This presentation is available for download from:
http://ciurana.eu

Mission-Critical Enterprise Cloud Applications

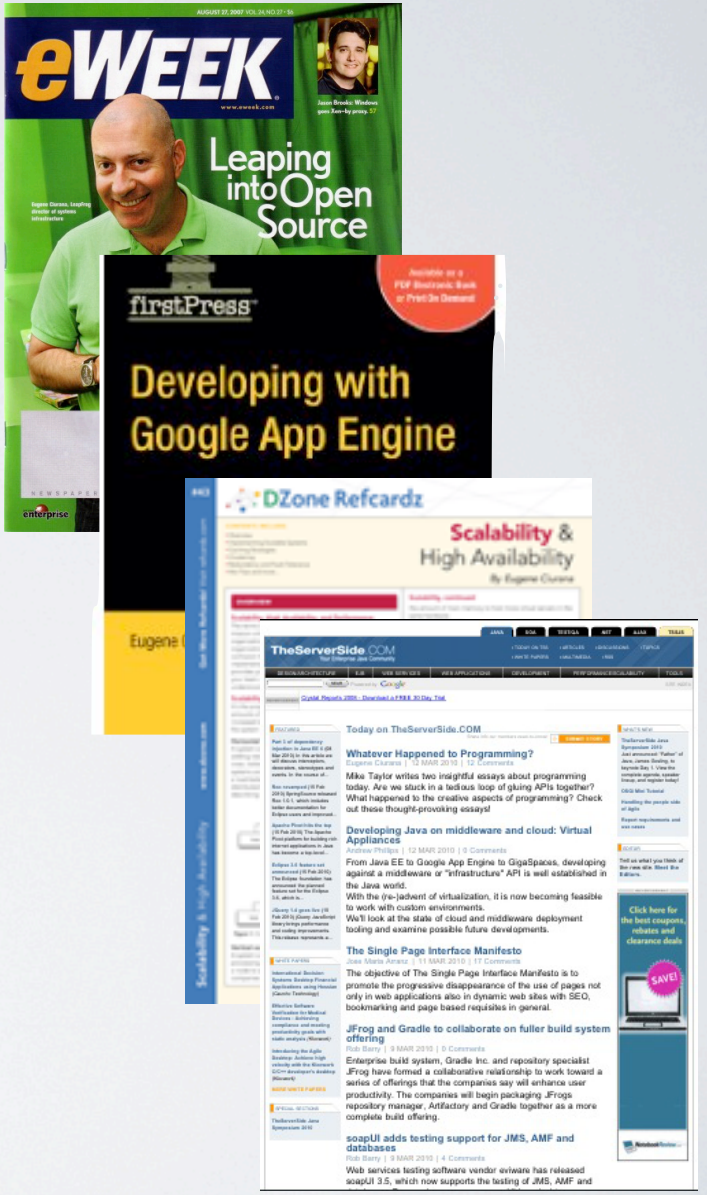
Eugene Ciurana
Open-Source Evangelist
CIME Software Labs

http://ciurana.eu/contact

epicenter

About Eugene...

- 15+ years building mission-critical, high-availability systems
- 14+ years Java work
- Open source evangelist
- Official adoption of open source/ Linux at Walmart worldwide
- State of the art tech for main line of business roll-outs
  - Largest companies in the world
  - Retail
  - Finance
  - Oil
  - Background: robotics to on-line retail



epicenter

This presentation is about...

- How to design, implement, and roll-out cloud/enterprise hybrid applications
- Different cloud architectures
  - PaaS
  - SaaS
  - Infrastructure as a service (IaaS)
- Technologies: ESB, clouds, mini-clouds, Java, App Engine, Chef!/Puppet, etc.
- How cloud technologies lower costs
- Understanding the advantages of:
  - Cloud deployments
  - Hybrid deployments distributed in the cloud and enterprise
  - Hybrid deployments involving Java and non-Java systems

What You'll Learn

- Identify the applications best suited for cloud deployments
- Identify the advantages of Platform-as-a-Service or Software-as-a-Service resources
- Learn the caveats of cross-platform and cross-language integration
- Learn about high-performance alternatives to XML serialization for data exchange
- Learn how to structure event-based, stateless apps for maximum scalability

What is the Cloud Anyway?

- Ask 10 different people, get 10 different answers
- In general, you may use 4 types of cloud offerings
  - Platform as a Service
  - Software as a Service
  - Infrastructure as a Service
  - Pure infrastructure
- Some times you integrate pre-fabricated apps, some times platform, some times both

Cloud Services Features

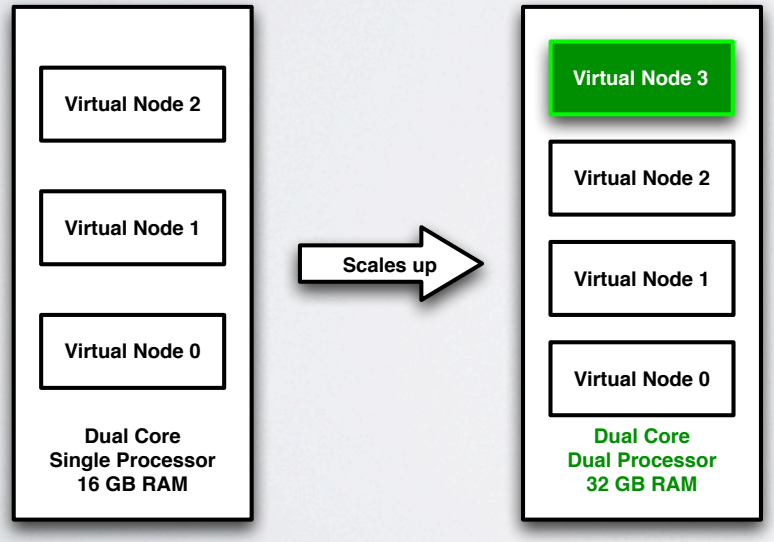
- Quick deployment of prepackaged components
- Uses commodity, virtualized hardware and network resources
  - Amazon Elastic Cloud 2 (EC2) and Simple Storage Service (S3)
  - Google App Engine (Python, Java)
  - Rackspace Cloud Services
- The overall model is “pay as you consume”
- Horizontal scalability is achieved by adding or removing resources as needed
- May host full applications or only services

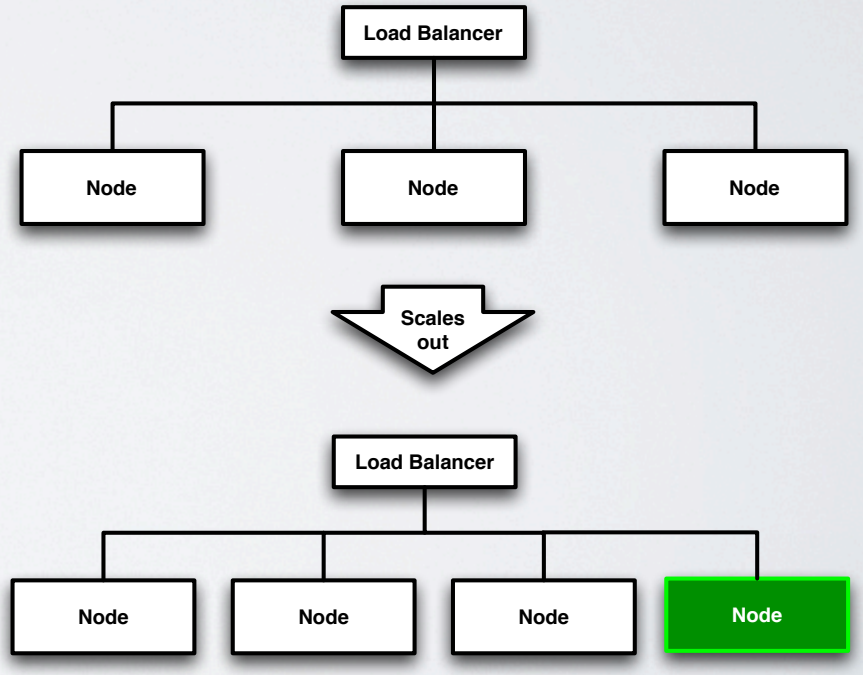
Cloud Services Features

- They could replace the data centre
- Basic administration moves to the application owner
  - It may move away from the IT team - **political fallout**
- For the bean counters… it's an operational expense!
  - Tax advantages
  - Turn on or off as needed
  - In a tight economy, IT infrastructure ends up under the CFO - give the guy options
- Assuming sensible SLAs, the ROI is better than for co-located or company-owned data centres

Scalability and High Availability

- What do **scalability** and **high-availability** mean?
- Scalability: the property of a system to handle bigger amounts of work or to be easily expanded in response to increased demand
  - Vertical
  - Horizontal





epicenter

Scalability and High Availability

- Availability: how the system provides useful resources over a set period of time
  - Uptime != availability
  - Many factors affect it: network, storage, processor, etc.

Availability	Downtime
90%	36.5 days
99%	4 days
99.9%	9 hours
99.99%	53 minutes
99.999%	5.3 minutes
99.9999%	32 se

Can you afford this?

- Amazon S3 Outage (summer 2008)
  - Major companies affected (LeapFrog, eBay, Apple, many others)
  - Amazon's SLA? "Sorry - it won't happen again."

epicenter

SLAs and Availability

- Service Level Agreements drive the architectural and technological decisions
- Cloud systems' scalability characteristics are often pitched
- What is the availability?
  - What is the impact on business?
  - How do you perform disaster recovery?
- What service level are the vendors offering?
  - The higher the availability (3-nines, 4-nines, 5-nines), the higher the cost
  - Co-lo, data centre deployments still make more sense for 4-nines availability and above

Platform and Software as a Service

- Identify the type of implementation after establishing the availability requirements
  - From a vendor?
  - In-house development?
- Software as a Service (SaaS)
  - The vendor provides the full application and is responsible for scaling it and meeting SLAs
  - Integration with the vendor via web services or EDI
  - Examples: GSI Commerce, Salesforce.com
- Platform as a Service (PaaS)
  - The vendor provides the operating system and/or application stack
  - The vendor provides a well-defined API for interacting with the system
  - Examples: Amazon Web Services, Google App Engine, Nirvanix

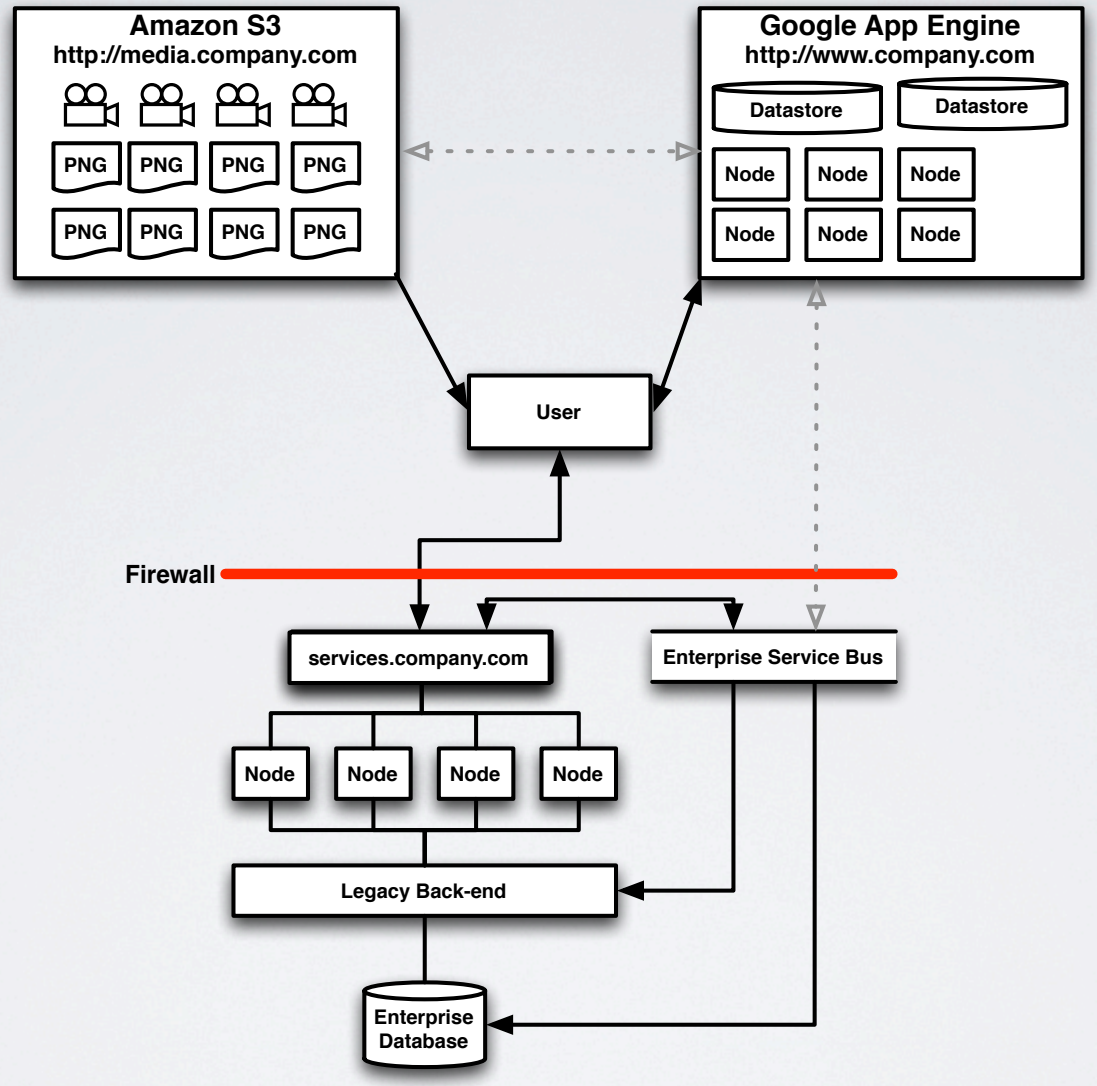
PaaS Systems

- Amazon EC2 and S3
  - Virtual images with various Intel-based operating systems, app servers, databases, etc.
  - Billed hourly + bandwidth
  - Java, PHP, Ruby, other technologies supported
  - “Data centre in the sky”
  - Lower cost than Rackspace or Nirvanix
    - Weaker SLAs?
- Google App Engine
  - The apps are written as callbacks from a virtualized service
  - Datastore ~= database; Memcache ~= caching; most interaction via HTTP
  - Billed on usage
  - “You run on the same infrastructure as Google”
  - Low cost, still evolving, minimal SLAs

A Hybrid Cloud Architecture

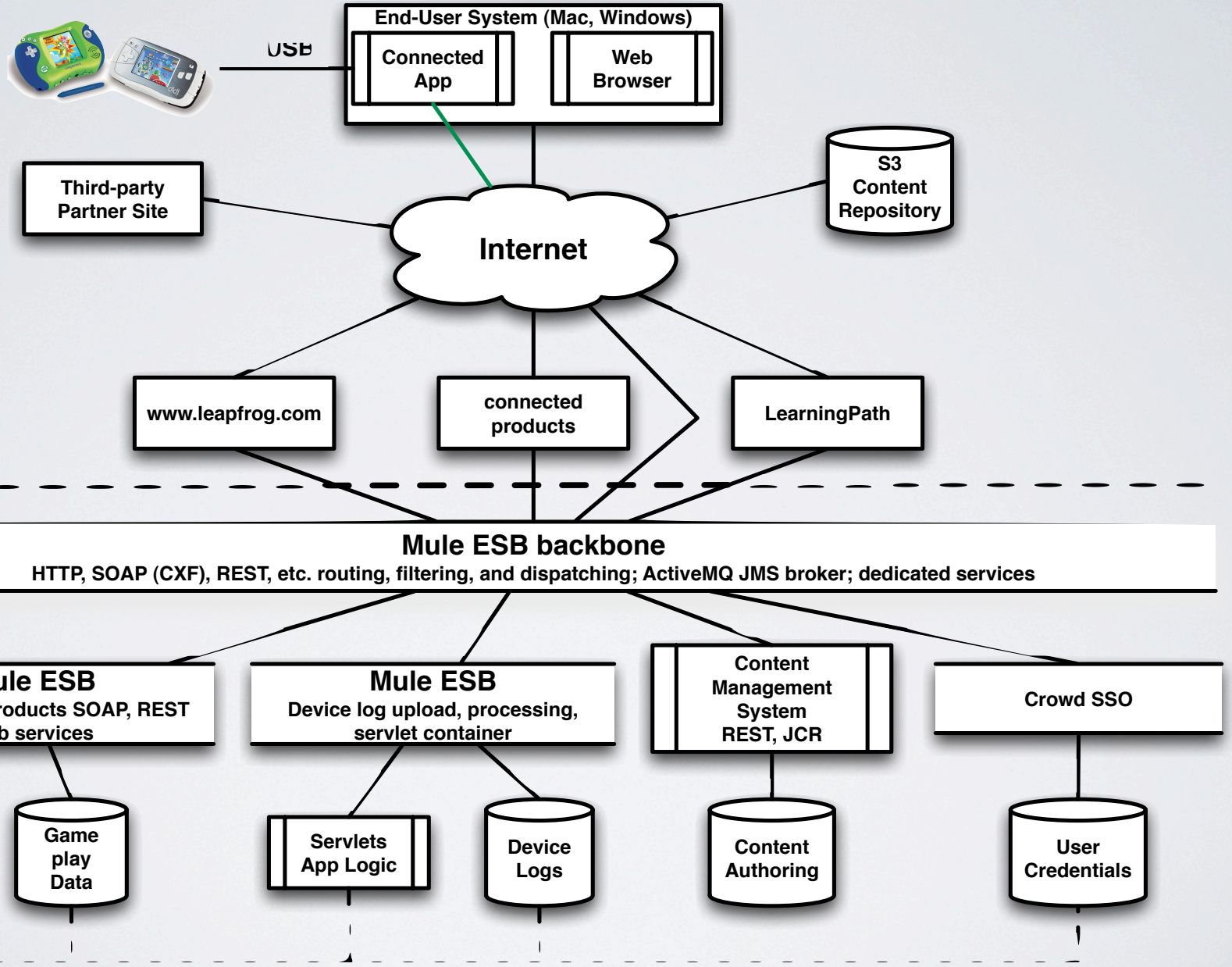
- Many mission-critical systems will live behind the corporate firewall
- The cloud is used for high-load applications and services
- The clouds applications work independently of the data center applications, and vice versa
  - Loose coupling over web services
  - Assume that the data in the cloud is “throwaway” - it’s either consolidated or sourced in the data centre
  - The cloud exists mostly for distribution and scalability

A Hybrid Cloud Architecture



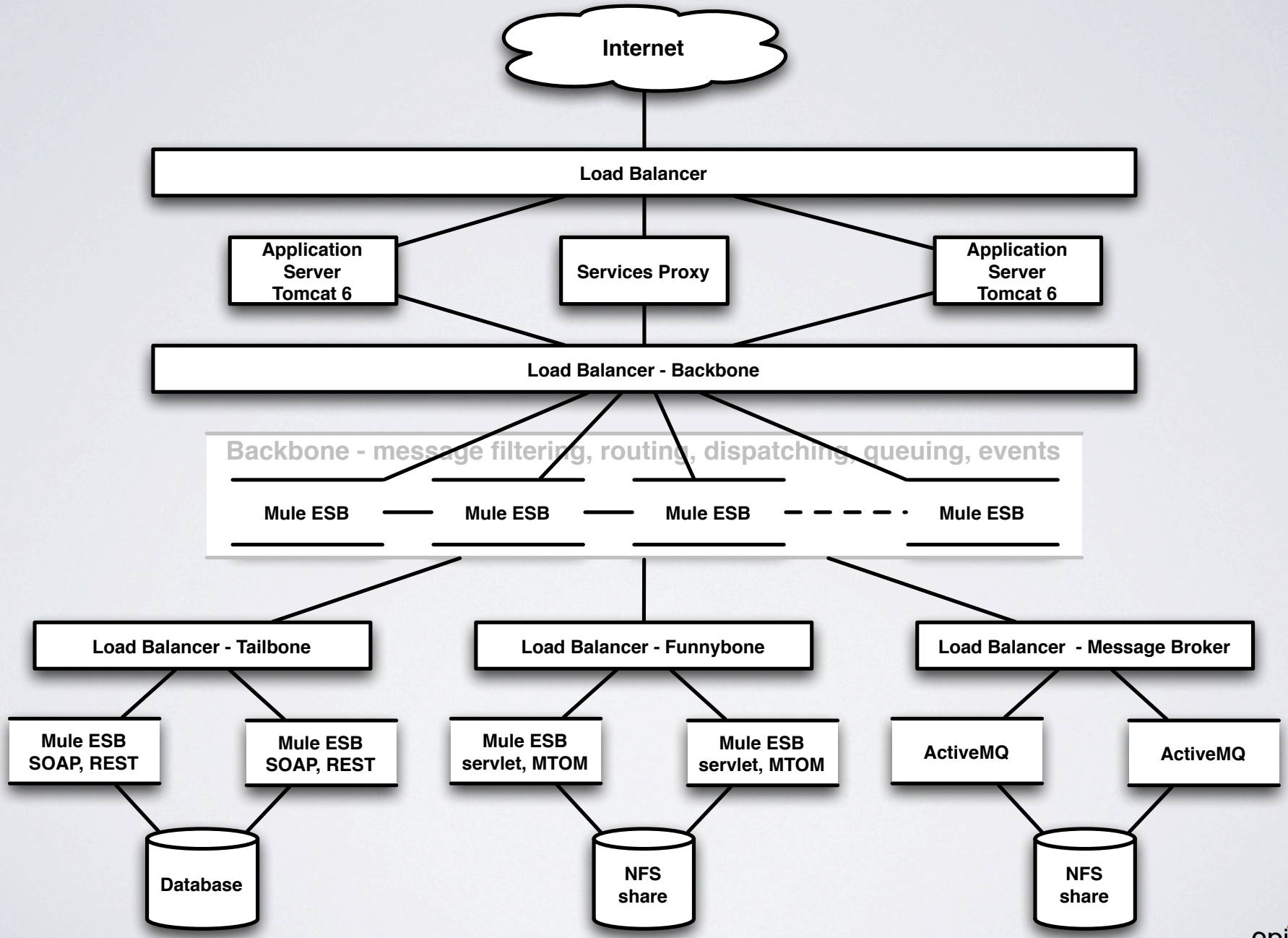
epicenter

A Hybrid Cloud Architecture



epicenter

Which Parts Go To The Cloud?



epicenter

Which Parts Go To The Cloud?

- Depending on the cloud configuration, load balancing may not be necessary
  - Amazon provides Elastic Load Balancers and Elastic IP
  - Google App Engine provides stateless calls only
    - The client itself or Memcache keep state instead
  - Rackspace provides either “elastic” addresses or explicit load balancers
- Enterprise Service Bus and other integration may exist in both the cloud and the data centre
  - Message routing is OK is if latency is below the SLA requirements
  - Strategic partner, application, and services integration may occur 100% on the cloud, with data consolidation behind the firewall

Case Study

- Large consumer and services company
- .Net / Windows servers infrastructure
  - ASP.Net
  - SQL Server
- Two-tier architecture
  - The system “just grew” and became a business
- Server pages, services, media
- 6-month scalability project began in April
  - Implementation complete 25.Sep of the same year

Case Study - Objectives

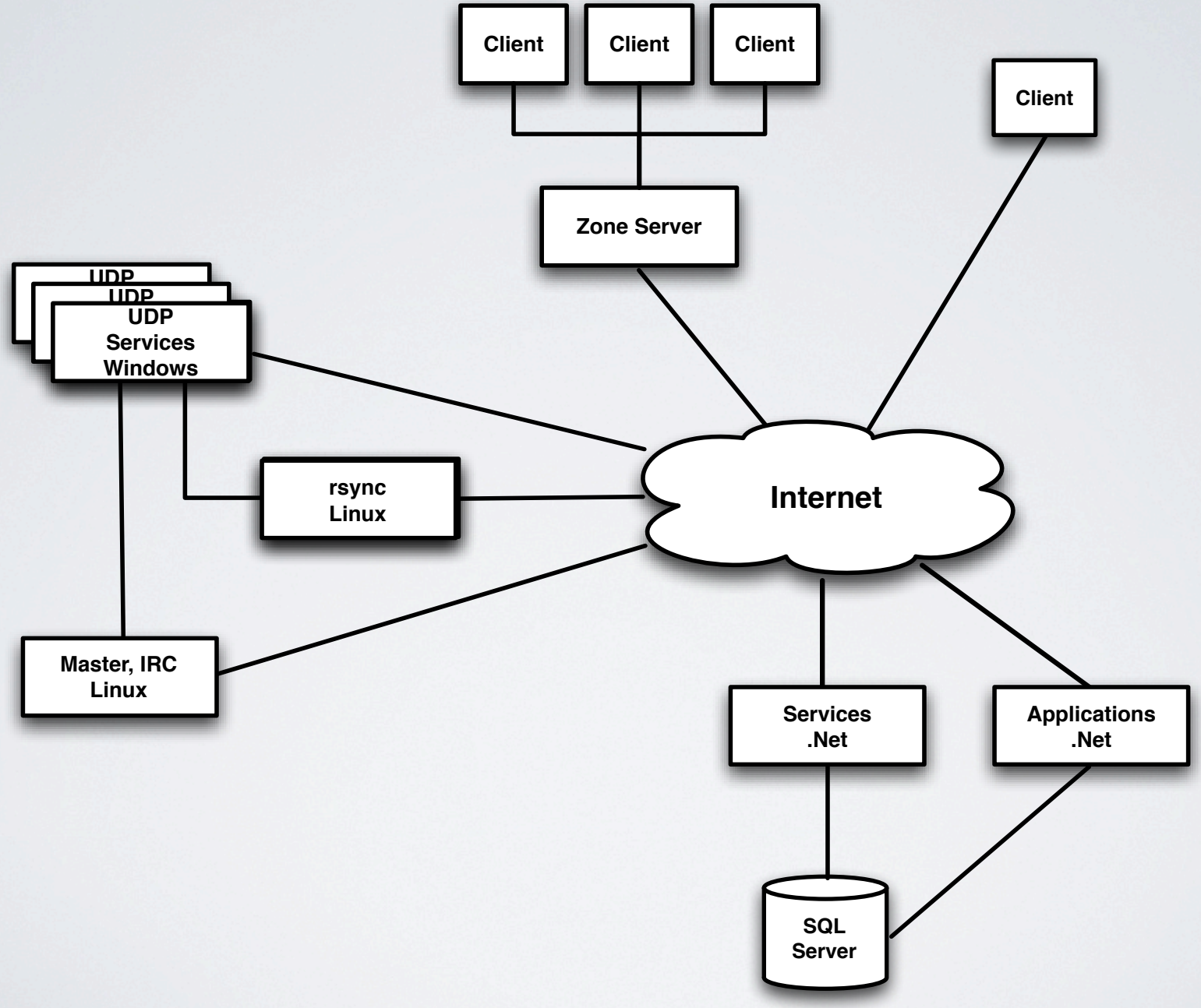
- Stable architecture defined by July
- Low cost
- Build scalability wherever possible
- Optimal data transfer rate for all properties
  - Web systems
  - Media properties
- Meet business requirements and SLAs
  - Hard dates set by the business, no input from the technology team
  - Milestones: 1.July, 10.Oct

Case Study - Initial Configuration

- System grew “as needed” - no planning
- Combines end-user and internal applications in the same server
- It only scales vertically -- and not very well
- Applications and database are tightly coupled
- Application servers and services tightly coupled
- Single environment: from development to production without passing Go nor collecting $200
- Disaster recovery?
  - 4 hours minimum
  - From (stale?) backups

epicenter

Case Study - Initial Configuration



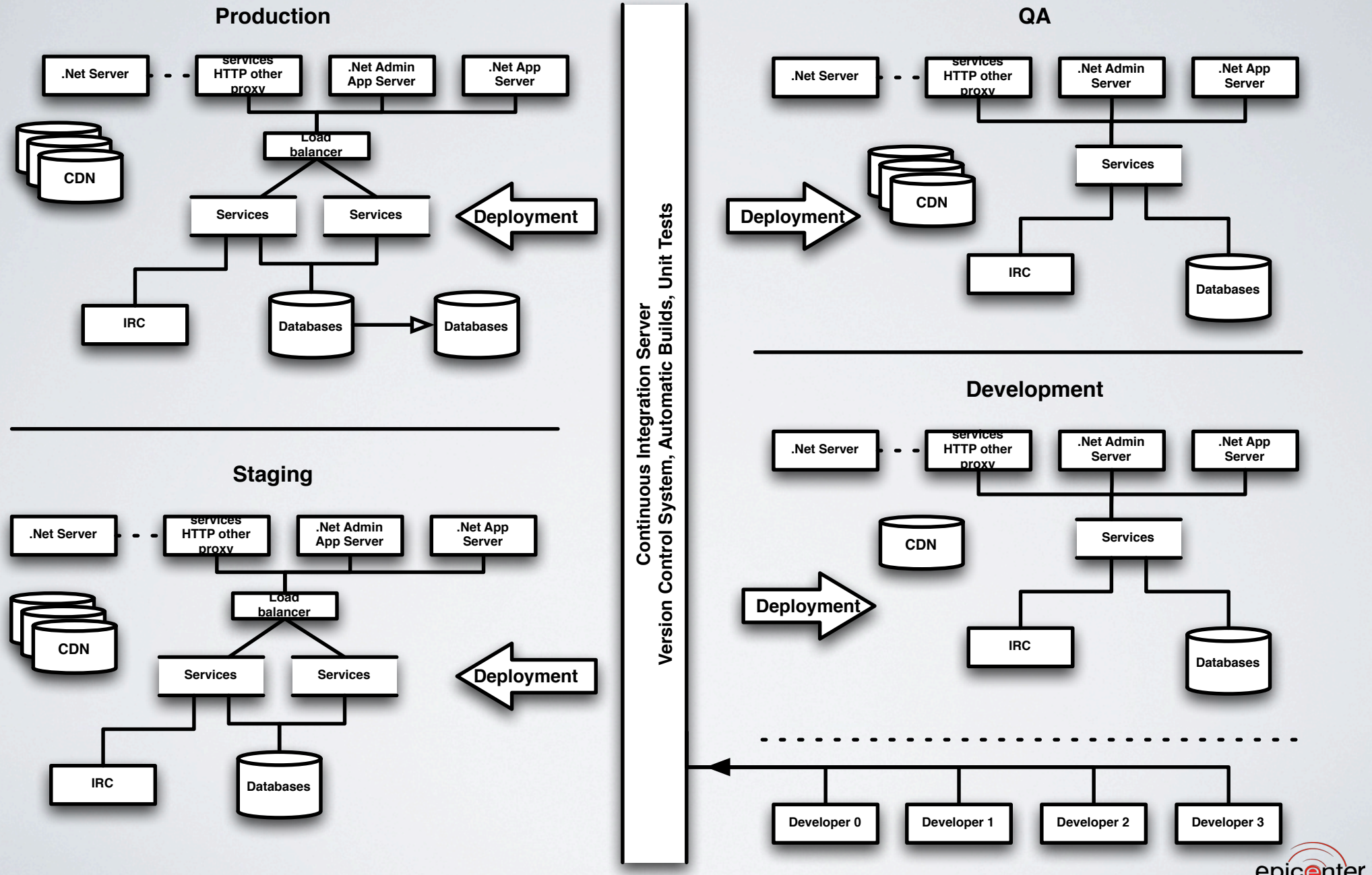
epicenter

Case Study - Phase I Scalability

Case Study - Phase I Scalability

- Introduces a CDN for asset delivery
  - Amazon S3 (optional Cloud Front) for asset delivery
  - Reduces load on company servers and bandwidth costs
- Introduces database replication for production environments
- Establishes a continuous integration environment
  - Set tools for hybrid UNIX/Windows environment
    - UNIX tools take precedence; Cygwin everywhere
  - Improved build/release process

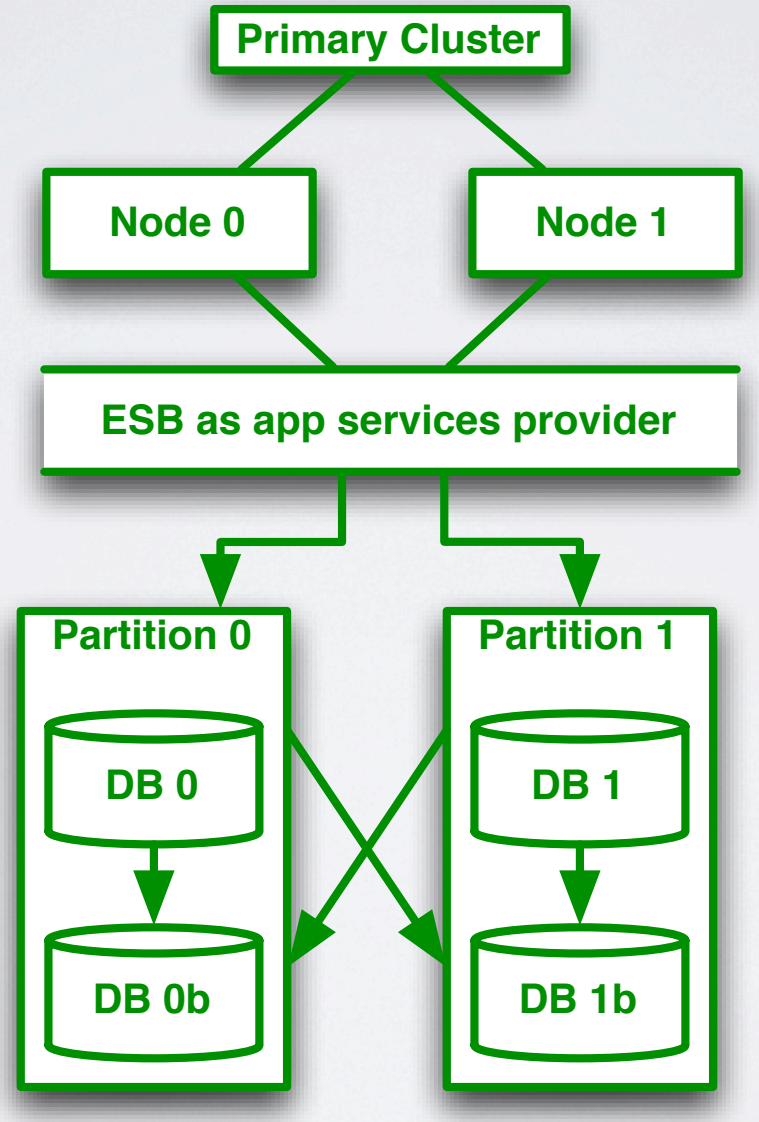
Phase I Scalability - July 2009



epicenter

Case Study - Phase I Scalability

- Fail-over with traditional database replication techniques



epicenter

Case Study - Phase II Cloud Deployment

Case Study - Phase II Cloud Deployment

- Web applications move to a uniform technology (.Net)
- The database and stored procedures are normalized and optimized
- Applications use common resources via Mule ESB and services
  - No more direct calls from apps to databases
  - Business logic is implemented as stateless POJOs
- Software stack is “best of breed”
  - Windows, other servers driven by business requirements
  - Open source software wherever possible

Case Study - Phase II Cloud Deployment

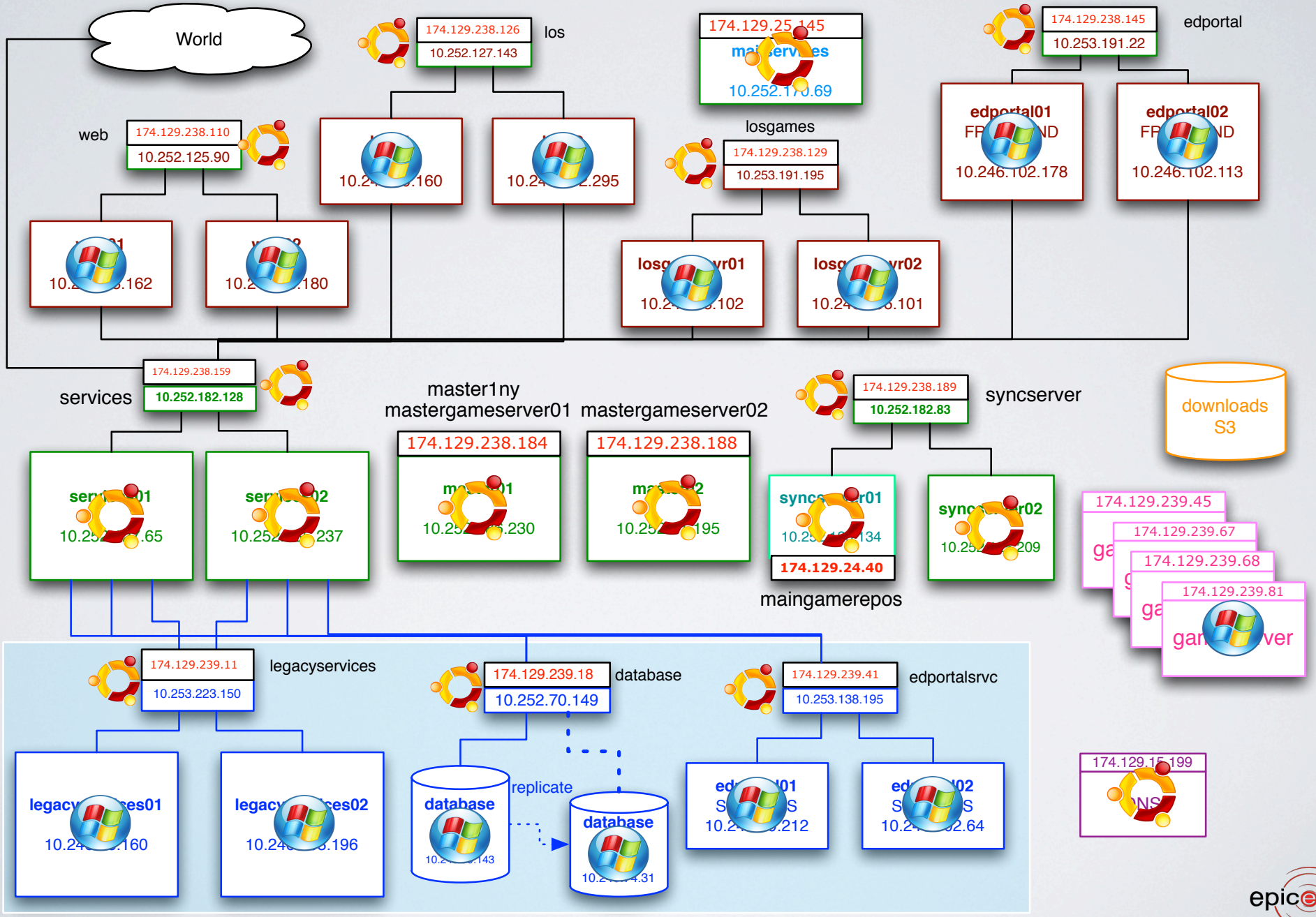
- Web and other RPC services must coexist
  - Different partners use different protocols
  - Mule transformers take care of all the interfaces so that development may scale / continue independently of what happens in other layers
- Bandwidth can be expen$ive!
- Data exchange protocols
  - Clients: custom, XML, JSON
  - Images: HTTP, S3
  - Cloud-to-HQ: custom, XML/SOAP, protocol buffers
  - HQ data centre: XML/SOAP, protocol buffers
- Replication strategy: data centre
  - The cloud isn't trustworthy yet

epicenter

Case Study - Phase II Cloud Deployment

- Deployment involves using an Amazon Machine Image (AMI)
  - Use existing ones from Amazon or third parties
  - Configure your own to meet your software requirements
- AMIs need a post-configuration step in a load-balanced environment
  - Unique file processing (/etc/hosts, /etc/hostname, app configuration, etc.)
  - Necessary intermediate step between launching the AMI and having a useful machine
- Elastic Load Balancer and Elastic IP limitations
  - ELB only works for dynamic IP addresses
  - ELIPs are limited to 5 per account

Phase II Cloud Deployment - Sep. 2009



epicenter

Case Study - Phase II Cloud Deployment

Traditional Hosted Environments

	Setup	Monthly	Bandwidth	Total / year
Production	$1,300	$5,5580	$2,000	$70,260
Staging	$1,300	$5,5580	$ 0	$68,260
QA	$ 600	$1,450	$ 0	$18,000
Dev	$ 600	$1,450	$ 0	$18,000
			Total	**$174,520**

epicenter

Case Study - Phase II Cloud Deployment

Amazon EC2 Configuration

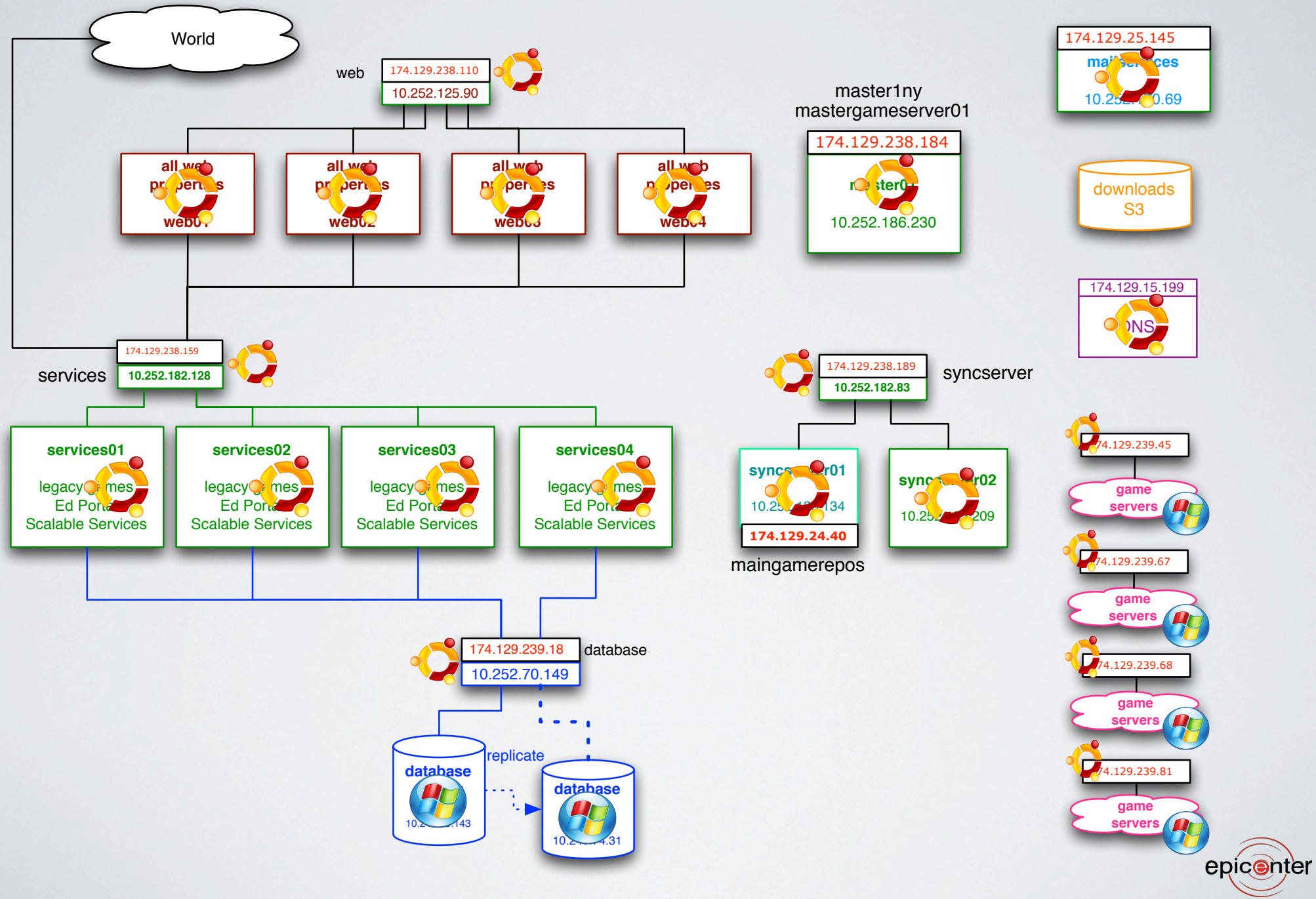
	Setup	Monthly	Bandwidth	Total / year
Production	$ 0	$4,320	$1,200	$66,240
Staging	$ 0	$4,320	$ 0	$51,840
QA	$ 0	$1,080	$ 0	$12,960
Dev	$ 0	$ 0	$ 0	$ 0
			Total	$131,040

$43,000 cheaper

epicenter

Case Study - Phase III
Final Deployment

Phase III Final Deployment - April 2010



Thanks for Coming!

Wanna know more about real life cloud, scalable systems?

Subscribe to the newsletter!

http://ciurana.eu/scalablesystems

http://twitter.com/ciurana technology tweets

Questions?

Eugene Ciurana

Open source evangelist

irishdev@ciurana.eu

+1 415 387 3800