An Opening in the Clouds: Open-source Cloud Computing at UCSB

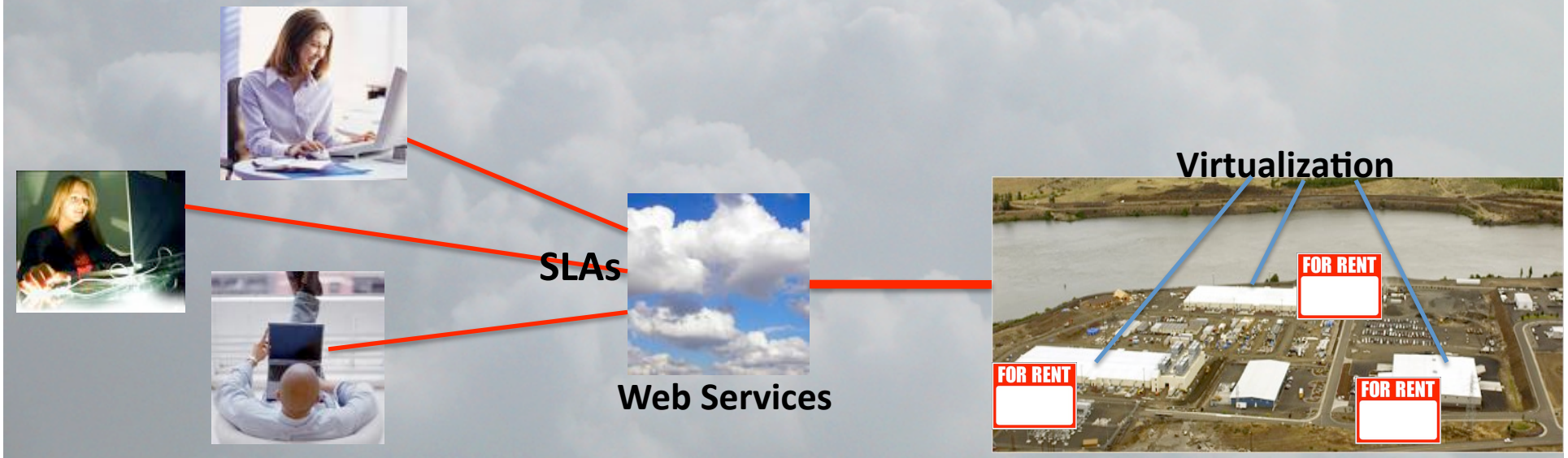
Chandra Krintz
Associate Professor
Computer Science Dept, UCSB

July 2009

Cloud Computing

- Software systems for accessing easily and transparently scalable CPU/storage/network resources via a network connection or web interface – "*as-a-service*"

- On a rental basis

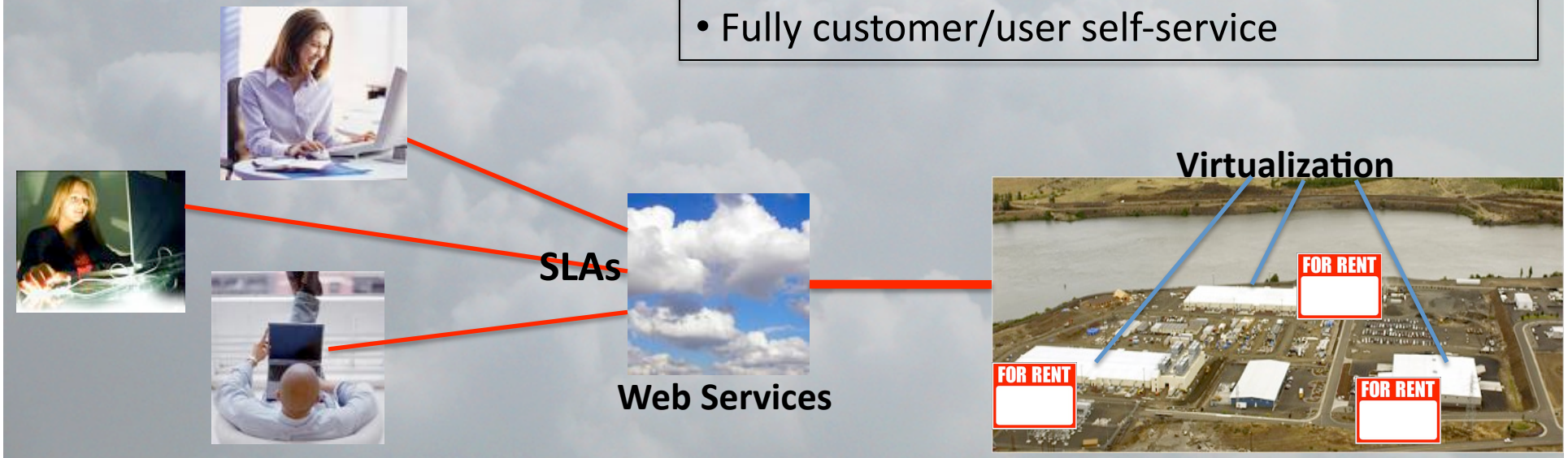


Cloud Computing

- Software systems for accessing easily and transparently scalable CPU/storage/network resources via a network connection or web interface – "*as-a-service*"

- On a rental basis

- Service level agreements (SLAs)
  - Users get small fraction of resource pool
  - Resources are opaque
- Pay-as-you-go or flat-rate (e-commerce based)
- Fully customer/user self-service



Cloud Computing

- Software systems for accessing easily and transparently scalable CPU/storage/network resources via a network connection or web interface – "*as-a-service*"
  - Infrastructure, e.g. Amazon Web Services (AWS) **IaaS**
    - Provision isolated resources under contract
    - Full-system images deployed over virtual machine monitor
  - Platform, e.g. Google AppEngine (GAE), Microsoft Azure **PaaS**
    - Enable construction of network-accessible applications
    - Process-level runtime isolation
    - Specialized/scalable runtime and library support
  - Software, e.g. Salesforce **SaaS**
    - Remotely accessible and customizable applications

Cloud Fabrics Today

- Culmination of grid/cluster/utility/elastic computing
  - Software: Virtualization, operating systems, programming and runtime support, fault tolerance & distributed computing
  - Hardware (multicore platforms)
- Has experienced a rapid uptake in the commercial sector
  - Public clouds – you run your systems/apps on others' systems
    - Reduces hardware and IT costs, administration overhead
    - Very easy to use – broadens the user community
  - Availability guarantees and extreme scale
    - Enabled via significant constraints on resource and service use
      - SLAs limit resource use
        - » CPU hours, network bandwidth, memory, storage
      - Platform-level
        - » Restricted app domain and subset of language/libraries
        - » Responses must complete very quickly

Cloud Fabrics Today

- Culmination of grid/cluster/utility/elastic computing
  - Software: Virtualization, operating systems, programming and runtime support, fault tolerance & distributed computing
  - Hardware (multicore platforms)
- Has experienced a rapid uptake in the commercial sector
  - Public clouds – you run your systems/apps on others' systems
    - Reduces hardware and IT costs, administration overhead
    - Very easy to use – broadens the user community
  - Availability guarantees and extreme scale
    - Enabled via significant constraints on resource and service use
- Application domain continues to be primarily web services

Cloudy Issues

- Many institutions and companies own IT infrastructure
- Public cloud features also useful for "on-premise" clouds
  - Privacy of code and data
  - Avoids vendor "lock-in" and pay-per-use
  - Potential for hybrid and customized approaches
  - Potential for easing resource constraints
    - Storage/data management, cpu/memory, communication
- Public clouds are opaque – open APIs, closed implementation
- Can cloud fabrics support other application domains, services, performance/availability requirements?

An Opening in the Clouds

- Open-source cloud computing systems from the **UCSB Computer Science Department**
  - **Goal**: Bring popular cloud fabrics to "on-premise" clusters that are easy to use and are transparent
  - To facilitate investigation of
    - Novel application domains, services, underlying device technology
    - Hybrid cloud solutions (public and on-premise)
    - Support technologies (e.g. tools, data management, autoscaling)
    - Customization (availability, performance, application behavior)
  - By emulating key cloud layers from the commercial sector
    - Private clouds are hybrid clouds – users want the same APIs
      - Applications/services/tools execute on either
    - Leverage extant software technologies
  - Not replacement technology for any Public Cloud service

Cloud Computing

- Software systems for accessing easily and transparently scalable CPU/storage/network resources via a network connection or web interface – "*as-a-service*"
  - Infrastructure, e.g. Amazon Web Services (AWS) **IaaS**
    - Provision isolated resources under contract
    - Full-system images deployed over virtual machine monitor
  - Platform, e.g. Google AppEngine (GAE), Microsoft Azure **PaaS**
    - Enable construction of network-accessible applications
    - Process-level runtime isolation
    - Specialized/scalable runtime and library support
  - Software, e.g. Salesforce **SaaS**
    - Remotely accessible and customizable applications

Cloud Computing

- Software systems for accessing easily and transparently scalable CPU/storage/network resources via a network connection or web interface – "*as-a-service*"
  - Infrastructure, e.g. Amazon Web Services (AWS)
    - Provision isolated resources under contract
    - Full-system images deployed over virtual machine monitor
  - Platform, e.g. Google AppEngine (GAE), Microsoft Azure **PaaS**
    - Enable construction of network-accessible applications
    - Process-level runtime isolation
    - Specialized/scalable runtime and library support
  - **AppScale**
    - Web services based implementation of Google App Engine
      - Complete application stack for MVC-based web applications
      - Written in Python orJava

From Google App Engine to AppScale

- Open-source platform-as-a-service (PaaS) that emulates Google App Engine (GAE)
- GAE is a full application stack that facilitates construction of interactive webpages with a database backing store
  - Users develop Python and Java apps using well defined APIs
    - Highly scalable proprietary implementation on Google resources

    Datastore -> Bigtable/Mapreduce

    MemCache -> in-memory datastore

    Authentication -> Google Accounts

    Mail -> GMail

    URL-Fetch (for HTTP/S communication)

    Images

    Task Queues for short background jobs

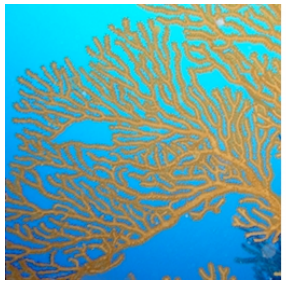
From Google App Engine to AppScale

- Open-source platform-as-a-service (PaaS) that emulates Google App Engine (GAE)
- GAE is a full application stack that facilitates construction of interactive webpages with a database backing store
  - Users develop Python and Java apps using well defined APIs
    - Test/debug via a non-scalable SDK
      - Simple implementations of APIs, e.g. flat file for Datastore
    - Upload to Google's highly scalable resources
      - Execute within a **sandbox** for isolation, control, autoscaling
      - Only a small subset of the language APIs can be employed

From Google App Engine to AppScale

- Open-source platform-as-a-service (PaaS) that emulates Google App Engine (GAE)
- GAE is a full application stack that facilitates construction of interactive webpages with a database backing store
  - **Sandbox / restrictions**

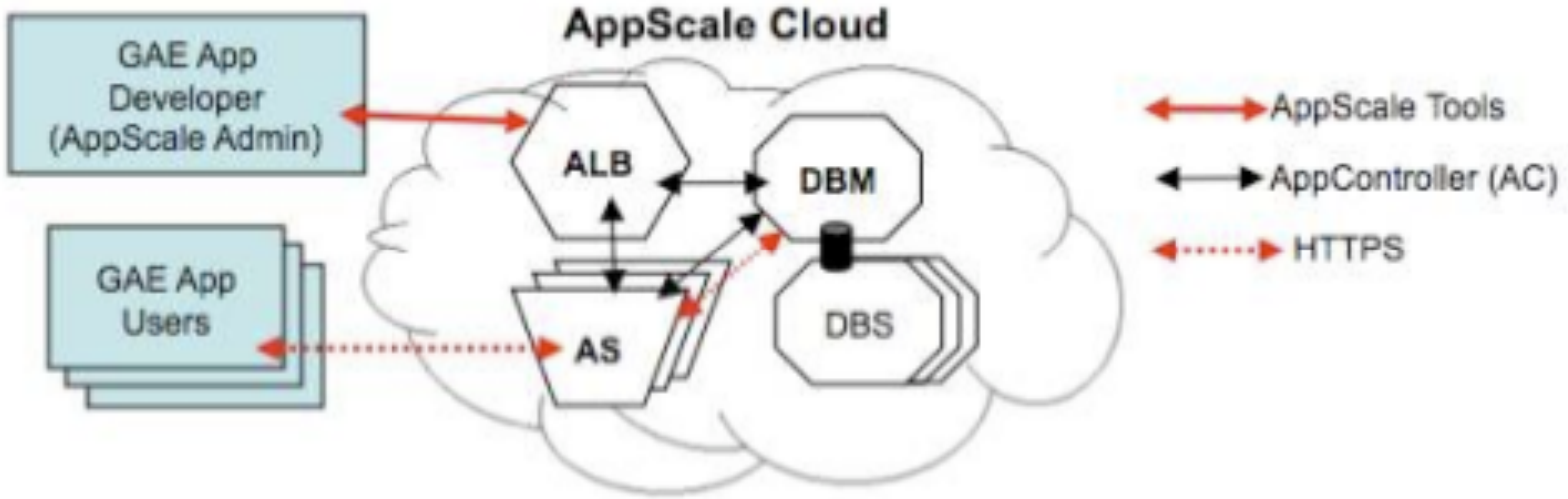
    Pure Python or Java, no thread/subprocess spawning, system calls
    No writes to file system, reads only to static files uploaded w/app
    Storage using key-value, schema-free datastore (Bigtable-based)
    HTTP/S communication only, CGI to handle page requests
    Limit on number of datastore elements accessed per request
    Limit on response duration, task frequency, request rate
    Enforced quotas (BW, CPU, requests/s, files, app size, …)



AppScale

- Extension of GAE SDK with API implementations replaced
  - Datastore -> HBase, Hypertable, Cassandra, Voldemorte, MySQL
  - MapReduce -> Hadoop
  - Authentication -> built-in, decoupled from Google Accounts
  - All inter-component communication via SSL

AppLoadBalancer (ALB)
AppServer (AS)
Database Master/Peer (DBM)
Database Slave/Peer (DBS)



AppScale

- Process
  - Deploy AppScale using the ***AppScale Tools*** (admin)
    - Specifies the datastore implementation for the system
    - Multiple GAE apps can use single AppScale infrastructure
  - Upload/remove GAE apps to the AppScale Deployment (devs)
  - *Users* of GAE apps access the ALB initially then an AS directly once rerouted
    - If AS goes down, user revisits the ALB to locate another AS

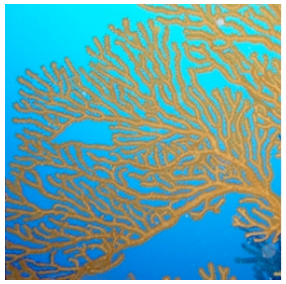
appscale-run-instances	//deploy a new AppScale instance
- identifies the cloud type, initial GAE app if any, and datastore	
appscale-describe-instances	//list all running AppScale instances
appscale-upload-app	//upload a GAE app to an AppScale instance
appscale-remove-app	//shutdown/remove an uploaded GAE app
appscale-terminate-instances	//shutdown an AppScale instance

AppScale

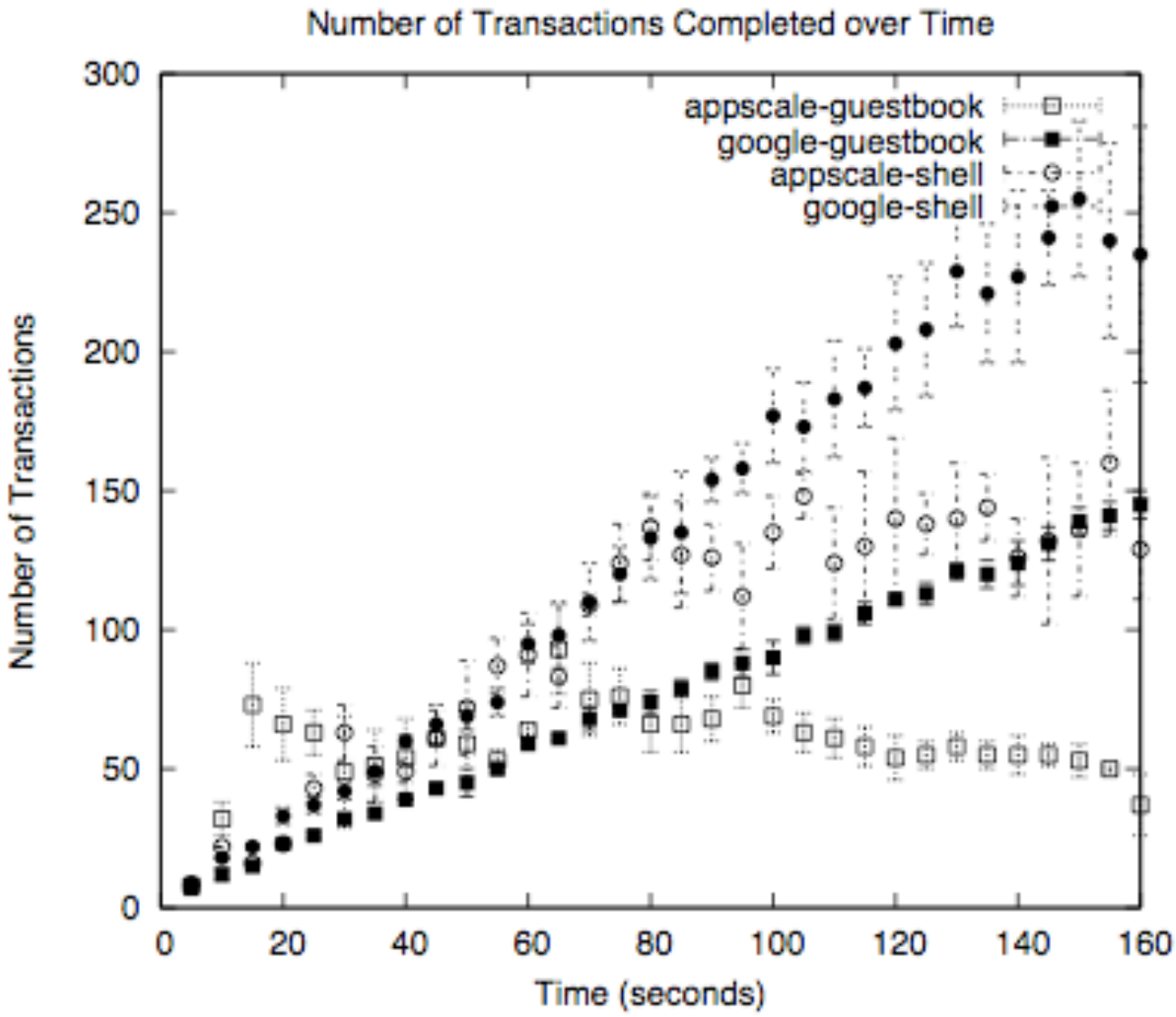
- Deploys and executes automatically over (cloud types)
  - Eucalyptus **Lead: Rich Wolski, Eucalyptus Systems, Inc**
    - Elastic Utility Computing Architecture Linking Your Programs To Useful Systems
    - Open source Infrastructure-as-a-service (IaaS) framework
    - Web services based implementation of elastic/utility/cloud computing infrastructure
    - Linux image hosting via virtualization
      - Available via popular Linux distros: Ubuntu, Debian, CentOS…
    - Emulates the **Amazon AWS interface**
    - **http://www.eucalyptus.com**

AppScale

- Deploys and executes automatically over (cloud types)
  - Eucalyptus
    - Virtualization via Xen, VMWare, KVM
    - Prefered installation/use of AppScale
  - Amazon's EC2
- With some manual intervention to set up VM instances
  - Xen
  - KVM
- Can run on non-virtualized Linux systems as well
- Suggested distro: Ubuntu Hardy and soon Ubuntu Jaunty



AppScale

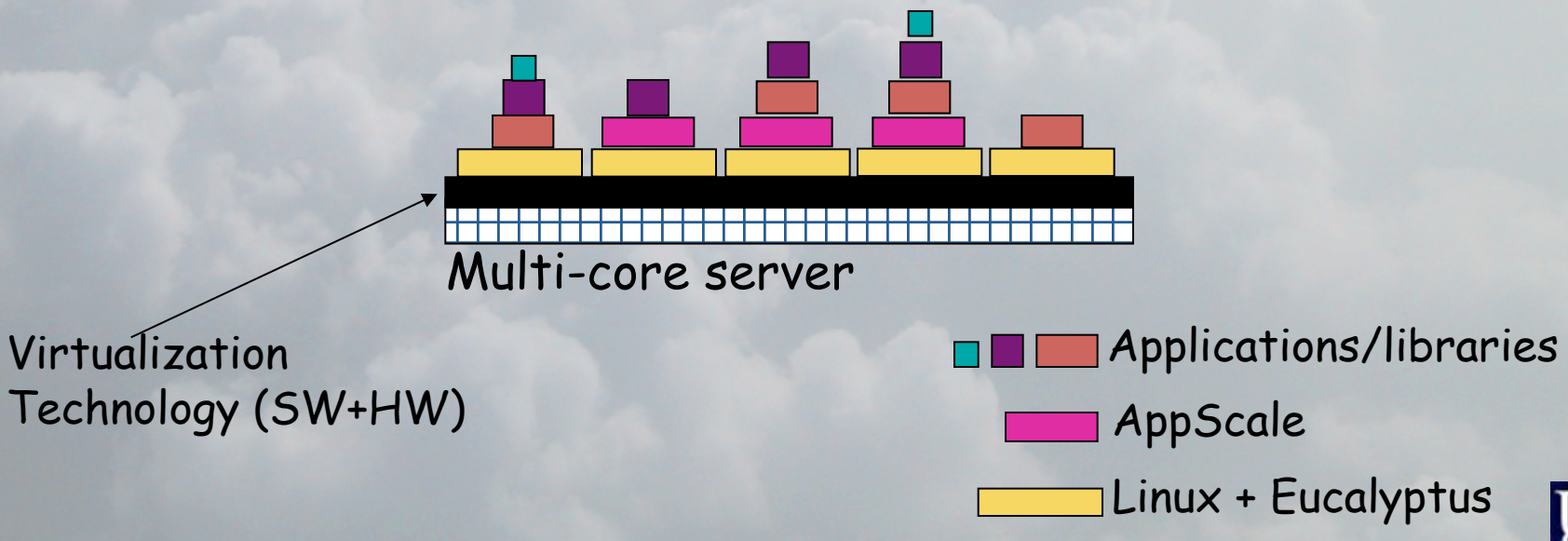

Number of Transactions Completed over Time
appscale-guestbook
google-guestbook
appscale-shell
google-shell
Number of Transactions
Time (seconds)
0
50
100
150
200
250
300
0
20
40
60
80
100
120
140
160

AppScale

- Research and development roadmap
  - Support for Java and additional DBs
  - ASs and DBs grow and shrink according to load and failure
    - Resource monitoring & allocation (SLA support)
      - Automatic and dynamic renegotiation, improved scaling
  - Performance and availability monitoring
    - Capture full-system behavior via sampling
      - For debugging, performance/energy feedback, optimization
  - Administrator/Developer control of
    - Replication of data for fault tolerance
    - Type and amount of system monitoring
    - Parallelism/Concurrency
    - Scaling triggers
    - Sandbox restrictions
    - MapReduce tasks
  - Alternative computation models, e.g. streaming

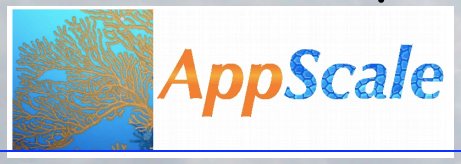
AppScale + Eucalyptus

- Research and development roadmap
  - PaaS integration with ***other cloud fabrics***
  - Paas+IaaS integration (AppScale + Eucalyptus)
    - Resource allocation, specialization/customization, alternative application domains (computationally intensive, data intensive)
    - Isolation/performance tradeoffs



Cloud Computing at UCSB

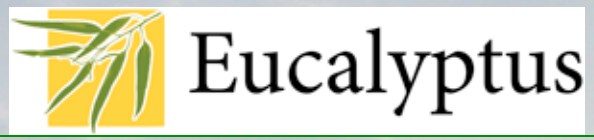
Open-source implementations of popular cloud systems



- Platform-as-a-service (PaaS) framework
- Web services based implementation of **Google AppEngine APIs**
- Runs over Eucalyptus, Amazon EC2, and virtualization layers (Xen/KVM)
- Implements multiple database backends (Hbase, Hypertable, Cassandra, MySQL,…)
- Real use, real users, real impact
  - International user community

http://appscale.cs.ucsb.edu

Lead: Chandra Krintz



- Infrastructure-as-a-service (IaaS) framework
- Web services based implementation of elastic/utility/cloud computing infrastructure
- Linux image hosting via virtualization
- Emulates the **Amazon AWS interface** – applications and tools can't tell the difference
- Real use, real users, real impact
  - Distributed with Ubuntu
  - Large international user community

http://www.eucalyptus.com

Lead: Rich Wolski

- Open, extensible, easy to install/use/maintain, transparent, scalable
- Enables investigation of and experimentation with
  - Real applications in real settings
  - IaaS + PaaS interoperation and integration
- Frameworks for investigation the next generation of distributed systems *technologies, languages, applications, and services*