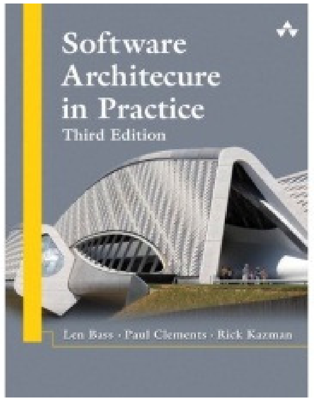
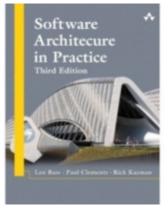
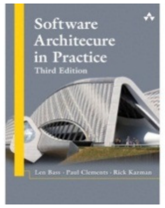


Architectures for the Cloud



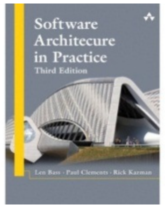
Chapter Outline

- Basic Cloud Definitions
- Service Models and Deployment Options
- Economic Justification
- Base Mechanisms
- Sample Technologies
- Architecting in a Cloud Environment
- Summary



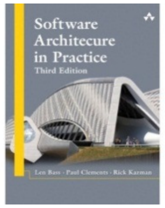
Basic Cloud Definitions (from NIST)

- *On-demand self-service.* A resource consumer can unilaterally provision computing services, such as server time and network storage, as needed automatically without requiring human interaction with each service’s provider.
- *Ubiquitous network access.* Cloud services and resources are available over the network and accessed through standard networking mechanisms that promote use by a heterogeneous collection of clients.
- *Resource pooling.* The cloud provider’s computing resources are pooled.
- *Location independence.* The location of the resources need not be of concern to the consumer of the resources.
- *Rapid elasticity.* *C*apabilities can be rapidly and elastically provisioned.
- *Measured service.* Resource usage can be monitored, controlled, and reported so that consumers of the services are billed only for what they use.
- *Multi-tenancy.* Applications and resources can be shared among multiple consumers who are unaware of each other.



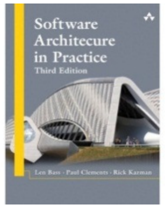
Basic Service Models

- **Software as a Service (SaaS).** The consumer in this case is an end user. The consumer uses applications that happen to be running on a cloud. E.g. mail services or data storage services.
- **Platform as a Service (PaaS).** The consumer in this case is a developer or system administrator. The consumer deploys applications onto the cloud infrastructure using programming languages and tools supported by the provider.
- **Infrastructure as a Service (IaaS).** The consumer in this case is a developer or system administrator. The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications.



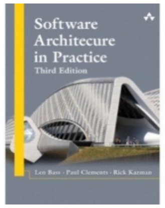
Deployment Models

- *Private cloud.* The cloud infrastructure is owned solely by a single organization and operated solely for applications owned by that organization.
- *Public cloud.* The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.
- *Community cloud.* The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns.
- *Hybrid cloud.* The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities.



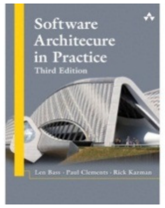
Economic Justification

- Economies of scale
- Utilization of equipment
- Multi-tenancy



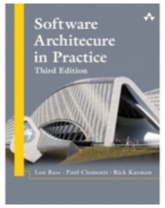
Economies of Scale

- Large data centers are cheaper to operate (per unit measure) than small data centers.
- *Large* in this context means 100,000+ servers
- *Small* in this context means <10,000 servers.



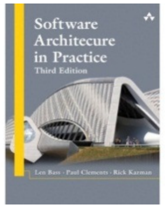
Reasons for Economies of Scale

- *Cost of power.* The cost of electricity to operate a data center currently is 15 to 20 percent of the total cost of operation.
- Per-server power costs are lower in large data centers
  – Sharing of items such as racks and switches.
  – Negotiated prices. Large power users can negotiate significant discounts.
  – Geographic choice. Large data centers can be located where power costs are lowest.
  – Acquisition of cheaper power sources such as wind farms and rooftop solar energy.
- *Infrastructure labor costs. More efficient utilization of system administrators*
  – Small data center administrators service ~150 servers.
  – Large data center administrators service >1000 servers.



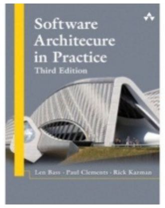
More Reasons for Economies of Scale

- *Security and reliability.* Maintaining a given level of security, redundancy, and disaster recovery essentially requires a fixed level of investment. Larger data centers can amortize that investment over their larger number of servers.
- *Hardware costs.* Operators of large data centers can get discounts on hardware purchases of up to 30 percent over smaller buyers.



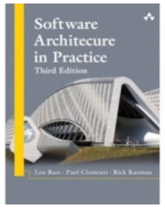
Utilization of Equipment

- Use of virtualization technology allows for easy co-location of distinct applications and their associated operating systems on the same server hardware. The effect of this co-location is to increase the utilization of servers.
- Variations in workload can be managed to increase utilization.
  - *Random access.* End users may access applications randomly. The more likely that the randomness of their accesses will end up imposing a uniform load on the server.
- *Time of day.*
  - Co-locate those services that are workplace related with those that are consumer related.
  - Consider time differences among geographically distinct locations.
- *Time of year.* Consider yearly fluctuations in demand.
  - Holidays, tax preparation season
- *Resource usage patterns.* Co-locate heavier CPU services with heavier I/O services.
- *Uncertainty.* Consider spikes in usage.
  - news events, marketing events, sporting events



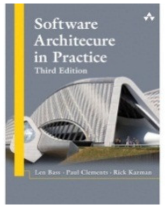
Multi-tenancy

- Some applications such as salesforce.com use a single application for multiple different consumers.
- This reduces costs by reducing costs of
  - Help desk support
  - Upgrade once, simultaneously, for all consumers
  - Single version of the software from a development and maintenance perspective.



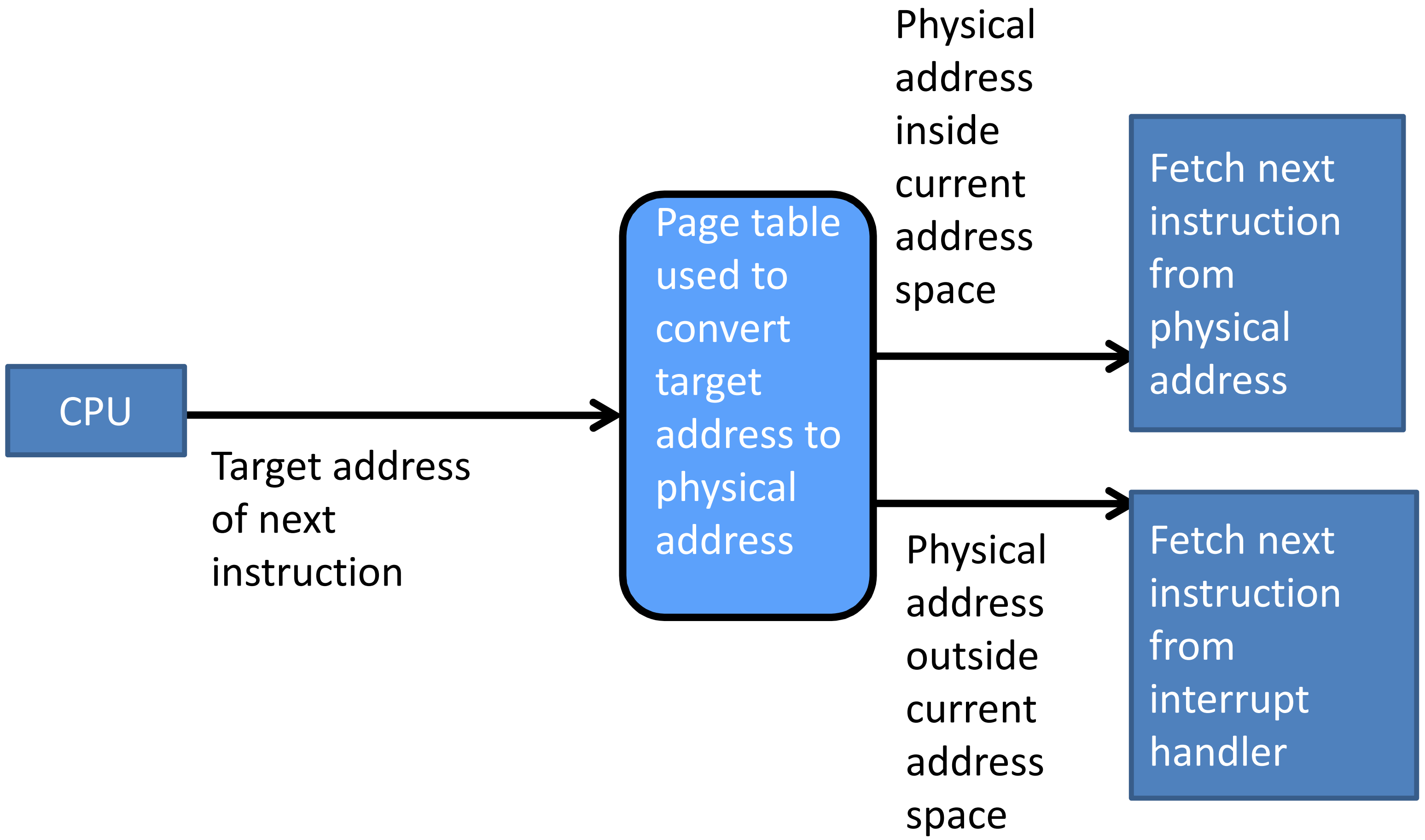
Basic Mechanisms

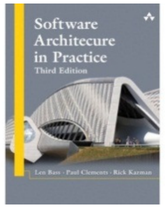
- Hypervisor
- Virtual Machine
- File system
- Network



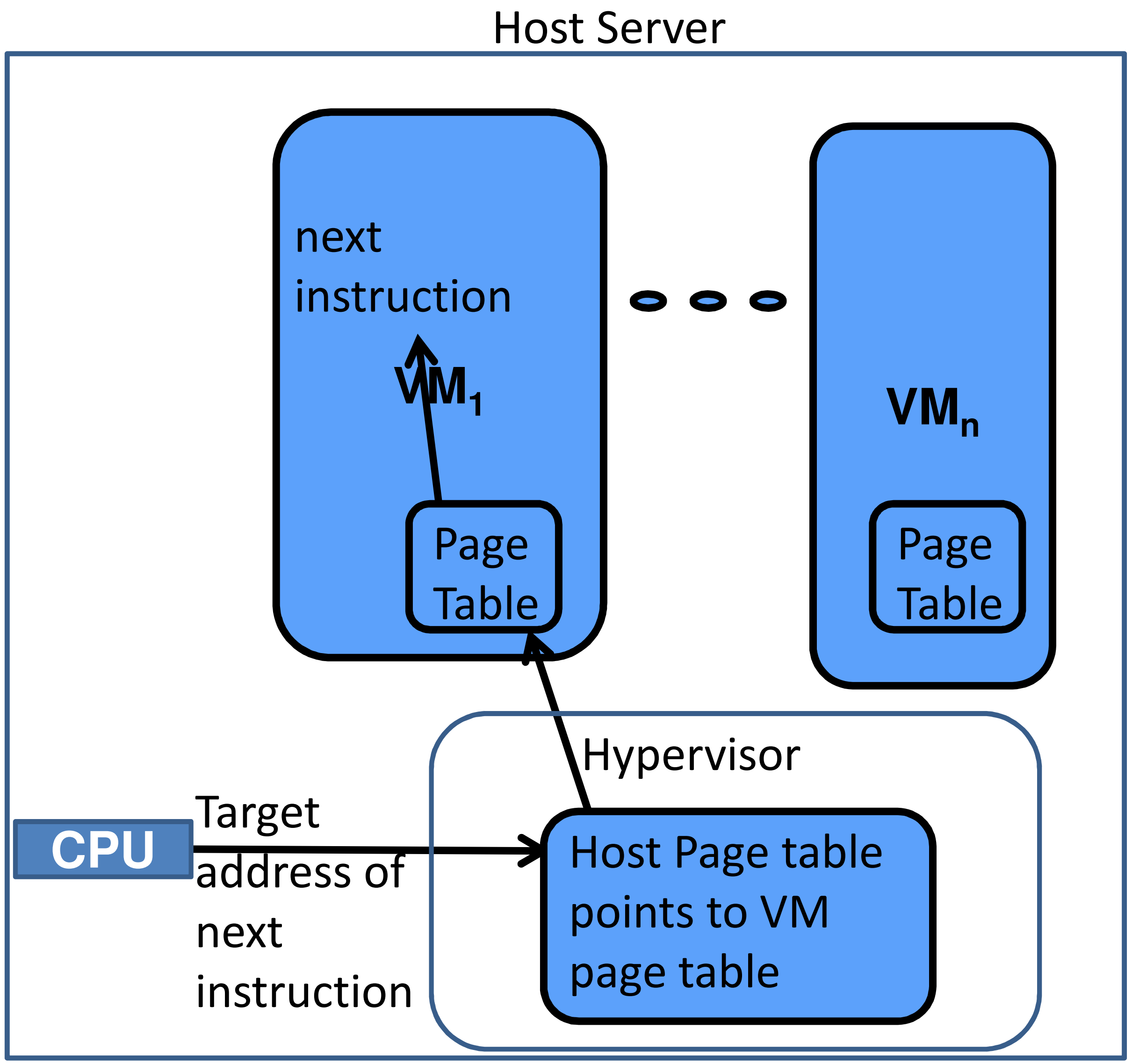
Virtual Memory Page Table

Virtual memory for non-virtualized application

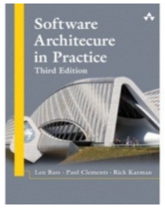

CPU
Target address of next instruction
Page table used to convert target address to physical address
Physical address inside current address space
Fetch next instruction from physical address
Physical address outside current address space
Fetch next instruction from interrupt handler



Hypervisor Manages Virtualization

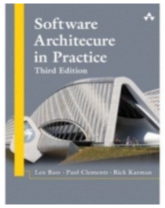


Host Server
next instruction
VM$_1$
Page Table
VM$_n$
Page Table
Hypervisor
CPU
Target address of next instruction
Host Page table points to VM page table



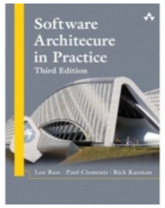
Virtual Machine

- A virtual machine has an address space isolated from any other virtual machine.
- Looks like a bare metal machine from the application perspective.
- Assigned an IP address and has network capability.
- Can be loaded with any operating system or applications that can execute on the processor of the host machine.

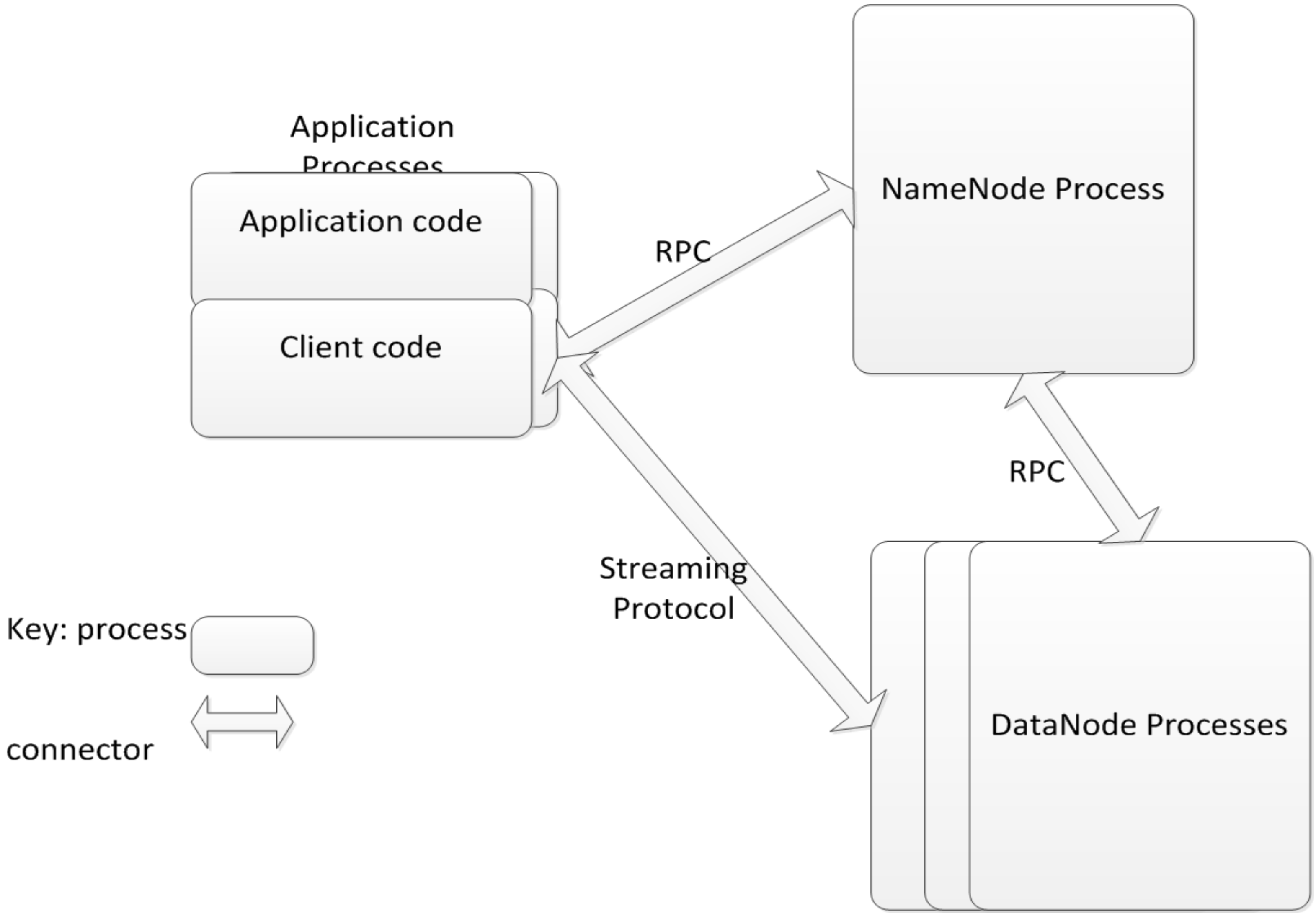


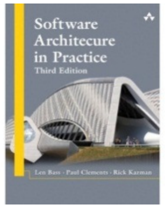
File System

- Each virtual machine has access to a file system.
- We will present HDFS (Hadoop Distributed File System) – a widely used open source cloud file system.
- We describe how HDFS uses redundancy to ensure availability.


Software Architecure in Practice
Third Edition
Len Bass · Paul Clements · Rick Kazman

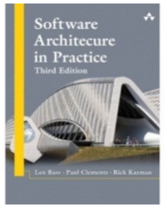
HDFS Components




Software Architecure in Practice
Third Edition
Len Bass · Paul Clements · Rick Kazman

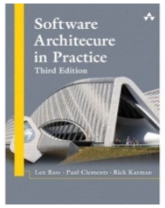
HDFS Write – Sunny Day Scenario

- Application writes as to any file system
- Client buffers until it gets 64K block
- Client informs NameNode it wishes to write a new block
- NameNode returns list of three DataNodes to hold block
- Client sends block to first DataNode and informs DataNode of other two replicas.
- First DataNode writes block and sends it to second DataNode. Second DataNode writes block and sends it to last DataNode.
- Each DataNode reports to client when it has completed its write
- Client commits write to NameNode when it has heard from all three DataNodes.



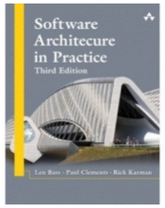
HDFS Write – Failure Cases

- Client fails
  - Application detects and retries
  - Write is not complete until committed by Client
- NameNode fails
  - Backup NameNode takes over
  - Log file maintained to avoid losing information
  - DataNodes maintain true list of which blocks they each have
  - Client detects and retries
- DataNode fails
  - Client (or earlier DataNode in pipeline) detects and asks NameNode for different DataNode.
- Since each block is replicated three times, a failure in a DataNode does not lose any data.



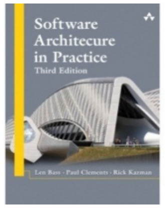
Network

- Every Virtual Machine is assigned an IP address.
- Every message using TCP/IP includes IP address in header.
- Gateway for cloud can adjust IP address for various purposes.



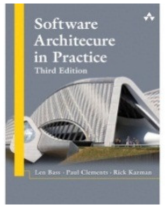
Sample Technologies

- IaaS
- PaaS
- DataBases

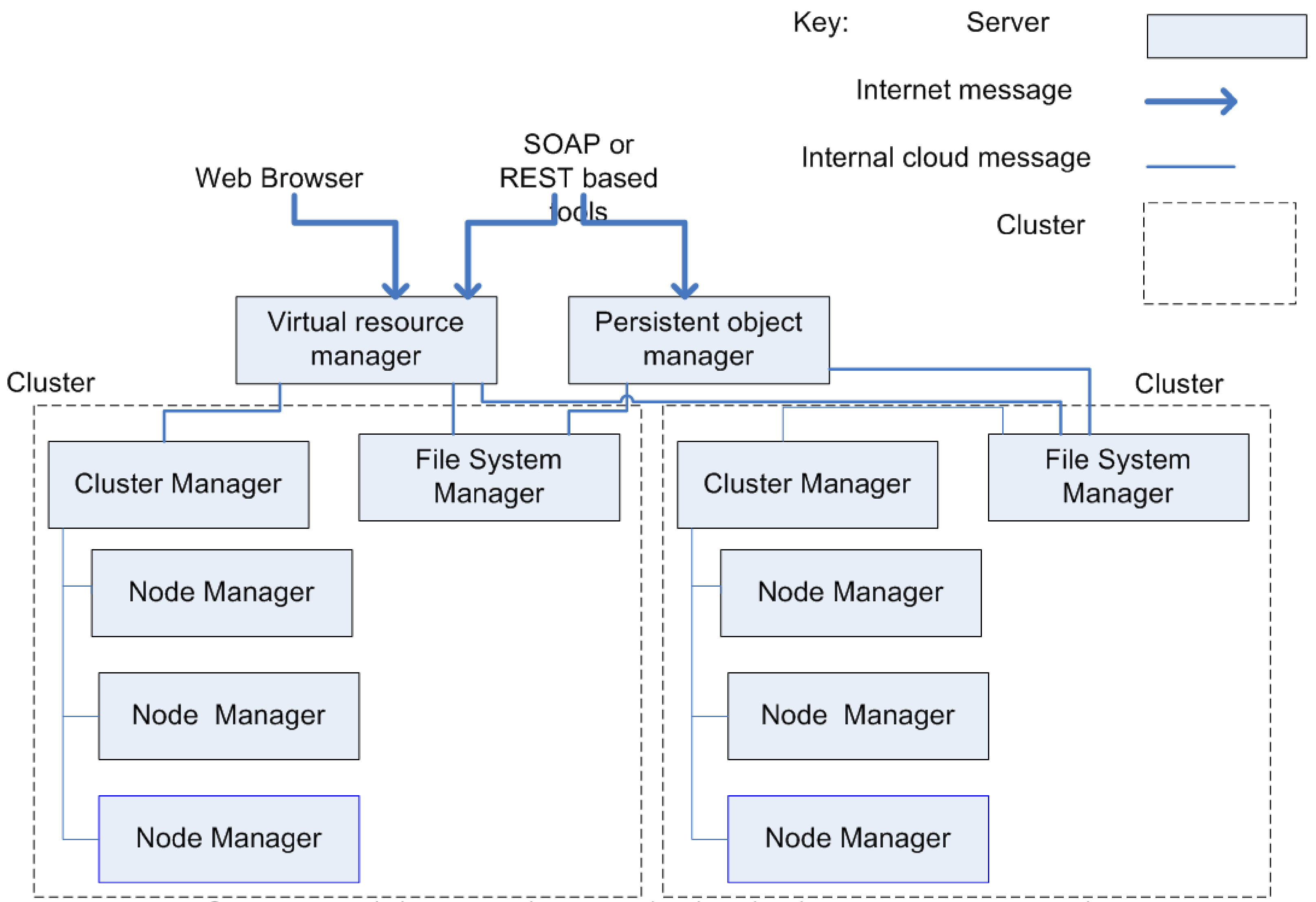


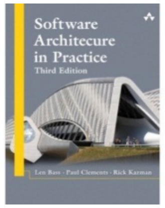
IaaS

- An arrangement of servers that manages the base technologies.
  - Servers are arranged in clusters
  - May be thousands of servers in a cluster
  - Some servers are used as the infrastructure of the IaaS
  - Every server has a hypervisor as its base.



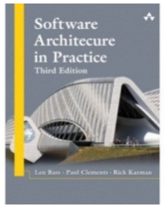
IaaS Architecture





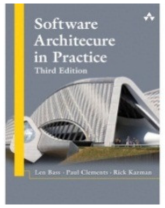
IaaS Architecture Components

- Cluster Manager responsible for managing each cluster
- Persistent Object Manager manages persistence
- Virtual Resource Manager manages other resources. It acts as a gateway for messages.
- The File System Manager is similar to HDFS. It manages the network wide file system.



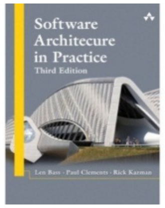
Services Provided by IaaS

- Automatic reallocation of IP addresses in the case of a failure of the underlying virtual machine instance.
- Automatic Scaling. Create or delete new virtual machines depending on load.



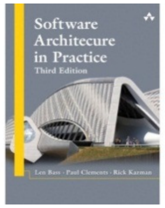
PaaS

- Provides an integrated stack for developer.
- E.g. LAMP stack
  - Linux, Apache, MySQL, Python
- The developer writes code in Python and the PaaS manages assignment to underlying layers of the stack.



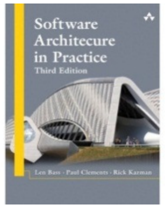
Databases

- Why relational databases came into question
  - Massive amounts of data are collected from web systems. Much of this data is processed sequentially and so RDBMSs introduce overhead, especially during creation and maintenance.
  - The CAP Theorem shows that it is not possible to simultaneously achieve consistency, availability, and partitioning.
  - The relational model is not the best model for some applications.
- Caused the introduction of new data models
  - Key-value
  - Document centric



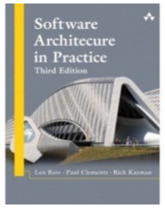
Key Value – HBase

- One column designated as a key. The others are all values
- No schema so data can have key + any other values. The values are identified by their variable name.
- Data values are also time stamped
  - Hbase does not support transactions. Time stamps are used to detect collisions after the fact.



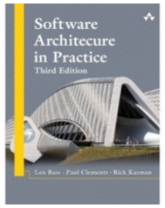
Document Centric – MongoDB

- Stores objects rather than data
- Access data through containing object
- Objects can also contain links to other objects
- No concept of primary or secondary index. A field is indexed or it is not.



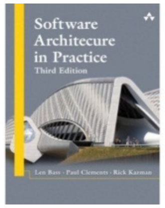
What is Omitted From These DBs

- Transactions. No locking is performed. The application must detect interference with other users.
- Schemas. No predefined schemas. The application must use correct name.
- Consistency. The CAP theorem says something must give. Usually consistency is replaced by "eventual consistency"
- Normalization and Joins. Performing a join requires that the join field is indexed. Because there is not a guaranteed index field, joins cannot be performed. This means normalization of tables is not supported.



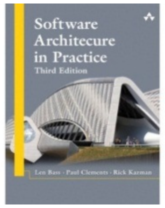
Architecting in a Cloud Environment

- Quality attributes that are different in a cloud
  - Security
  - Performance
  - Availability



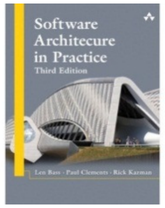
Security

- Multi-tenancy introduces additional concerns over non-cloud environments.
  - Inadvertent information sharing. Possible that information may be shared because of shared use of resources. E.g. information on a disk may remain if the disk is reallocated.
  - A virtual machine “escape”. One user can break the hypervisor. So far, purely academic.
  - Side channel attacks. One user can detect information through monitoring cache, for example. Again, so far, purely academic.
  - Denial of Service attacks. One users can consume resources and deny them to other users.
- Organizations need to consider risks when deciding what applications to host in the cloud.



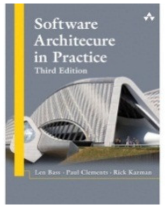
Performance

- Auto-scaling provides additional performance when load grows.
  - Response time for new resources may not be adequate
  - Architects need to be aware of resource requirements for applications
    - Build that knowledge into the applications
    - May applications self aware so that they can be proactive with respect to resource needs.



Availability

- Failure is a common occurrence in the cloud
  - With 1000s of servers, failure is to be expected
- Cloud providers ensure that the cloud itself will remain available with some notable exceptions.
- Application developers must assume instances will fail and build in detection and correction mechanisms in case of failure.



Summary

- The cloud provides a new platform for applications with some different characteristics.
- Architect needs to know how a cloud cluster works and pay special attention to
  - Security
  - Performance
  - Availability