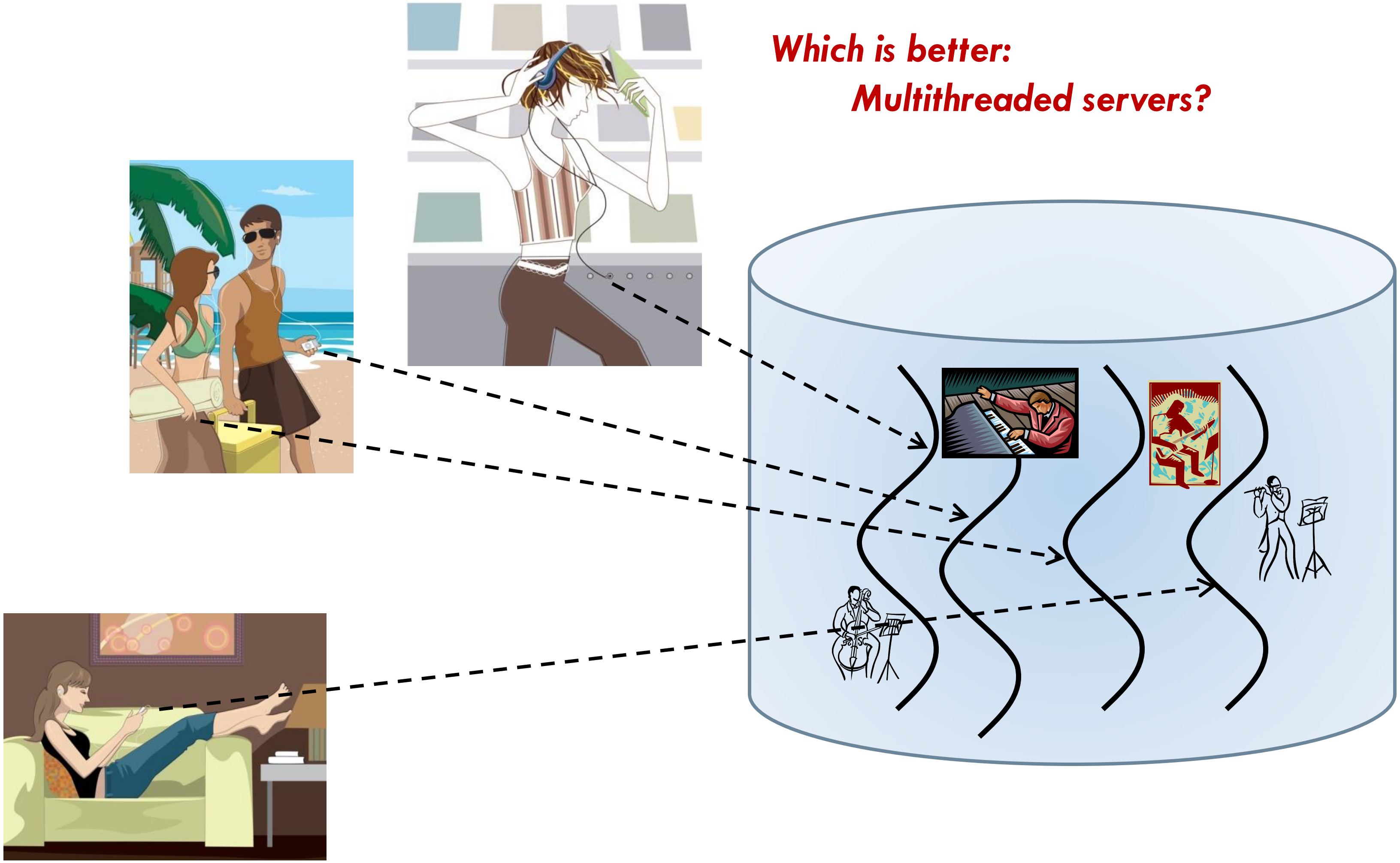
High Assurance: Many (conflicting) goals

- Security: Only correctly authorized users (who are properly authenticated) can perform actions
- Scalability: Can support lots of simultaneous users
- Privacy: Data doesn't leak to intruders
- Rapid response despite failures or disruption
- Consistency and coordinated behavior
- Ability to overcome attacks or mishaps
- Guarantee that center operates at a high level of efficiency and in a highly automated manner
- Archival protection of important data

Must ask many questions

- If we were to run high assurance solutions on today's cloud, what parts of the standards would limit or harm our assurance properties?
- Goal is to leverage the cloud or even run on standard clouds, yet to improve on normal options
- This forces us to look hard at how things work

Today's cloud focuses on easy stories

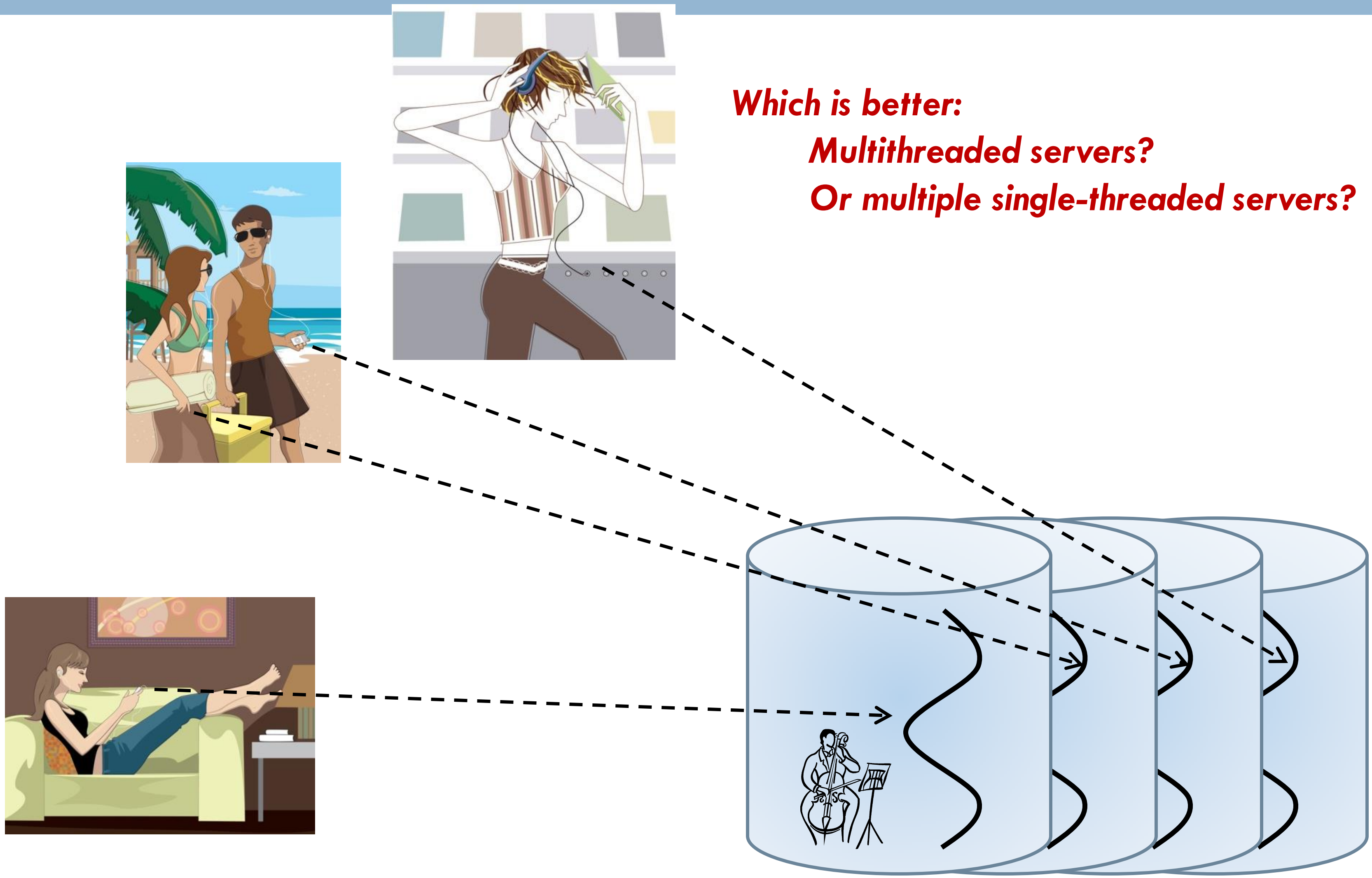


Today's cloud focuses on easy stories

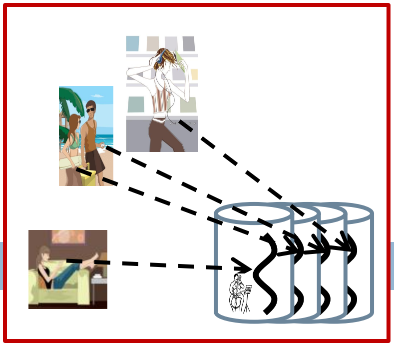
Which is better:

Multithreaded servers?

Or multiple single-threaded servers?



Which scales best?



- Build it the easy way!
  - One VM per server
  - Server handles one user
  - Make the server single threaded if possible
- Why?
  - Better fit to the hardware (no lock/memory contention)
  - Quicker way to build it, reuses existing stuff

Some of today's rules of thumb

- Built from things that already exist and already work, as much as possible
- Expect that each 10x scaleup will still break things and that much of your work will be on fixing them
- When feasible, go for "no brainer" scalability
  - Armies of cheap machines and cheap storage
  - A form of "brute force" solution
- Success stories of today's cloud often are applications that *naturally* fit this approach

Acronyms! (How to be a party bore)

- One issue with the cloud is that it has a million acronyms: IaaS, SaaS, PaaS, SOAP, AWS, EC2, S3…
  - These make for a *very* confusing landscape!
  - But a business perspective on the cloud only needs to focus on a few of them, as a starting point
- What does the "aaS" mean?
  - Cloud vendors sell "services"
  - "aaS" == "as a Service"

The Important *aaS options

- Infrastructure. (IaaS: Infrastructure as a Service)
  - Cloud vendor rents you some hardware
    - A network, perhaps a wide-area network
    - A machine, always “virtual” but perhaps just for you
    - A file server, again virtual, but you can save files in it
  - They operate this for you, and you pay for what you think you need (or sometimes, for what you use)
  - And they sell backup services too

- For example, you could rent a private Internet from AT&T, or 500 computers from Amazon EC2
  - AWS is elastic: you rent and pay by the hour
  - AWS can accommodate huge swings in your needs

amazon webservices™ EC2

The Important *aaS options

- **S**oftware. (SaaS: Software as a Service)
  - Cloud vendor runs some software that you use remotely
  - Classic example: SalesForce.com has a sophisticated infrastructure that manages your sales contact data
- In effect you "outsource" your sales support system and SalesForce.com runs it for you
  - Other SaaS options: accounting, billing, email, document handling, shared files…
  - They also apply patches, fix bugs…

The Important *aaS options

- **P**latform. (PaaS: Platform as a Service)
  - Cloud vendor creates a sophisticated platform (typically a software environment for some style of computing, or for database applications)
  - Your folks use it to create a custom solution
  - Cloud vendor runs your solution in an elastic way
- They promise that if you use their PaaS solution, you'll benefit from better scalability, performance, ease of development or other advantages

The Important *aaS options

- **Platform.** (PaaS: Platform as a Service)

Force.com
- Apex
- Force.com (Equinox)

Google Apps Engine
- Python
- Google

Microsoft Azure
- .Net
- Microsoft

Heroku
- Ruby, Java, Python, Node
- AWS

ORACLE
CLOUD COMPUTING

… these aren't the whole cloud

- The cloud *mixes* many models
  - Some integrate humans into the loop, such as outsourced audio-to-text, or Amazon's Mechanical Turk
  - There are companies with specialized roles
    - Akamai: The most famous data hosting company, especially successful for storing videos and images that are used in your web pages. They specialize in rapid data delivery
    - DoubleClick: You leave a frame on your web page, they put the perfect advertisement for this particular user in it
  - There are even cloud "HPC systems"! (Rent on demand)

But some standards pervade…

- The cloud really took off as an outgrowth from web sites and browsers
  - First we had browsers, HTML (a use of XML), HTTP, SSL
  - Then people had the idea of doing "client server" computing using browser web pages!
    - Called SOAP. A program makes a method call on a remote server… they encode it as a special web page
    - … this is sent to the server just as if it was a web request from a browser (in fact you can do it by hand…)
    - … result comes back in a special SOAP web page, extracted and returned to the calling program. Voila!

Web interoperability

- In fact the web is about interoperability
  - It is very easy to integrate
    - Data from multiple sources (e.g. Netflix sends you a web page but in fact the video comes via Akamai)
    - Different styles of computing (e.g. Weather.com fills a page with their content (the images come from Akamai), but the weather forecasts are from HPC computing systems and the advertisements are from DoubleClick. The ads might include a video hosted on YouTube, but Akamai might be the real source that sends the data…
  - By agreeing that "at the end of the day, web pages are the lingua franca" a great leap forward happened

(Web pages are inefficient…)

- The encodings used in the web are terribly inefficient, though
  - So they made browsers extensible
  - You get “plug ins” from Adobe, GZip, Microsoft, … and those plug-ins “extend” the browser to understand special data representations
- Modern browsers can download and run full programs coded in Javascript, Silverlight, Caja or even true Java… and these programs can do anything at all

Open source

- The cloud has hugely benefitted from
  - **open source** (basically, source for programs is made available to customers),
  - **free open source** (same, but no fee for use), and
  - **open development** (many developers at many companies contribute).

- In fact nothing about the cloud demands “open.”
- But these are certainly powerful factors that help explain the vibrant cloud ecosystem.

Open source debate

- Many companies debate open source
  - Quite a few have policies against it
- Yet they run Linux on their servers, build programs in C++ using gcc, allow employees to install their favorite browser add-ons, use Mono to create Linux versions of their Windows applications
- Java compiles to JIT code that reverse compiles back to Java source
- Believe me: You use open source even if you think you don't! (You probably even have employees who contribute to some open source projects…)

Deeper connection to cloud

- The cloud is a world of open standards
  - For the first time, the cloud tore down the high protectionist walls of proprietary products
  - At many levels, we can see how things work and jump in and modify things
  - Plug-and-play… from the client system into the network and right up to the datacenter!
- The cloud is a world of easily interconnected component technologies that play together nicely
  - And openness has been a key enabler in this happening

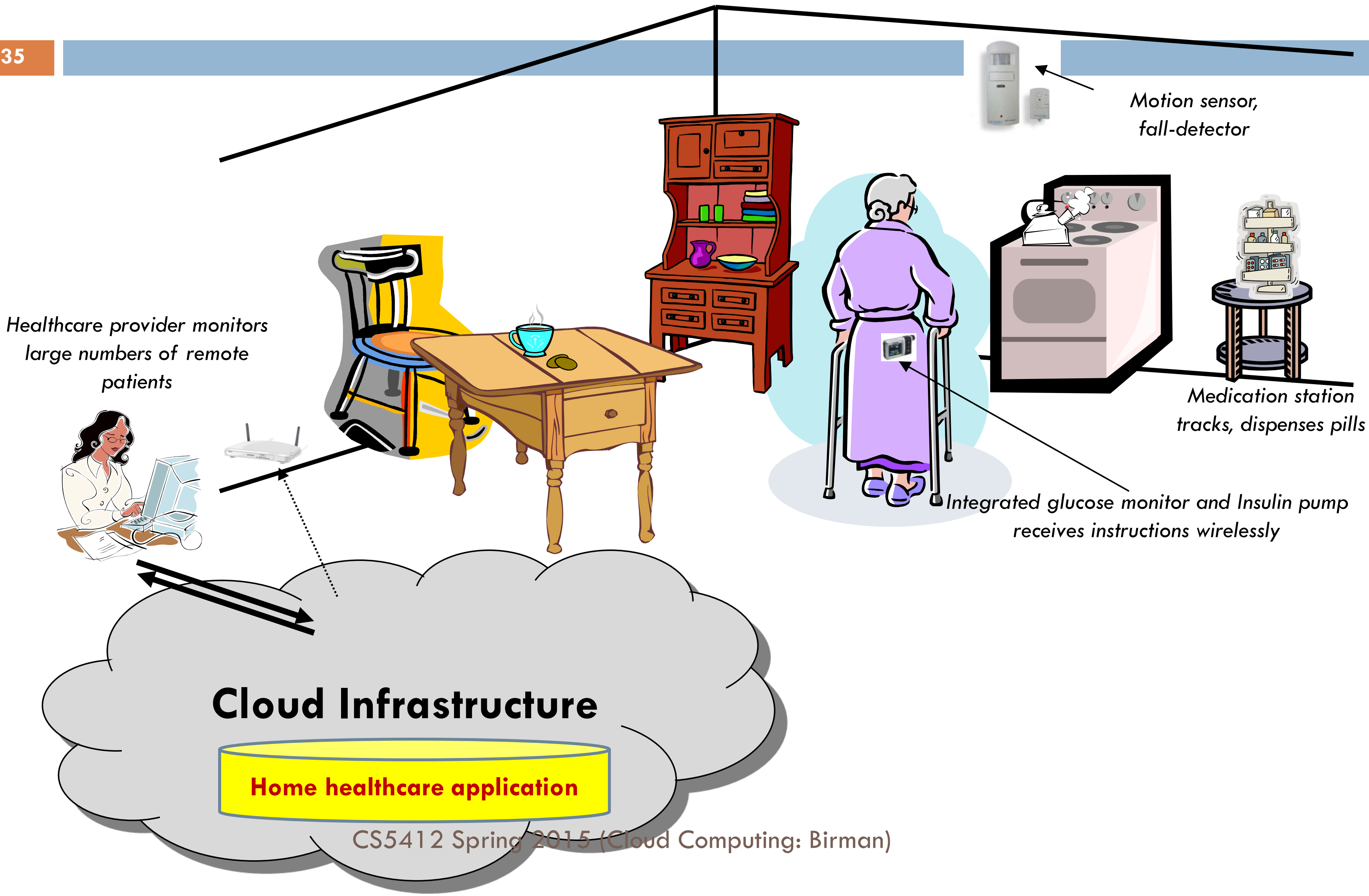
So… what's cloud computing?

- In some sense, the term means nothing!
  - If you make "full use" of modern off-the-shelf computing products and systems, you are a cloud computing user
  - You can't really buy "non-cloud" systems anymore

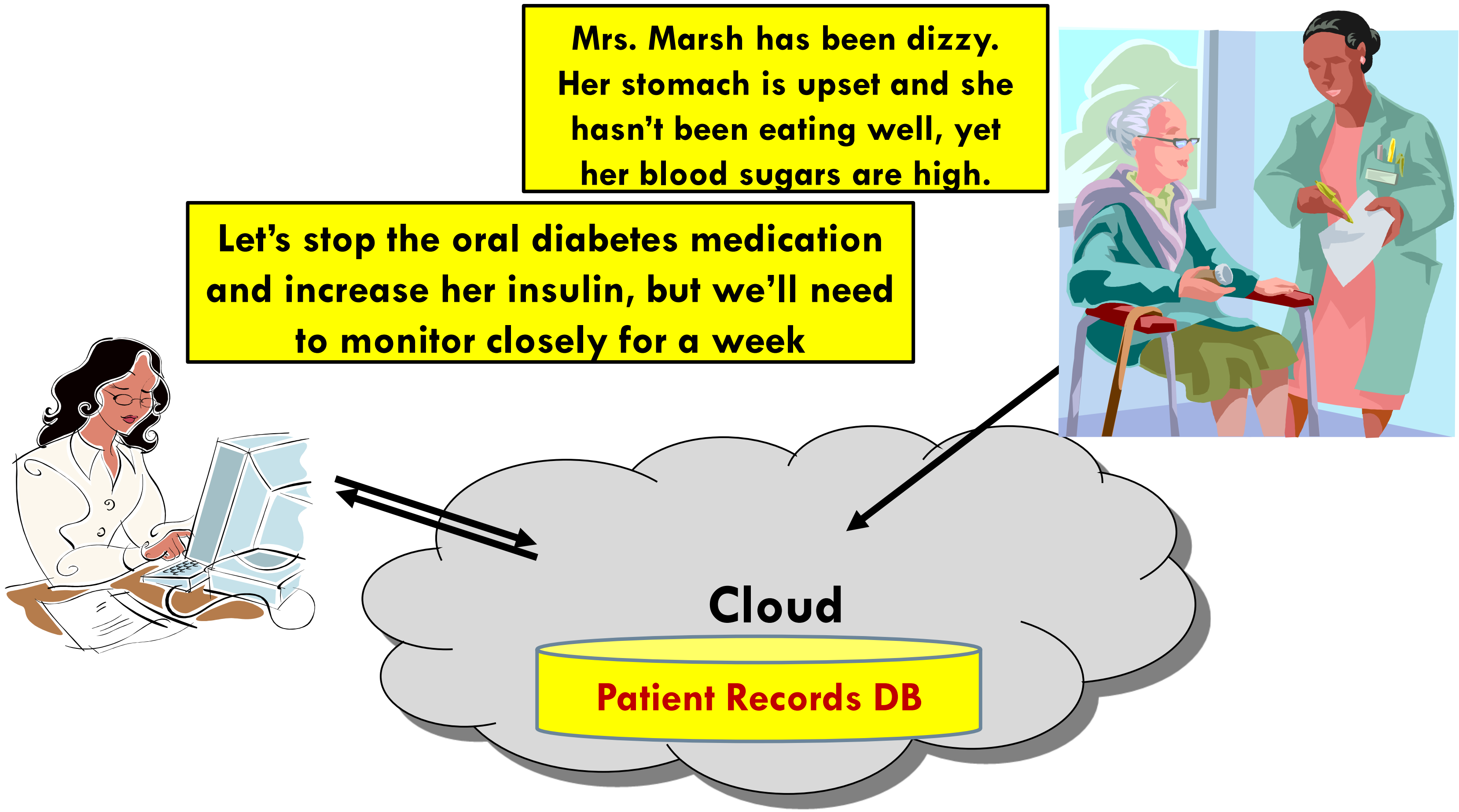
The Internet and cloud standards are built into everything

- You can block some features, but it is surprisingly hard to create a cloud-free computing system (one of many reasons it is so easy to break into many systems)

Can a cloud host high-assurance apps?

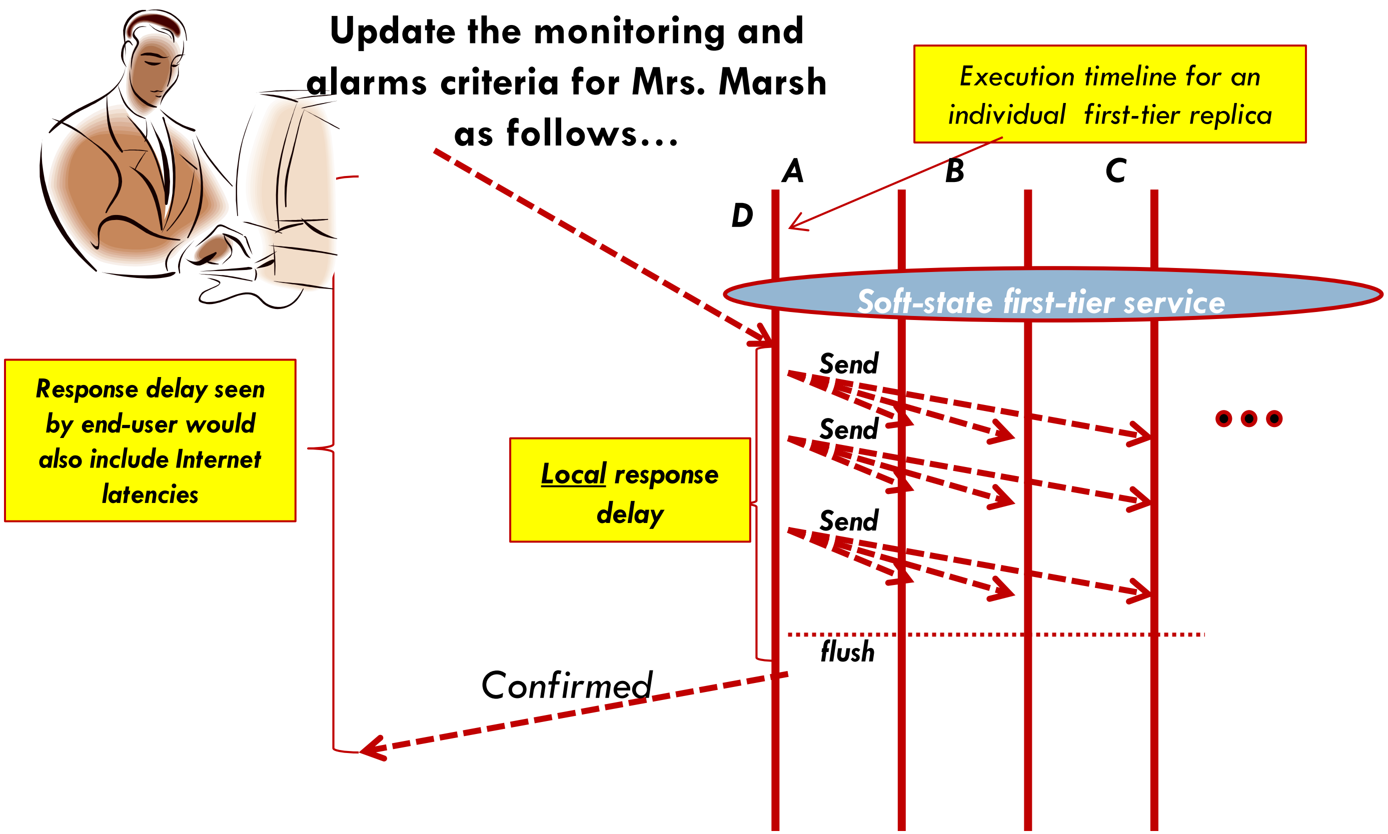


Which matters more: fast response, or durability of the data being updated?



- Tradeoffs determine speed and scalability!

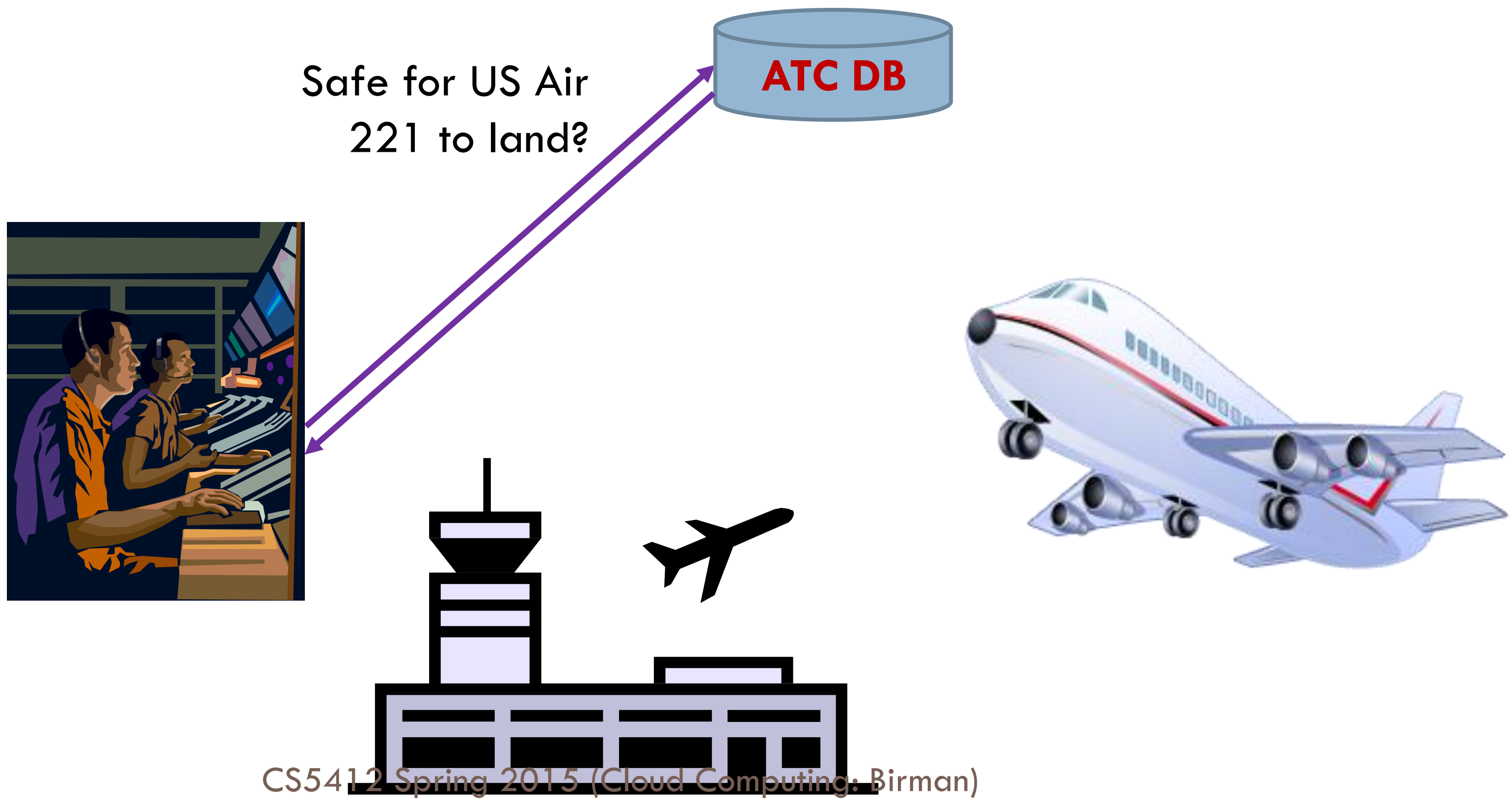
What if we were doing online monitoring?



- Durability matters more for patient records. But a monitoring system lives "in the moment" and mostly needs speed

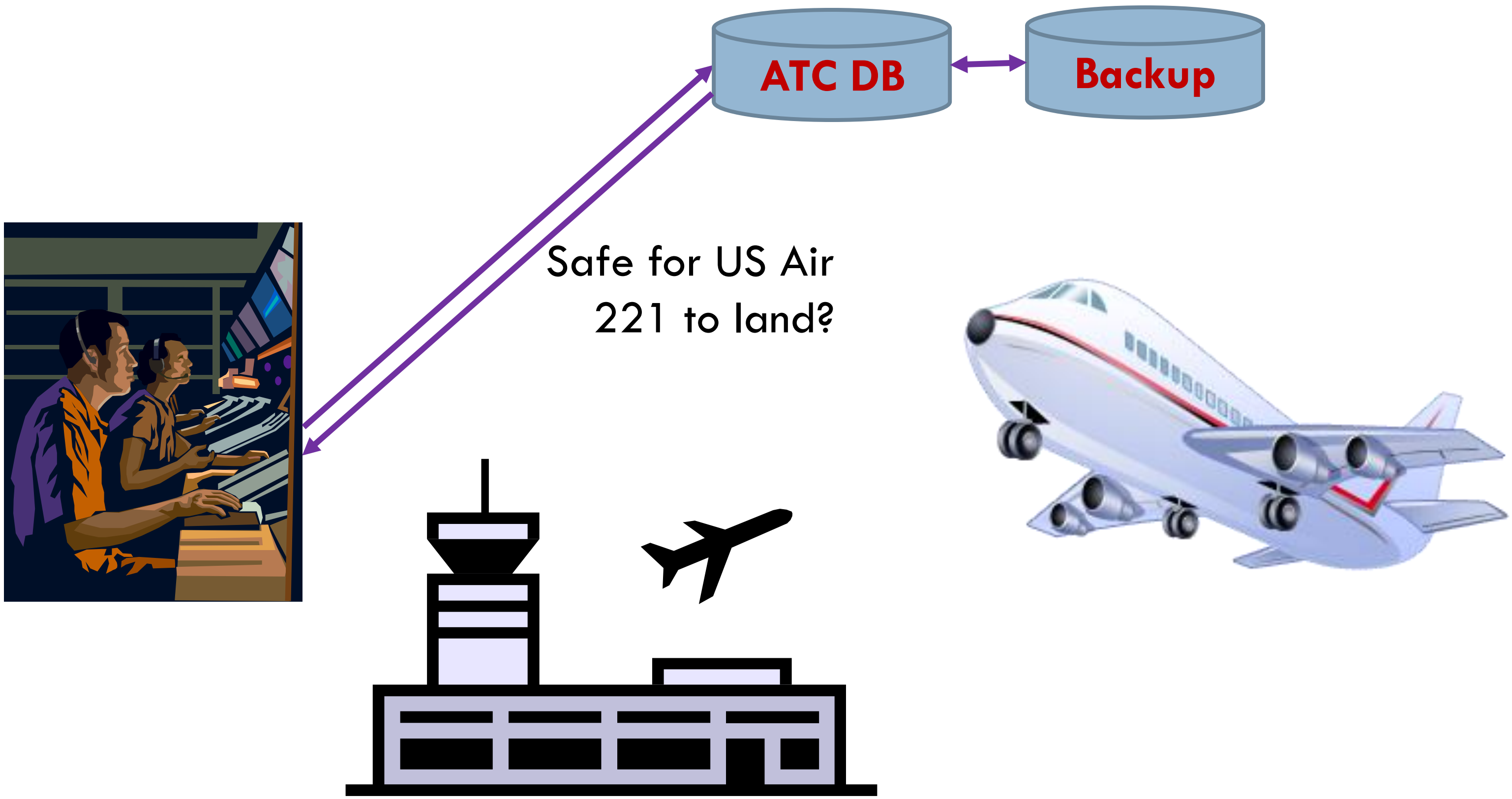
Which matters more: consistency or fast response?

- Air Traffic Controllers depend on consistent data
- With a single server this isn't hard to guarantee



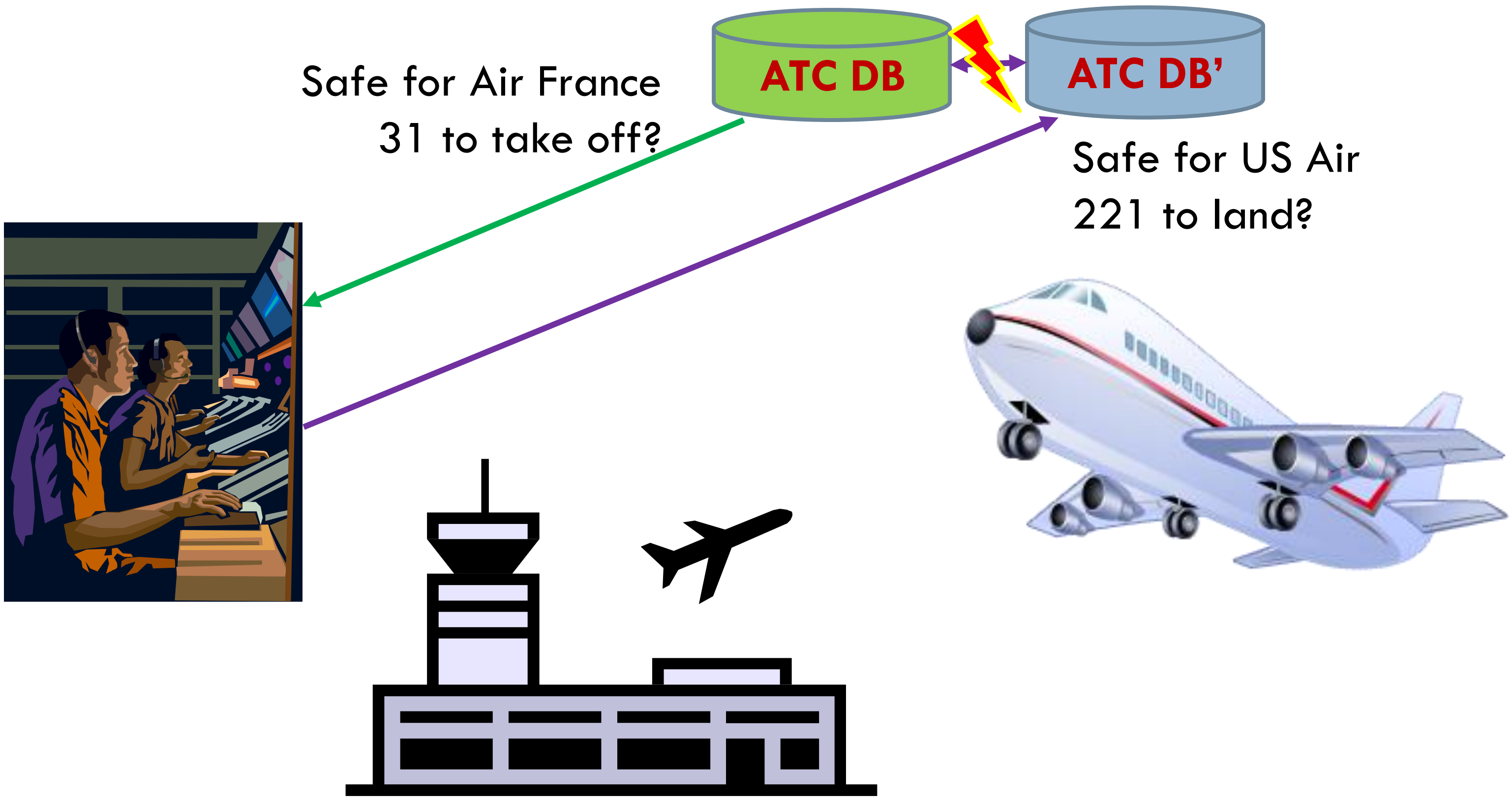
Which matters more: consistency or fast response?

- But suppose we replicate the server?
- Designate one as "primary"



Which matters more: consistency or fast response?

- Failure detection will be key to consistency
- Otherwise could end up with two primaries!



Cloud computing: A world of tradeoffs!

- Cloud computing systems
  - Overcome failure by replicating services
  - But have no standard way to decide which server is in charge for a given service
- Easiest form of failure "detection" is by timeout
  - But this might not be accurate: a network partitioning problem will look like a failure
    - Maybe just *some* connections will fail
    - And if the network then recovers, the old ATC service might not even know that we think it crashed!

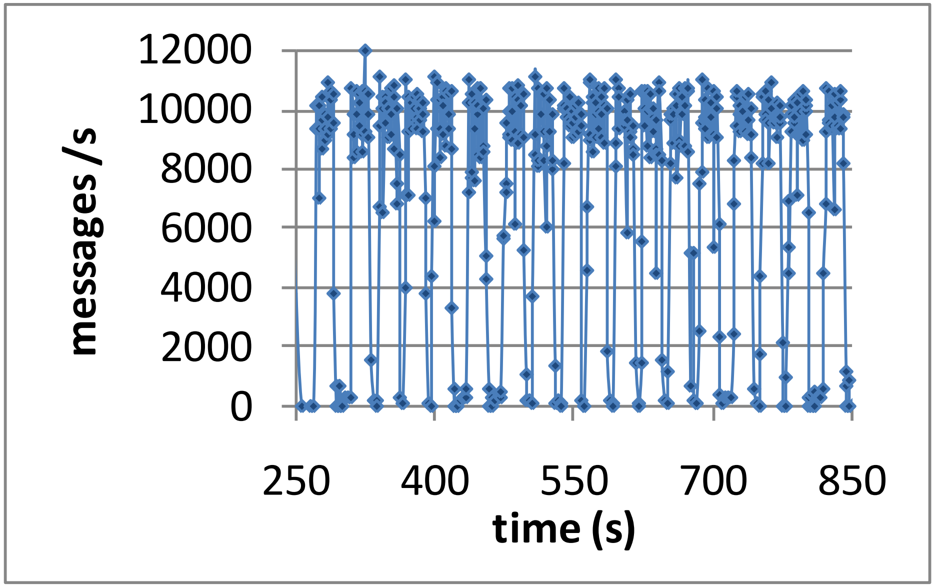
Replication is central throughout

- How to scale? Just add more replicas, balance load
- Fault-tolerance? If something crashes but has replicas, the impact is localized and other servers can take over
- Elasticity? Launch new replicas or shut some down

- What makes replication hard are cases where we need to think about coordination, concurrency control…
- If we don't worry about such things, may even be able to reuse existing applications!

Thrashing: Illustrates that 10x concern

- With small-scale replication, IPMC is a big win
- But IPMC "storms" can occur in a data center with many replicas and heavy update rates
- Wild load swings, heavy loss rates, thrashing



High assurance in the cloud

- Today's cloud is built with simple components and yet even so, exhibits problems like split brain behavior, thrashing, rolling failures, other issues
  - Companies spending a fortune to eliminate such issues
  - They can limit scalability
- Tomorrow's cloud thus poses a deep question
  - Will it be limited to simple applications?
  - Or can we migrate application like health care, transportation control, banking, etc to the cloud?

How will CS5412 approach such a complex set of problems?

- We'll take a step-by-step approach
- First look at properties of the client platform
- Next consider Internet and its evolution under pressure of the cloud (e.g. for controlled routing, higher availability, better security)
- Finally focus on the data center and look at it tier by tier from the first tier inwards

At each level look at assurance issues

- High assurance means different things in each layer
  - A client depending on a browser worries about apps, personalization, connectivity, mobility, web-site spoofing, viruses, key-stroke logging, privacy…
  - The network worries about efficient routing, BGP problem, DDoS attacks, authenticating
  - The cloud worries about maintaining rapid response, balancing load, automating management, consistency, fault-handling, etc.

CS5412 Gets more technical as we go

- For the first few weeks, we'll be more engineering oriented, because the first kinds of issues are ones that center on how scaled-out systems are built
- But then as we focus more on replicated processing and replicated data, we'll bring more theory into the picture
- Fault-tolerance will round off our investigation. We'll explore many fault "models" but limit ourselves to ones seen in practice. We won't do as much on security.

CS5412: Grades

- Approximately 25 lectures, with [0-5] surprise quizzes to make sure you come to class.
  - Must be in class on time to take quizzes. No makeups!
- Homework assignments: Everyone does them, work as individuals, gain hands-on experience.
- Prelim (in class) and final.
- Course is curved to a B+/A-

CS5412: Organization

- Professor Birman gives most lectures
- Course roughly parallels his textbook
  - Many assigned readings from textbook but they aren't really required per-se; intended to help you understand the material
  - Any quiz would focus on material covered in class because the goal of the quizzes is to ensure that you actually are coming to class
- We have one full-time TA and three part-time TAs
- Wednesday recitation: for homework help

CS5412: Projects

- Used to be a part of this course but not in spring 2015 due to huge number of students, limited TA resources and (very frankly) some academic integrity issues in 2014

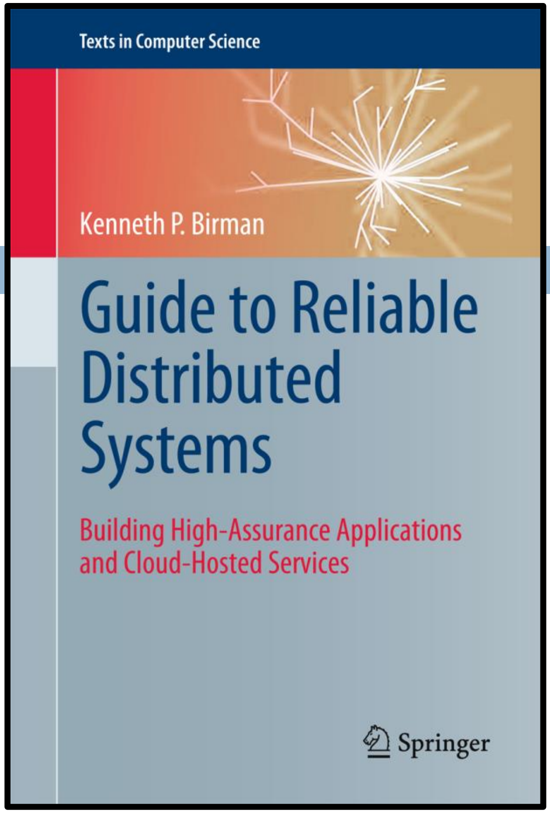
- In 2015 projects will only be for students seeking to satisfy the MEng project requirement. You need permission and would sign up for CS5999 credit (3 credits).

Examples of projects from 2012

- Integrate Isis^2 with Live Objects
- Build services of the kind Amazon uses for system monitoring using Code Partitioning Gossip
- Simulate and/or experiment on flow control for large scale replicated data sets, find best approach
- Implement a realistic Air Traffic Control system with high assurance properties (or a health care system)
- Explore best options for wide area file transfer

CS5412: Textbook

- We'll be using Ken's textbook
  - Written as a teaching tool
  - Ken doesn't earn royalties on it!



- Available on reserve in library if you prefer not to own a copy

- Reading assignments will often be from the book but we may also assign a few published papers

Background assumed?

- Solid understanding of computer architectures, operating systems, good programming skills including "threads" in Java, C++ or C#
- Some basic appreciation of how networks work, how operating systems work, virtualization
- Prior exposure to "distributed computing" not required or expected