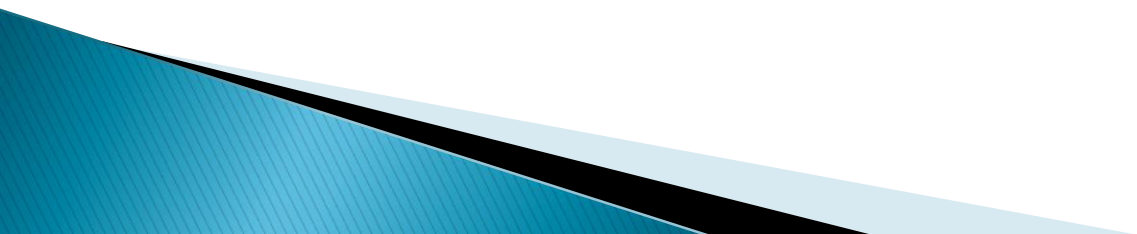
CLOUD COMPUTING

Authors

- Principles and Paradigms by Rajkumar Buyya, James Broberg and Andrzej M. Goscinski, Wiley, 2011.
- Distributed and Cloud Computing, Kai Hwang, Geoffery C.Fox, Jack J.Dongarra, Elsevier, 2012

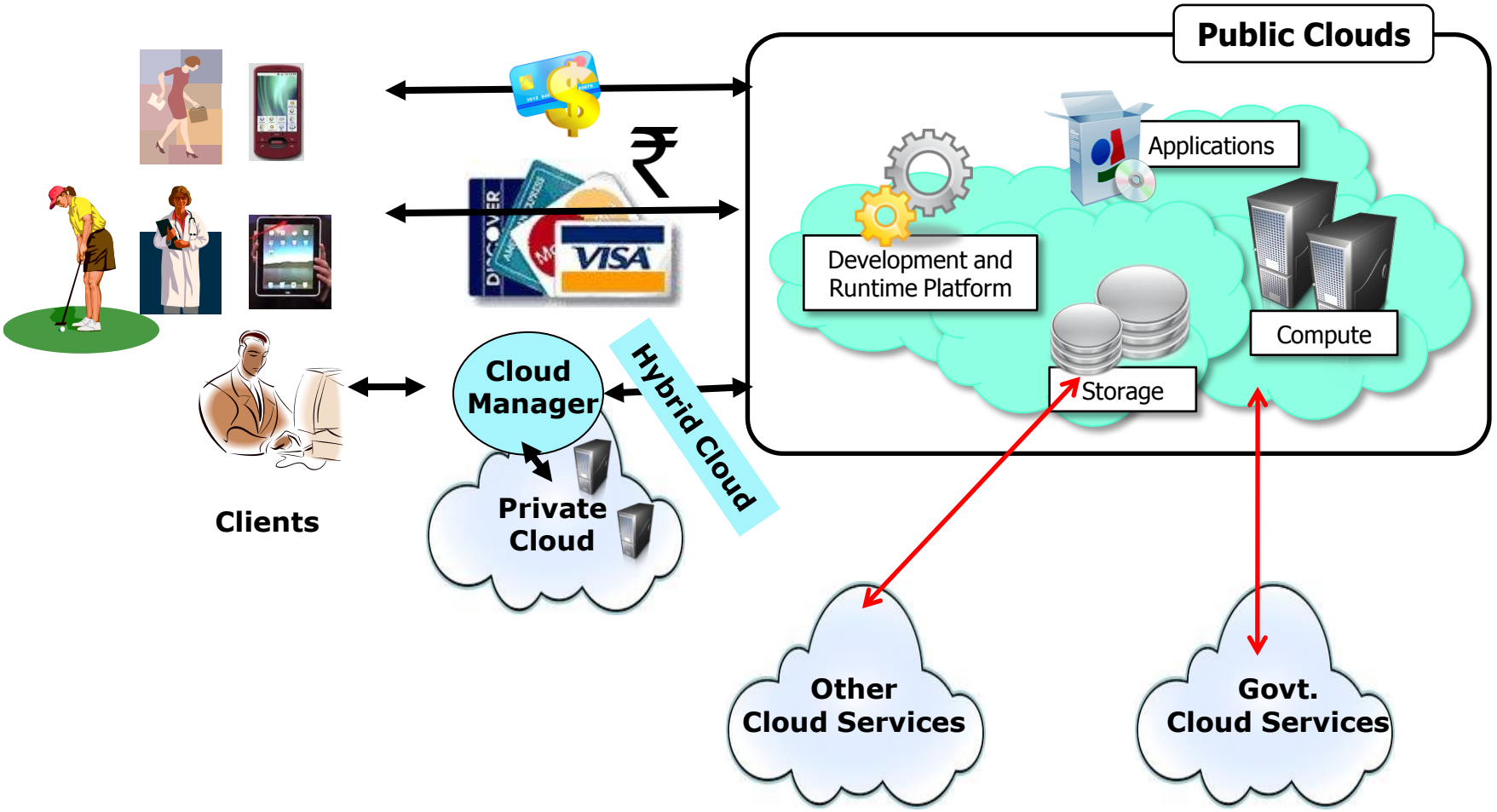
System modelling, Clustering, Virtualisation

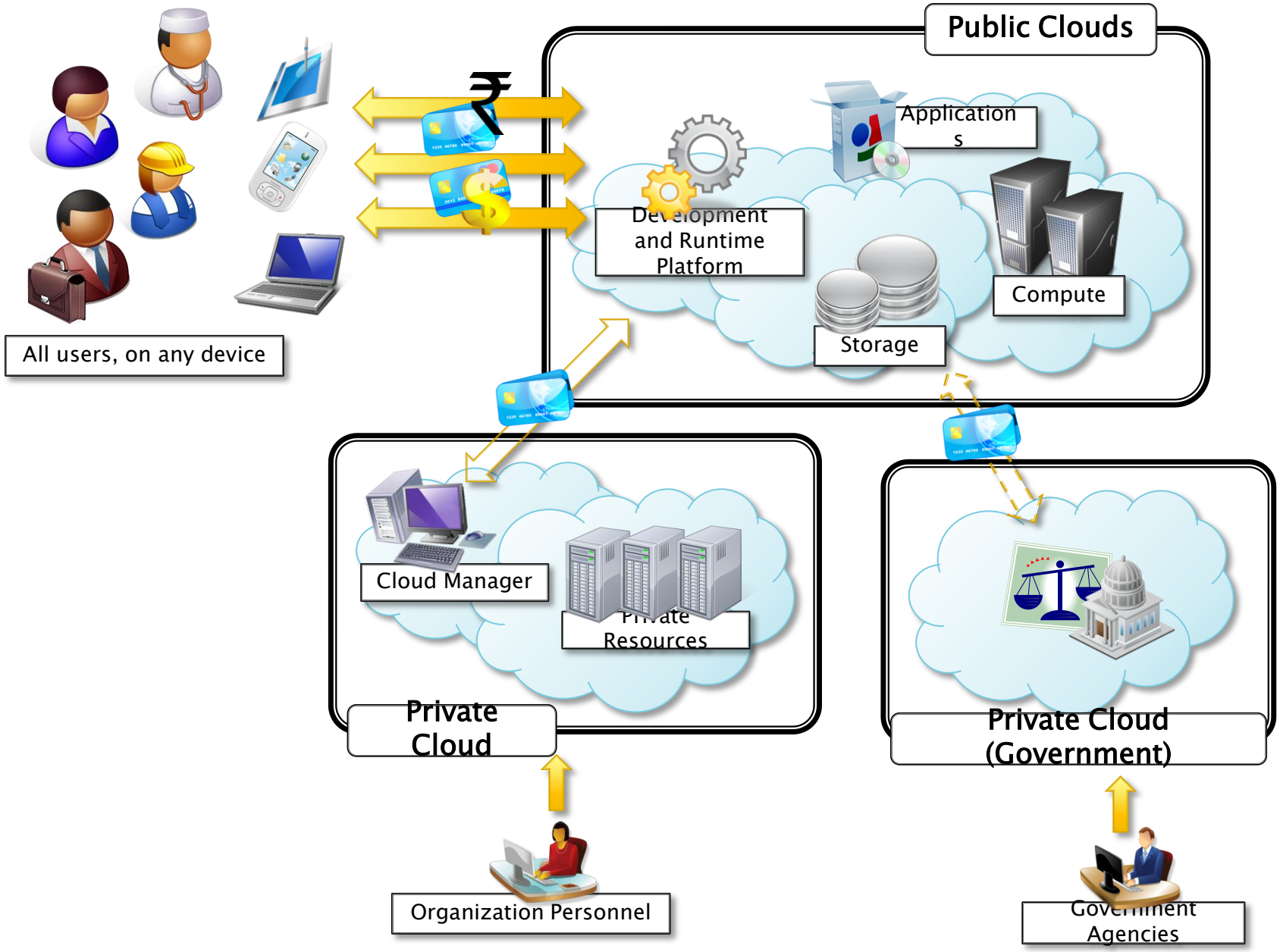



I need to grow my infrastructure, but I do not know for how long…
I have a lot of infrastructure that I want to rent …
I have a surplus of infrastructure that I want to make use of
I cannot invest in infrastructure, I just started my business….
I have infrastructure and middleware and I can host applications
Global Cloud Marketplace
I want to focus on application logic and not maintenance and scalability issues
I have infrastructure and provide application services
I want to access and edit my documents and photos from everywhere..

Subscription-Oriented Cloud Services:
X{compute, apps, data, ..} as a Service (..aaS)

Manjrasoft




Public Clouds
Application s
Development and Runtime Platform
Compute
Storage
All users, on any device
Cloud Manager
Private Resources
Private Cloud
Private Cloud (Government)
Organization Personnel
Government Agencies

Cloud Deployment Models

Public/Internet Clouds

* 3rd party, multi-tenant Cloud infrastructure & services:

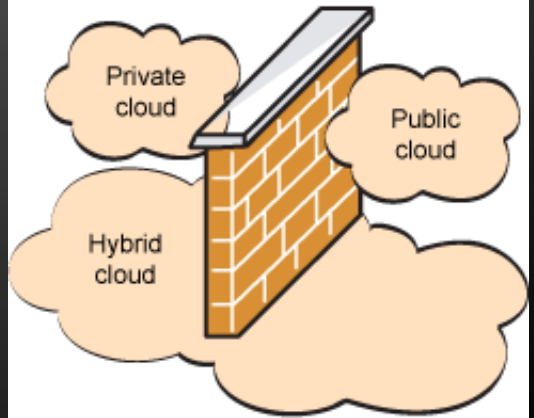
* available on subscription basis to all.

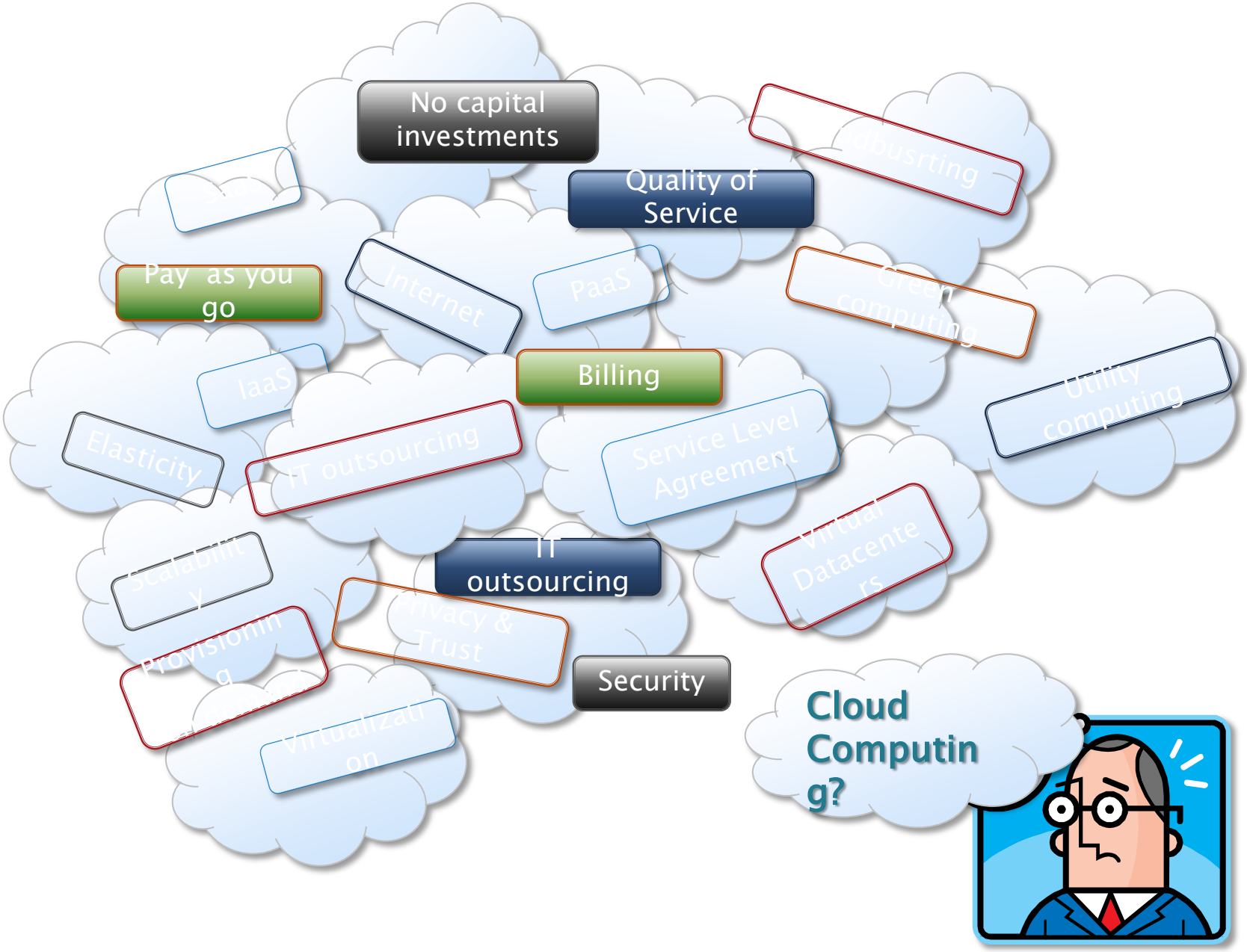
Private/Enterprise Clouds

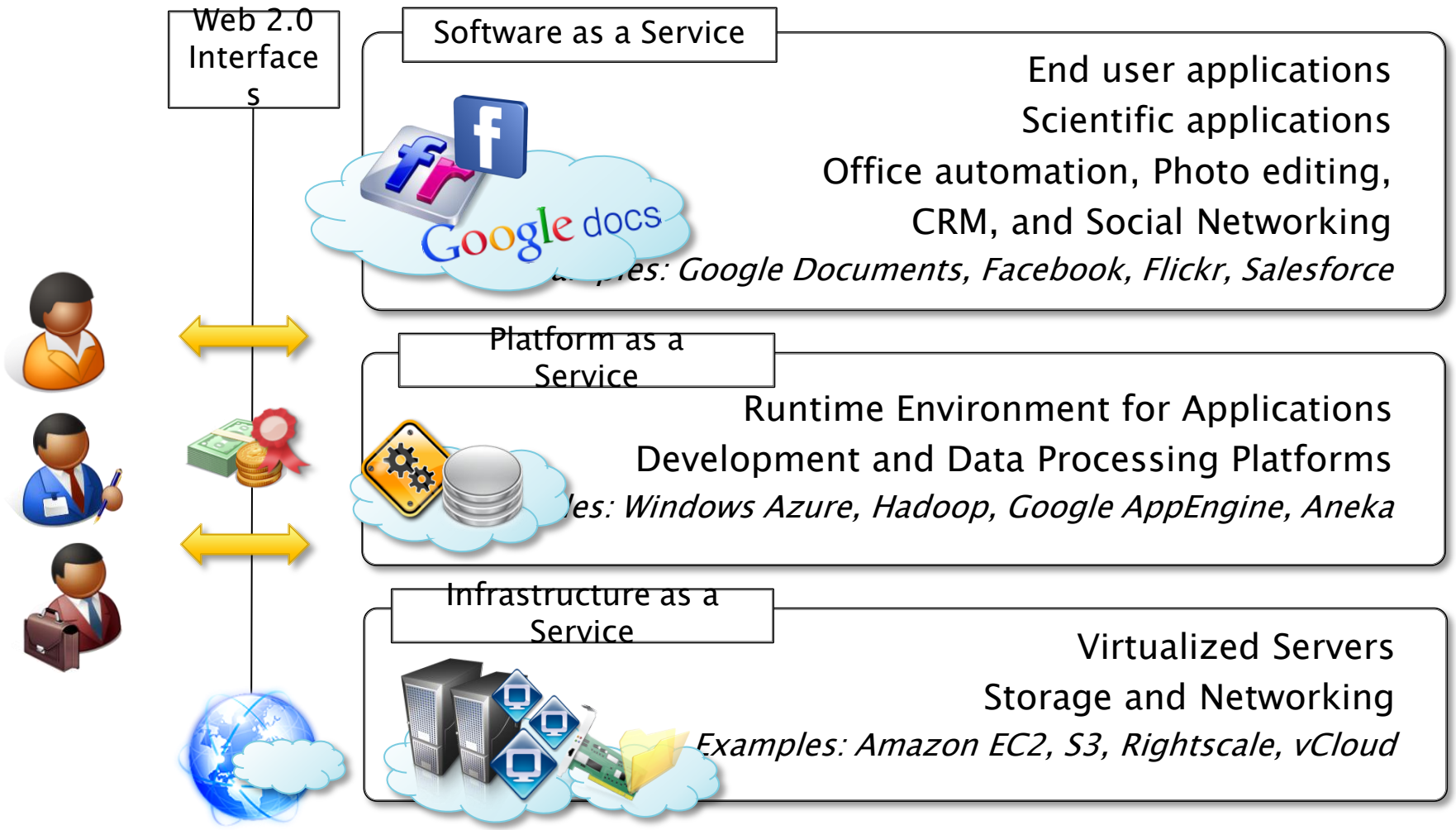
* A public Cloud model within a company's own Data Center / infrastructure for internal and/or partners use.

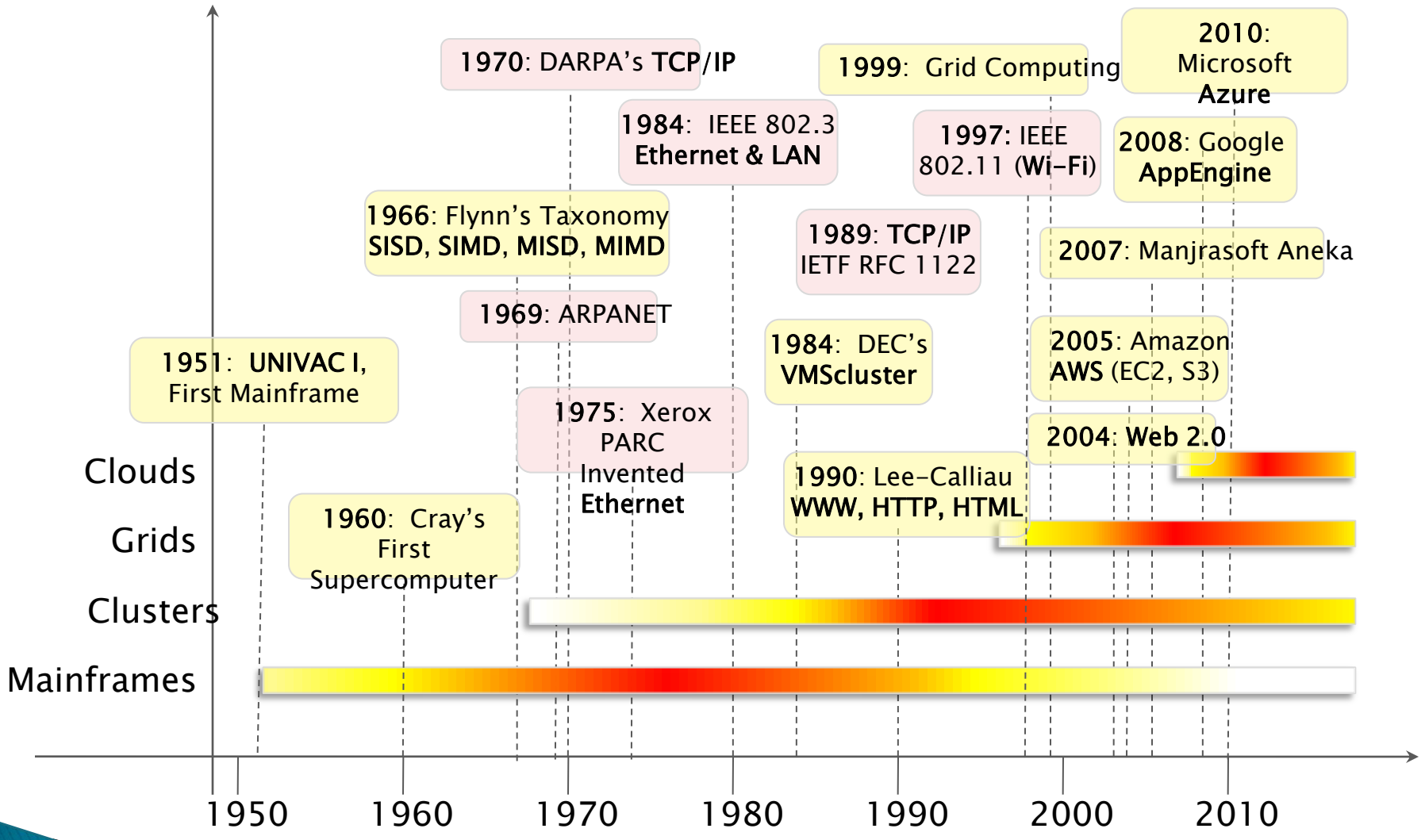
Hybrid/Inter Clouds

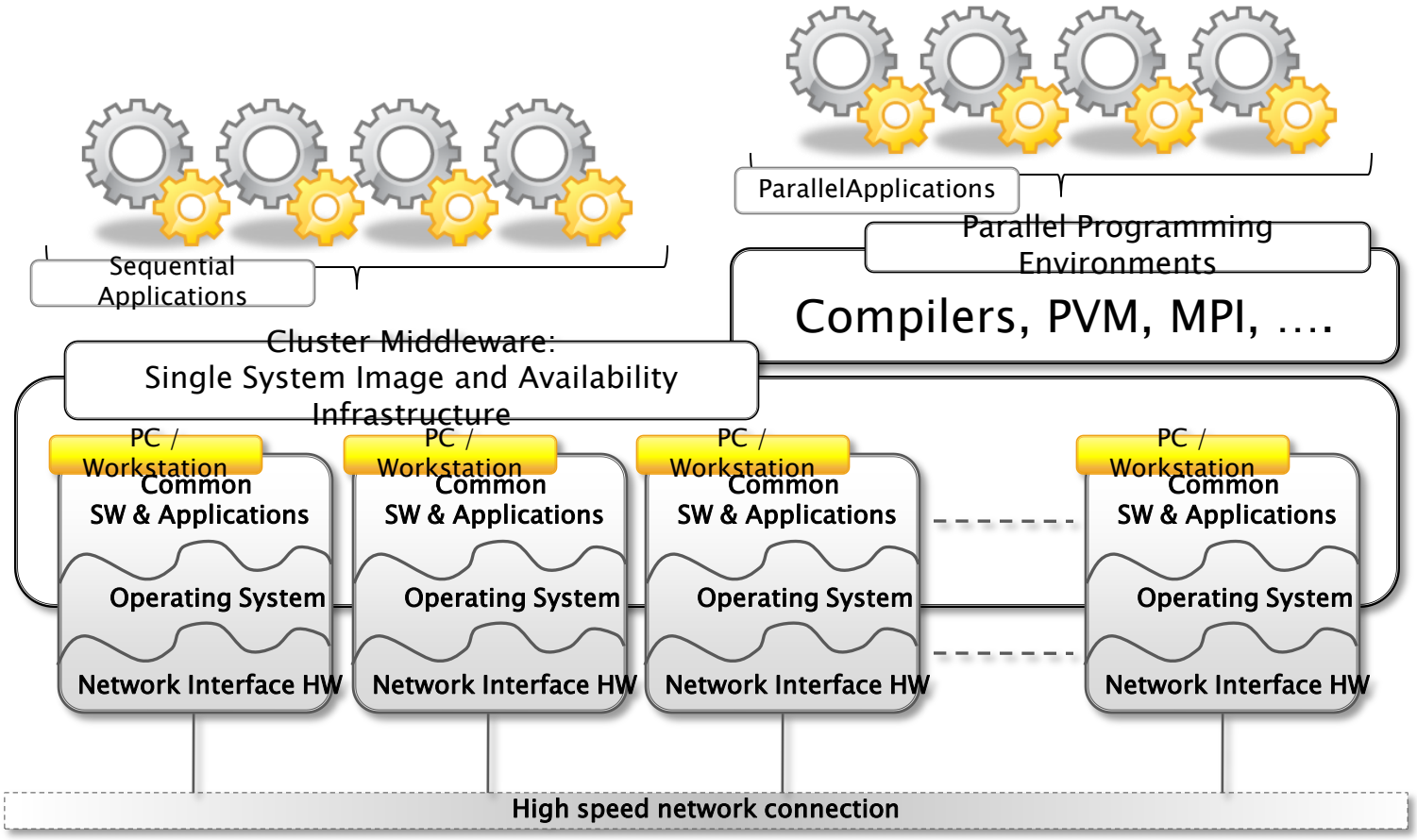
* Mixed usage of private and public Clouds: Leasing public cloud services when private cloud capacity is insufficient

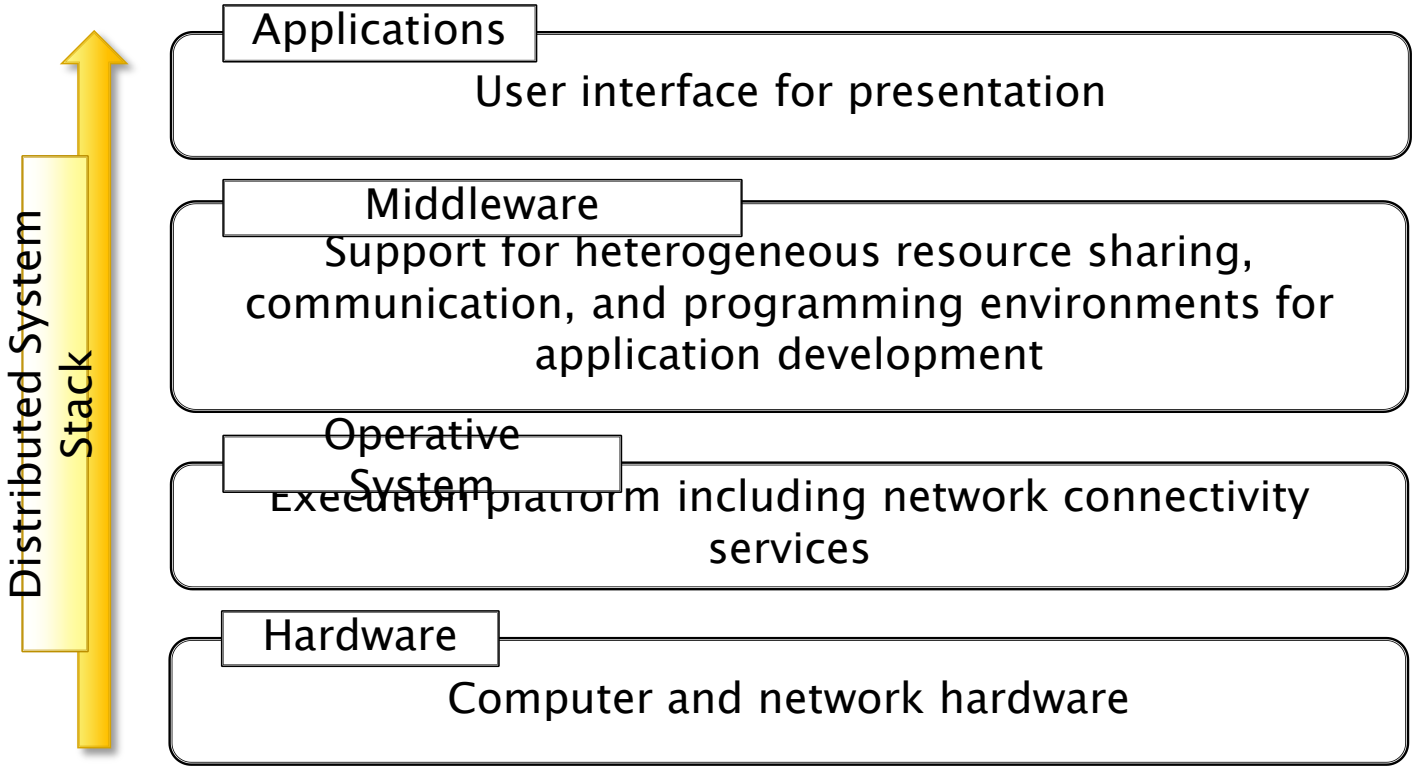


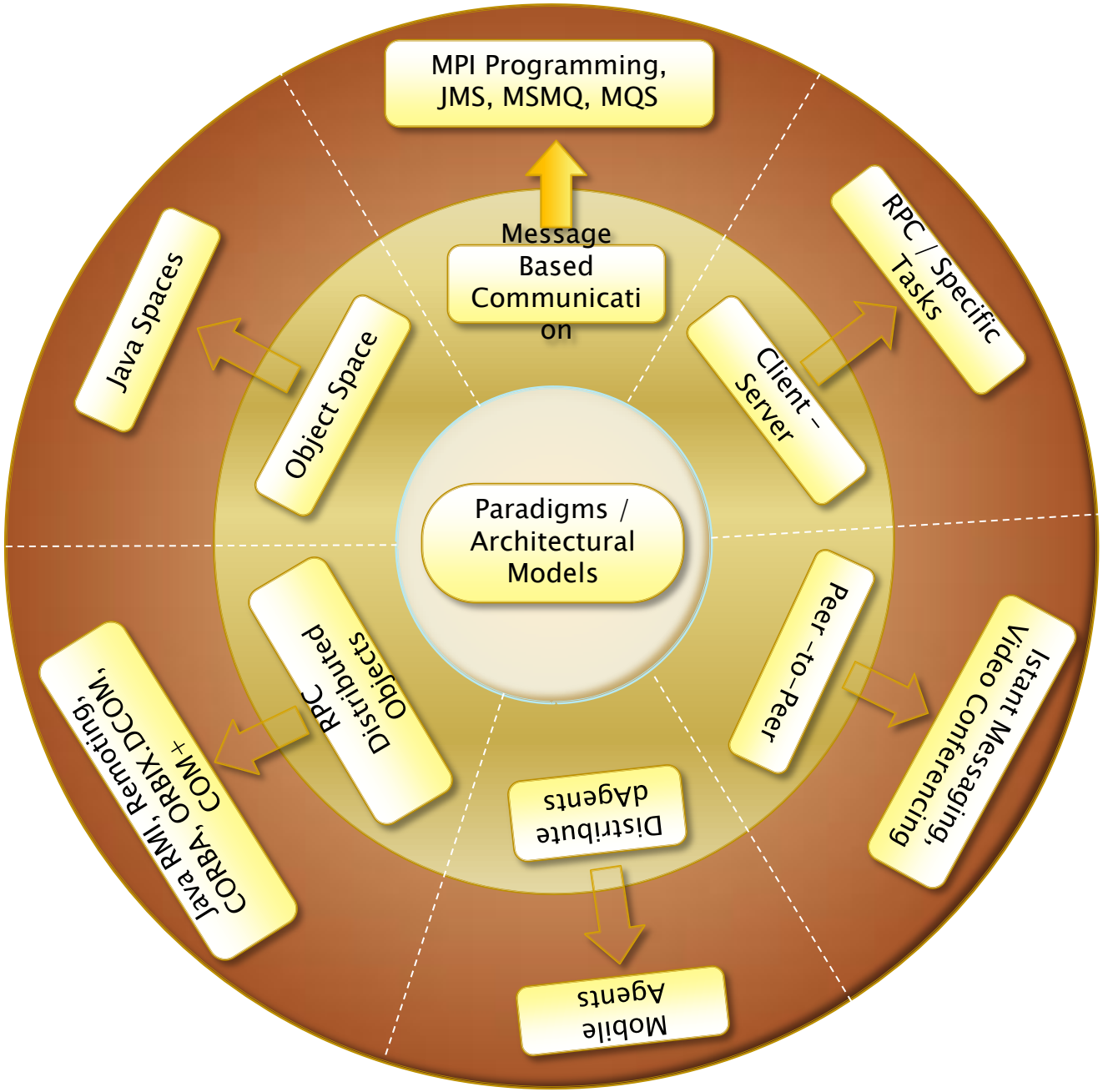

No capital investments
Quality of Service
Pay as you go
Internet
PaaS
Billing
IaaS
Elasticity
IT outsourcing
Service Level Agreement
Utility computing
Green computing
IT outsourcing
Privacy & Trust
Security
Virtualization
Cloud Computing?

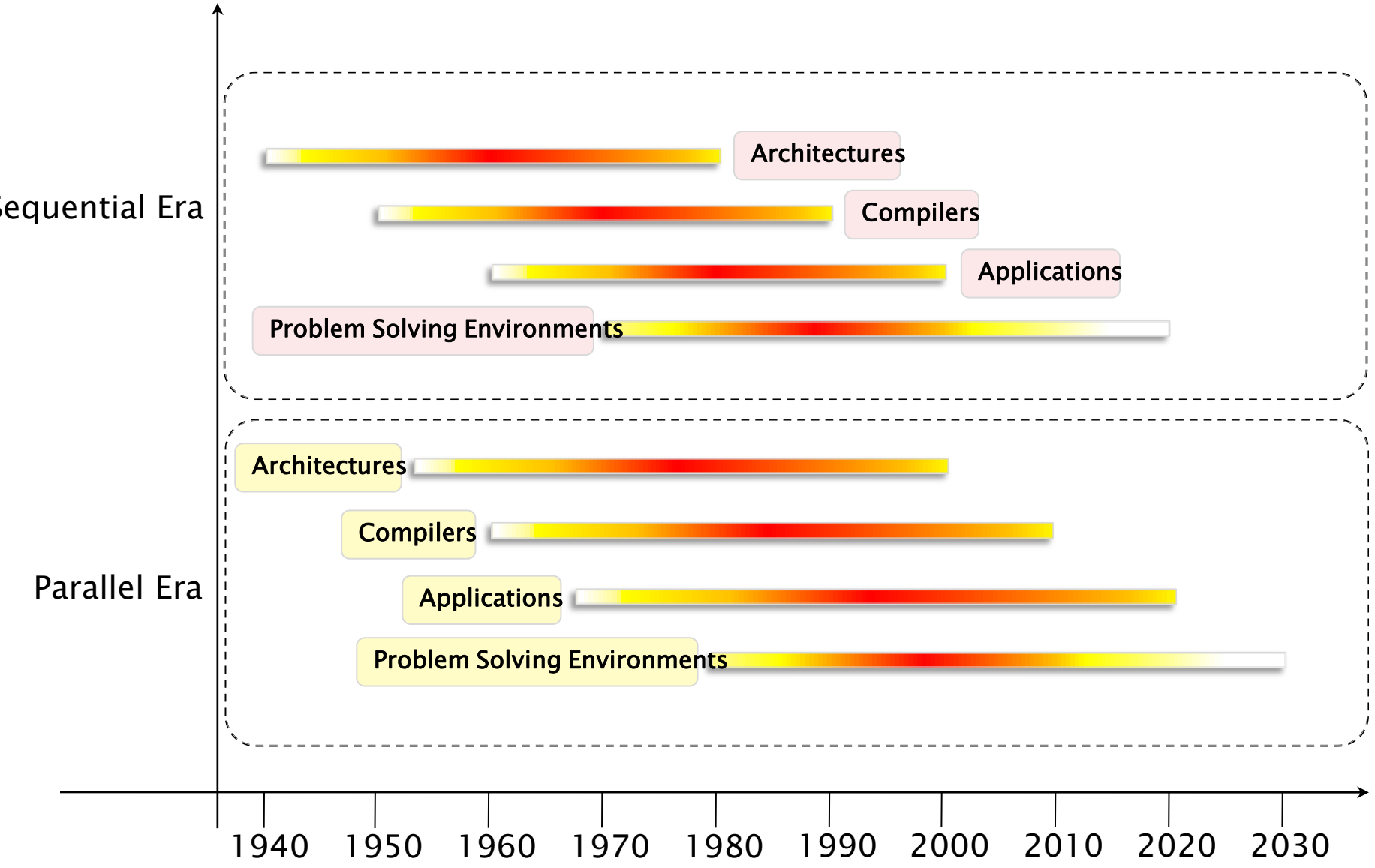

Web 2.0 Interfaces
Software as a Service
End user applications
Scientific applications
Office automation, Photo editing,
CRM, and Social Networking
Google docs
Platform as a Service
Runtime Environment for Applications
Development and Data Processing Platforms
Infrastructure as a Service
Virtualized Servers
Storage and Networking
Examples: Amazon EC2, S3, Rightscale, vCloud

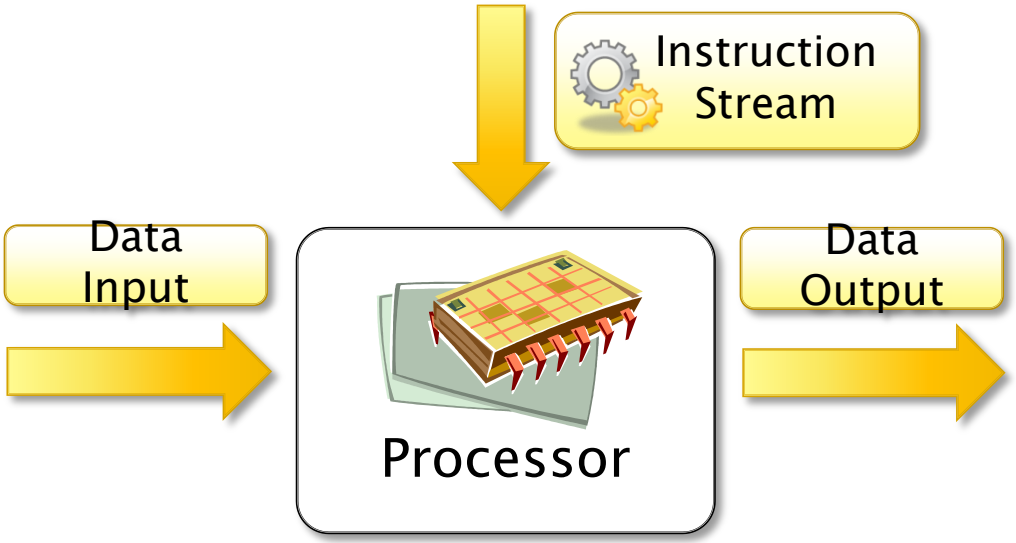

1970: DARPA's TCP/IP
1999: Grid Computing
2010: Microsoft Azure
1984: IEEE 802.3 Ethernet & LAN
1997: IEEE 802.11 (Wi-Fi)
2008: Google AppEngine
1966: Flynn's Taxonomy SISD, SIMD, MISD, MIMD
1989: TCP/IP IETF RFC 1122
2007: Manjrasoft Aneka
1969: ARPANET
1984: DEC's VMScluster
2005: Amazon AWS (EC2, S3)
1951: UNIVAC I, First Mainframe
1975: Xerox PARC Invented Ethernet
2004: Web 2.0
1990: Lee-Calliau WWW, HTTP, HTML
Clouds
Grids
1960: Cray's First Supercomputer
Clusters
Mainframes
1950
1960
1970
1980
1990
2000
2010

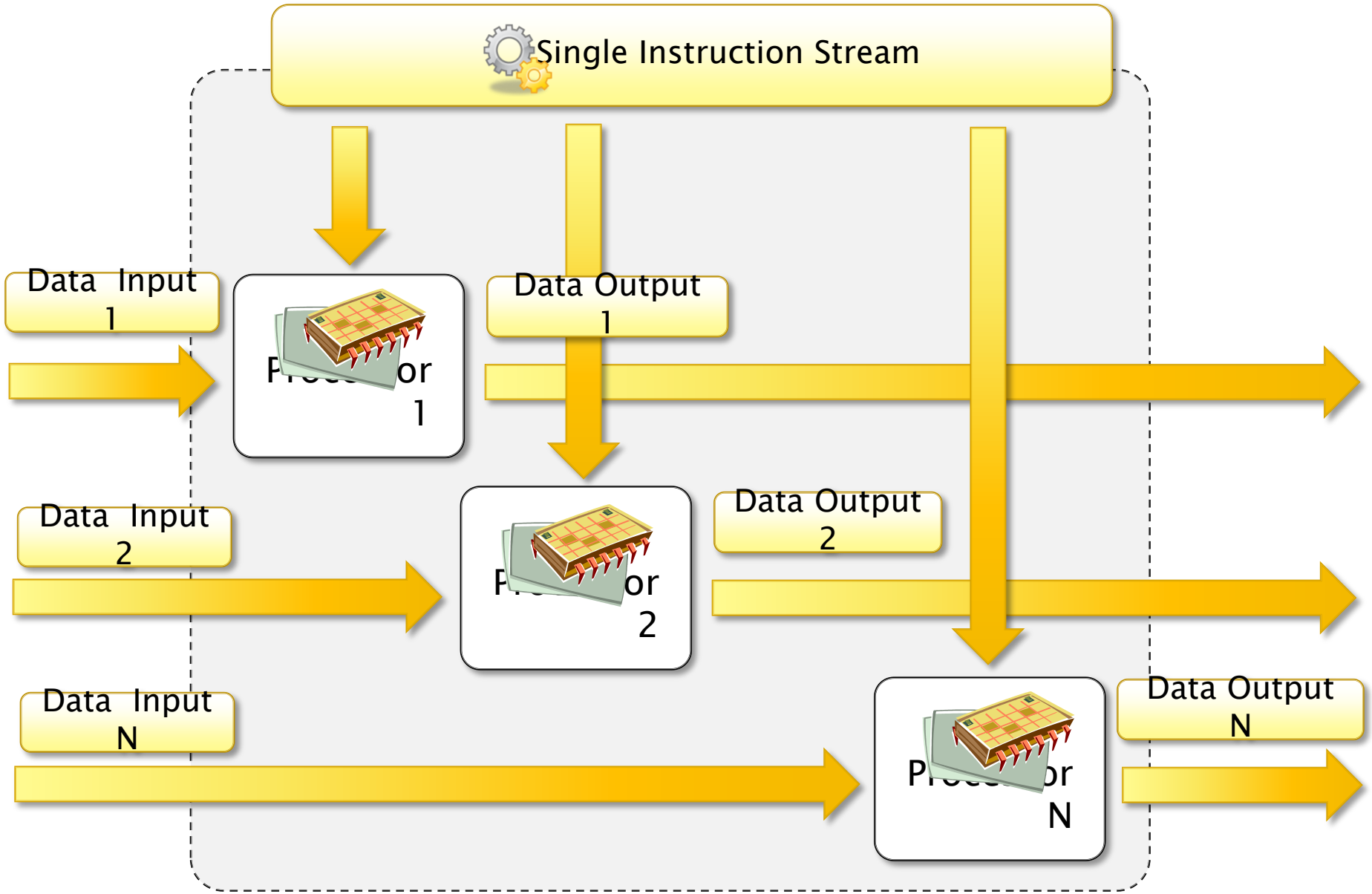

ParallelApplications
Parallel Programming Environments
Compilers, PVM, MPI, ….
Sequential Applications
Cluster Middleware:
Single System Image and Availability Infrastructure
PC / Workstation
Common SW & Applications
Operating System
Network Interface HW
PC / Workstation
Common SW & Applications
Operating System
Network Interface HW
PC / Workstation
Common SW & Applications
Operating System
Network Interface HW
PC / Workstation
Common SW & Applications
Operating System
Network Interface HW
High speed network connection

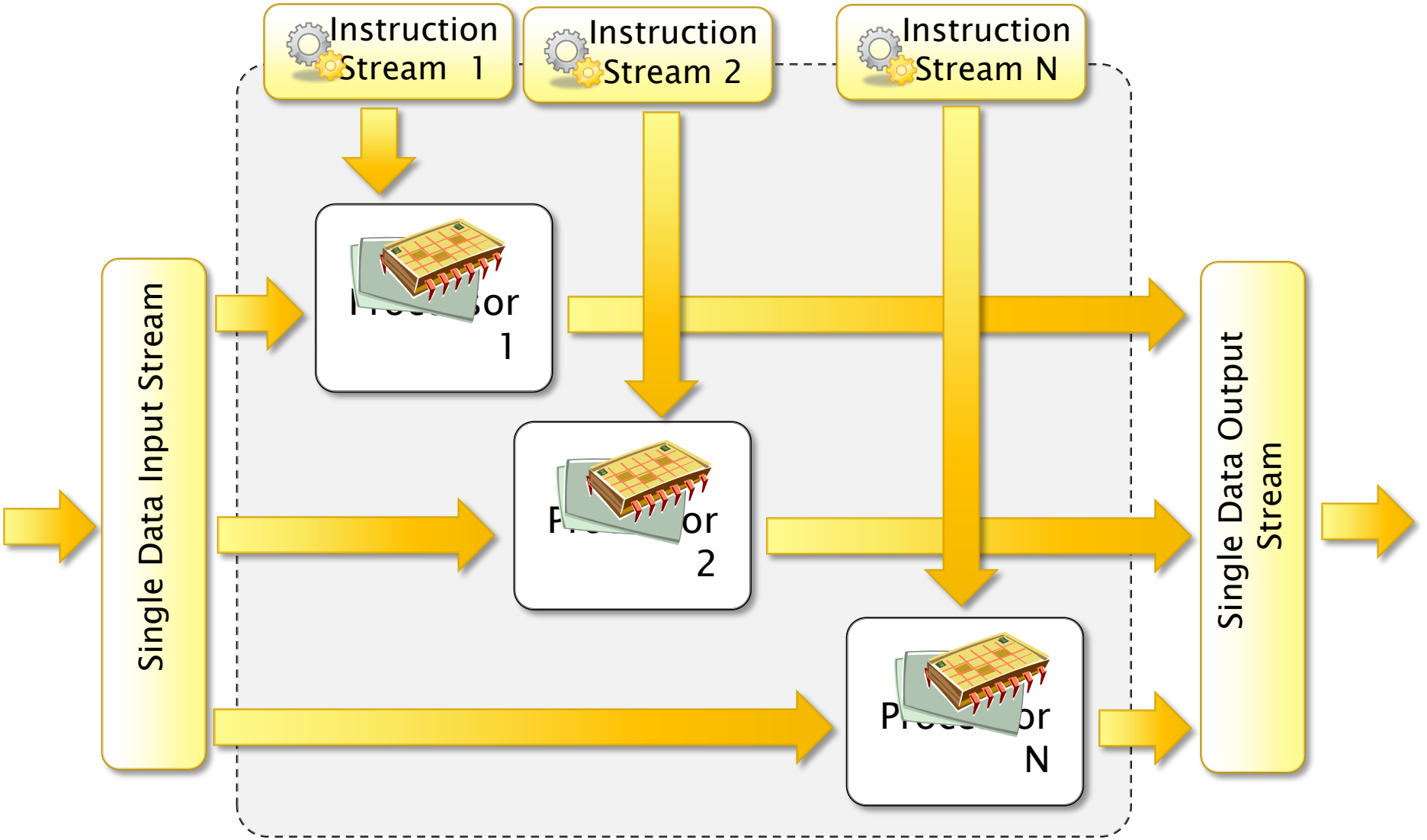

Distributed System Stack
Applications
User interface for presentation
Middleware
Support for heterogeneous resource sharing, communication, and programming environments for application development
Operative System
Execution platform including network connectivity services
Hardware
Computer and network hardware

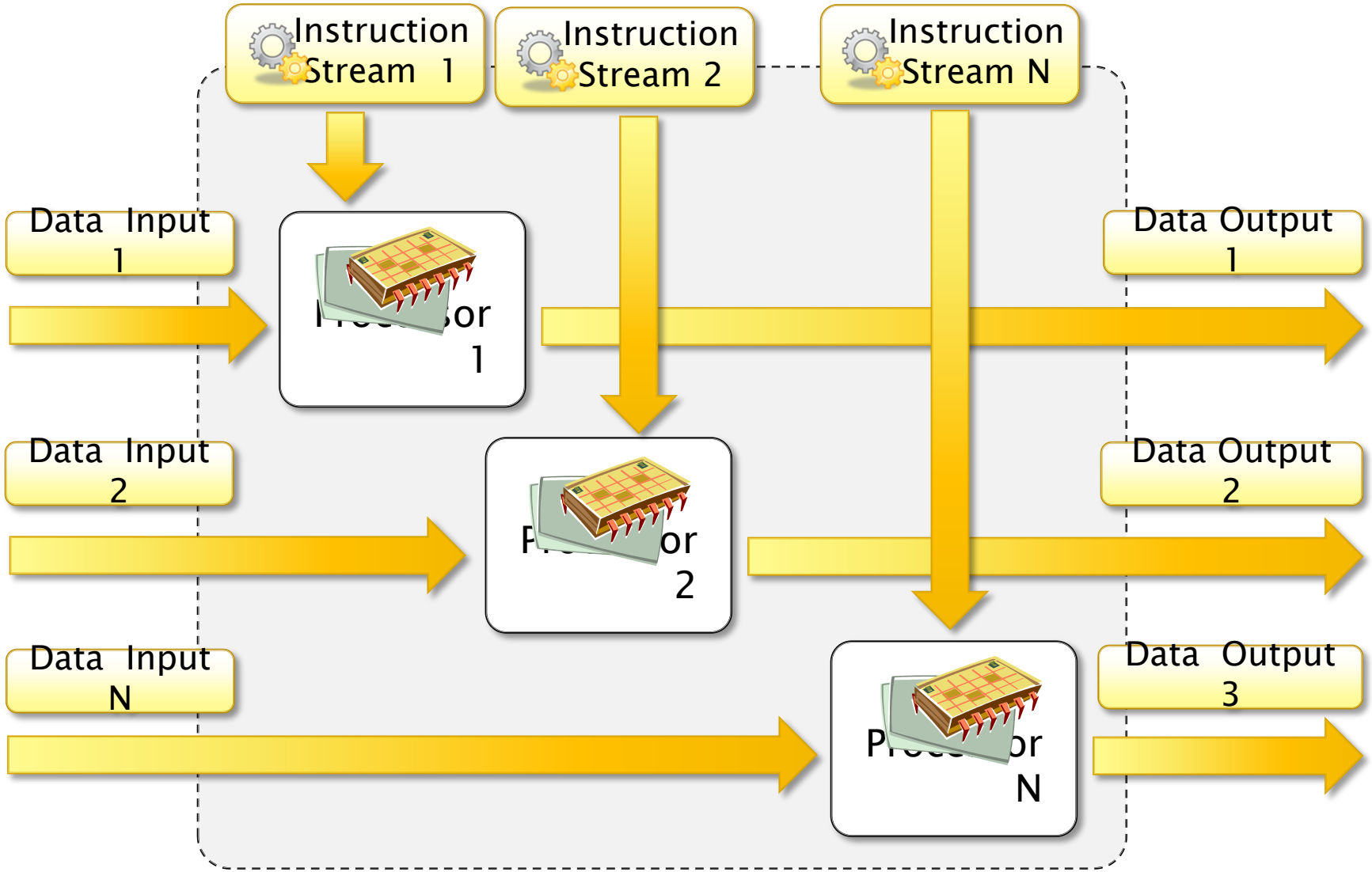

MPI Programming, JMS, MSMQ, MQS
Message Based Communication
Java Spaces
Object Space
RPC / Specific Tasks
Client – Server
Paradigms / Architectural Models
Distributed Objects RPC
Java RMI, Remoting, CORBA, ORBIX.DCOM, COM+
Peer – to-Peer
Istant Messaging, Video Conferencing
Distribute dAgents
Mobile Agents

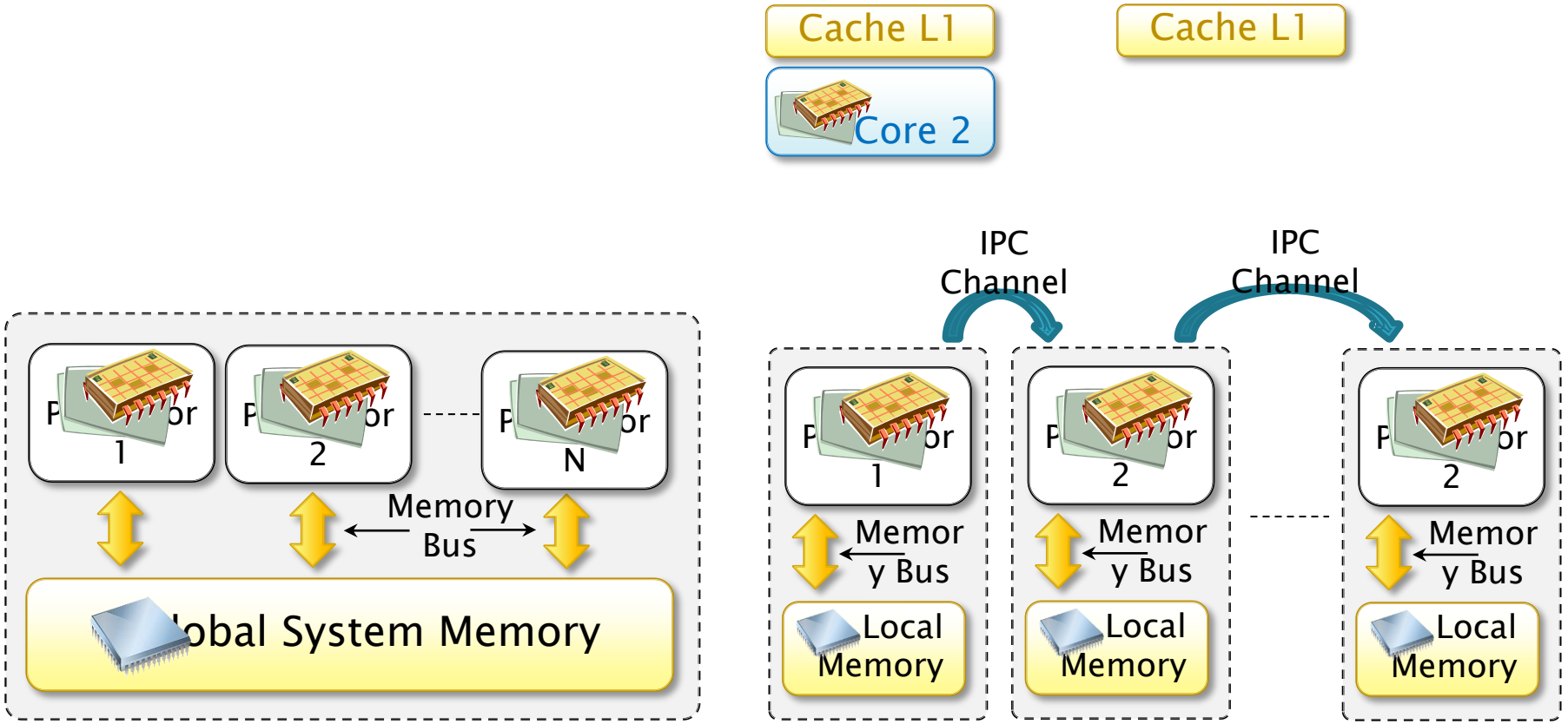

Sequential Era
Architectures
Compilers
Applications
Problem Solving Environments
Parallel Era
Architectures
Compilers
Applications
Problem Solving Environments
1940
1950
1960
1970
1980
1990
2000
2010
2020
2030

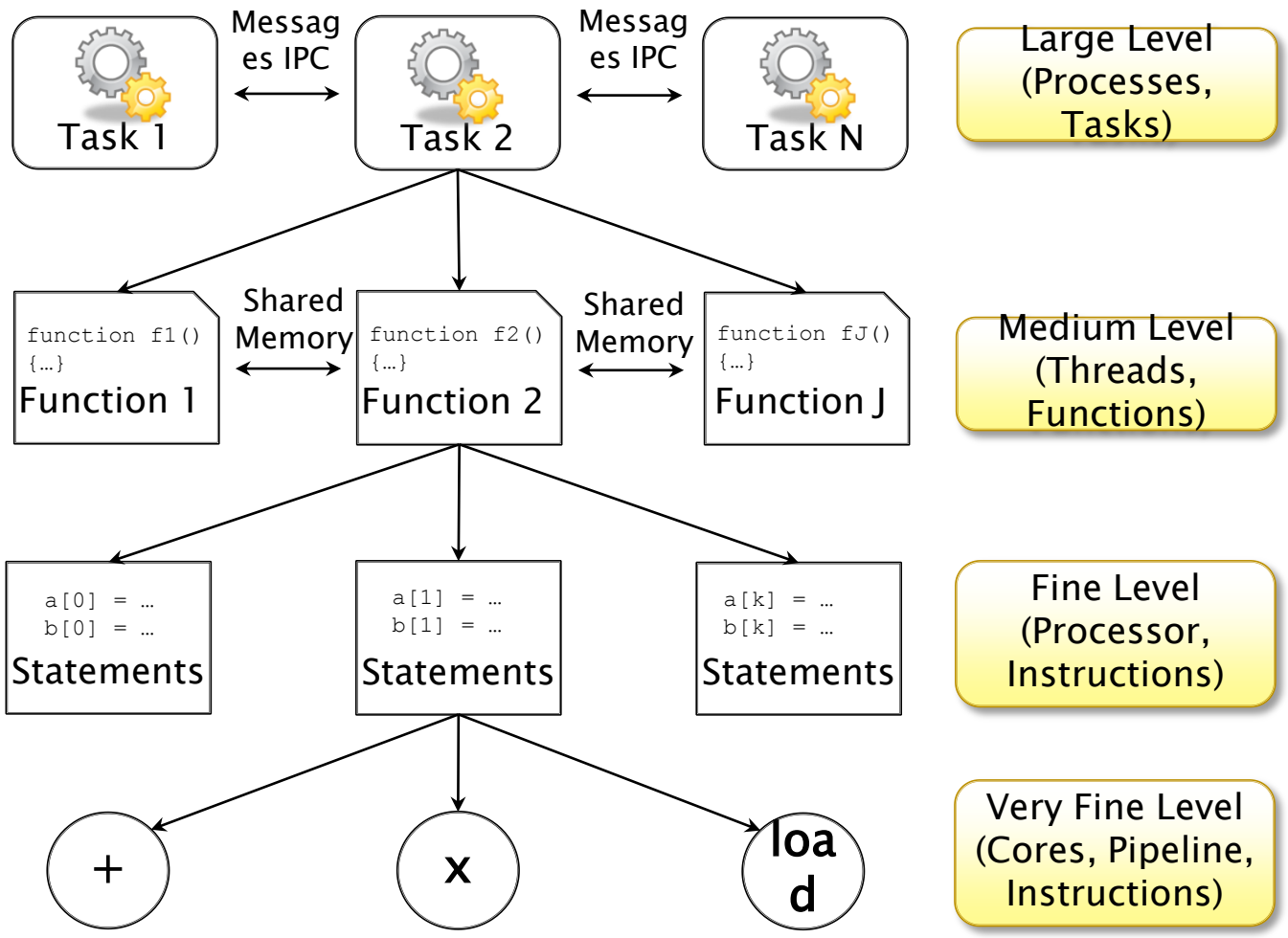

Instruction Stream
Data Input
Data Output
Processor

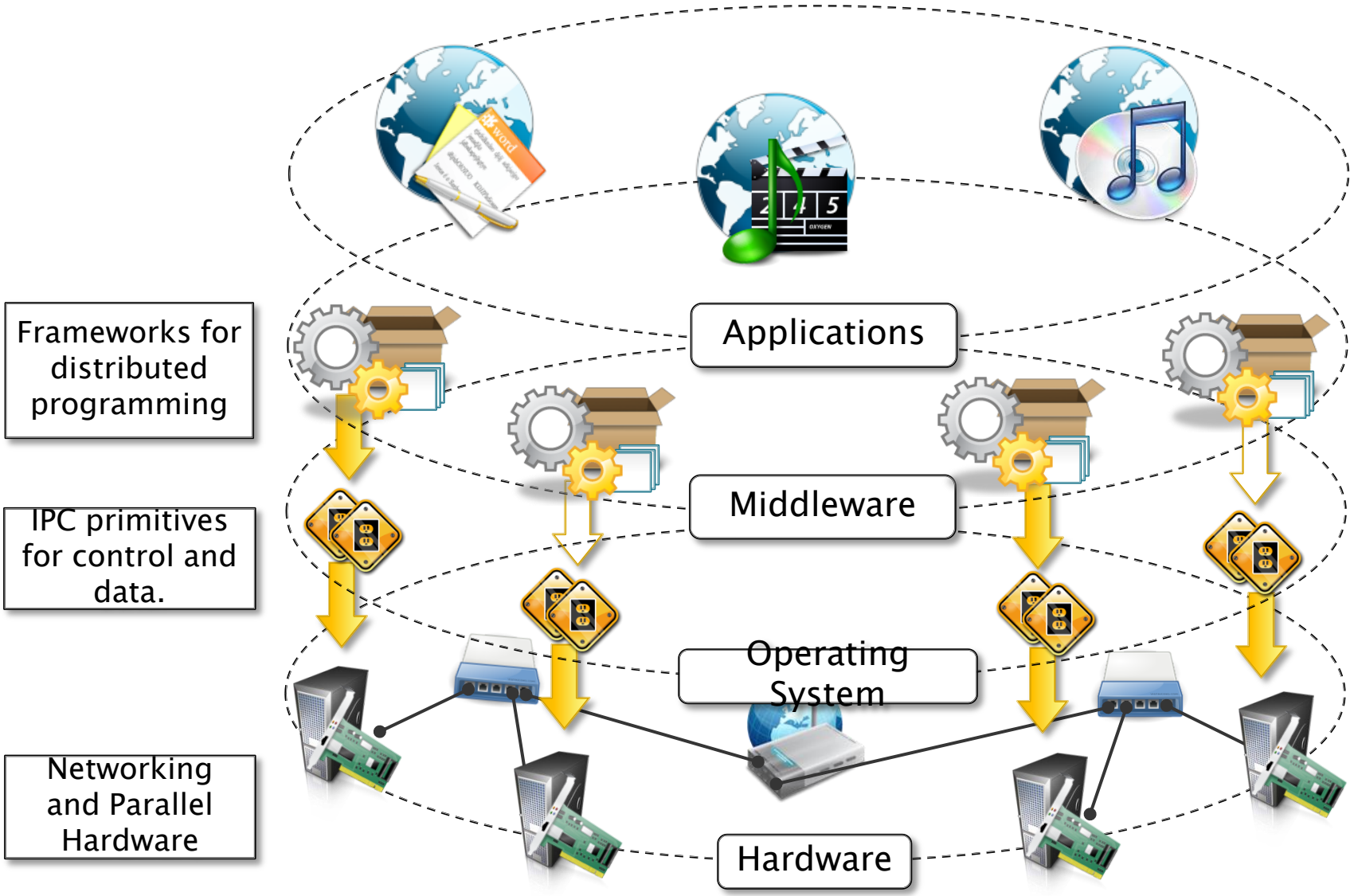

Single Instruction Stream
Data Input 1
Data Output 1
Data Input 2
Data Output 2
Data Input N
Data Output N

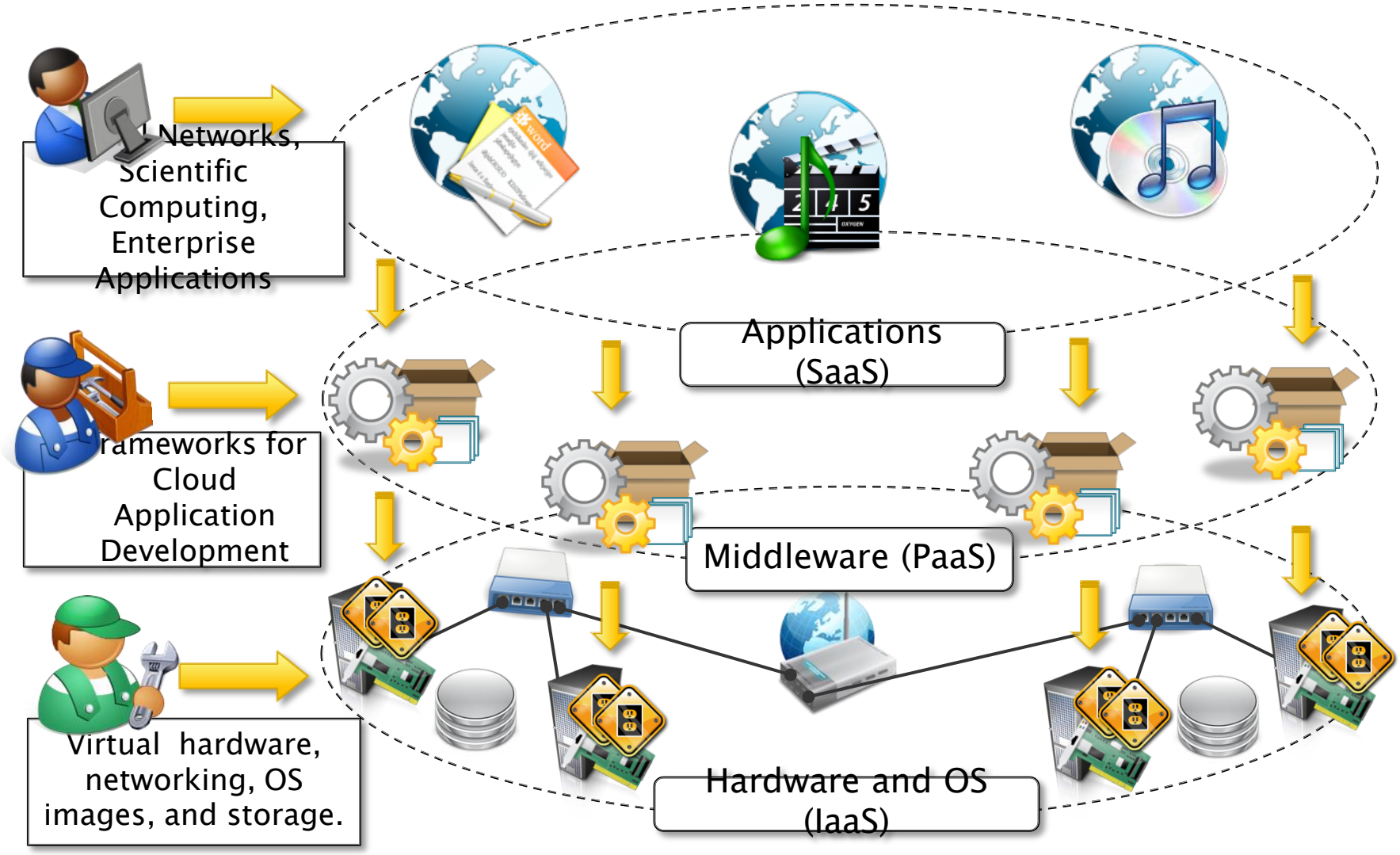

Instruction Stream 1
Instruction Stream 2
Instruction Stream N
Single Data Input Stream
Single Data Output Stream


Instruction Stream 1
Instruction Stream 2
Instruction Stream N
Data Input 1
Data Input 2
Data Input N
Data Output 1
Data Output 2
Data Output 3


Cache L1
Cache L1
Core 2
IPC
Channel
IPC
Channel
1
2
N
Memory
Bus
obal System Memory
1
2
2
Memor
y Bus
Memor
y Bus
Memor
y Bus
Local
Memory
Local
Memory
Local
Memory

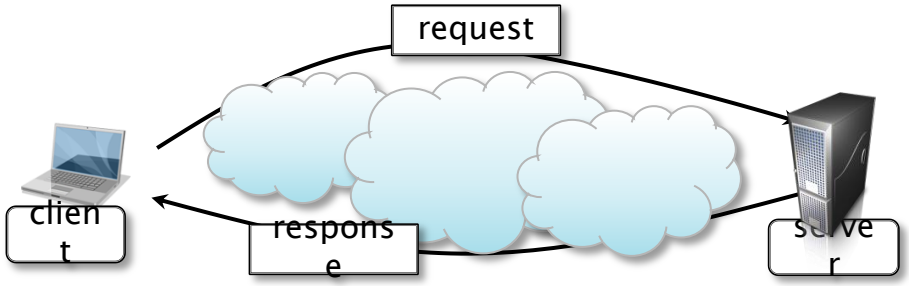

Task 1
Messag es IPC
Task 2
Messag es IPC
Task N
Large Level (Processes, Tasks)
function f1() {…}
Function 1
Shared Memory
function f2() {…}
Function 2
Shared Memory
function fJ() {…}
Function J
Medium Level (Threads, Functions)
a[0] = …
b[0] = …
Statements
a[1] = …
b[1] = …
Statements
a[k] = …
b[k] = …
Statements
Fine Level (Processor, Instructions)
+
x
loa d
Very Fine Level (Cores, Pipeline, Instructions)


Frameworks for distributed programming
IPC primitives for control and data.
Networking and Parallel Hardware
Applications
Middleware
Operating System
Hardware

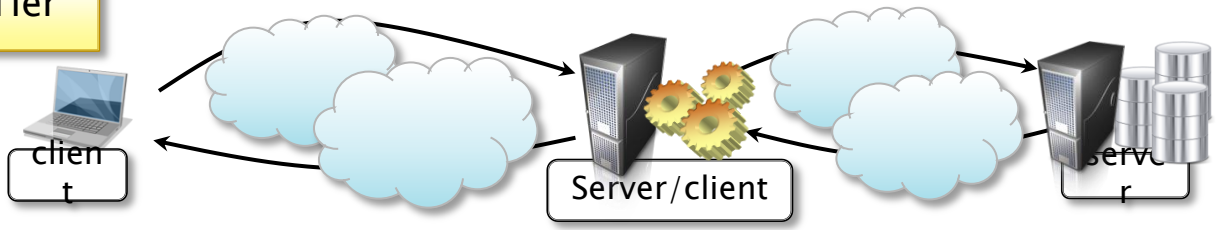

Networks, Scientific Computing, Enterprise Applications
Applications (SaaS)
rameworks for Cloud Application Development
Middleware (PaaS)
Virtual hardware, networking, OS images, and storage.
Hardware and OS (IaaS)

Client-server

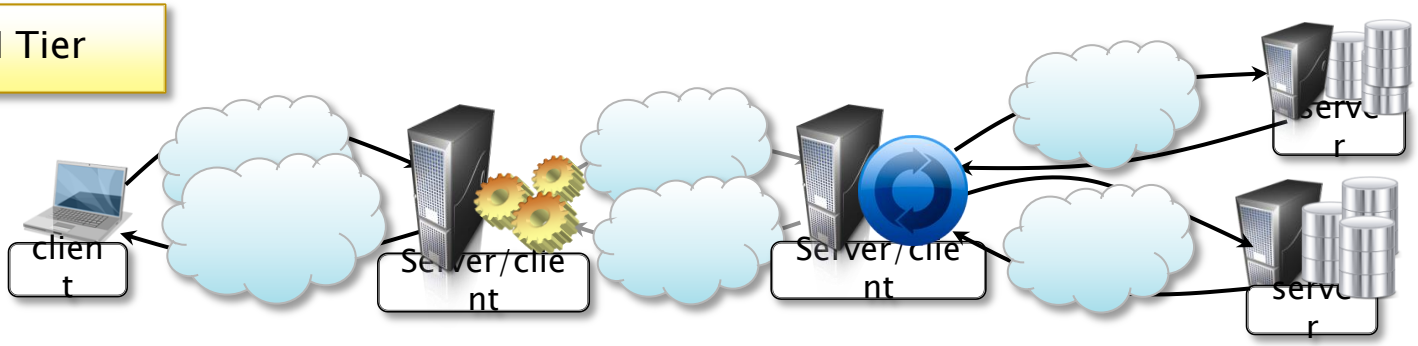
Two Tier (Classic Model)



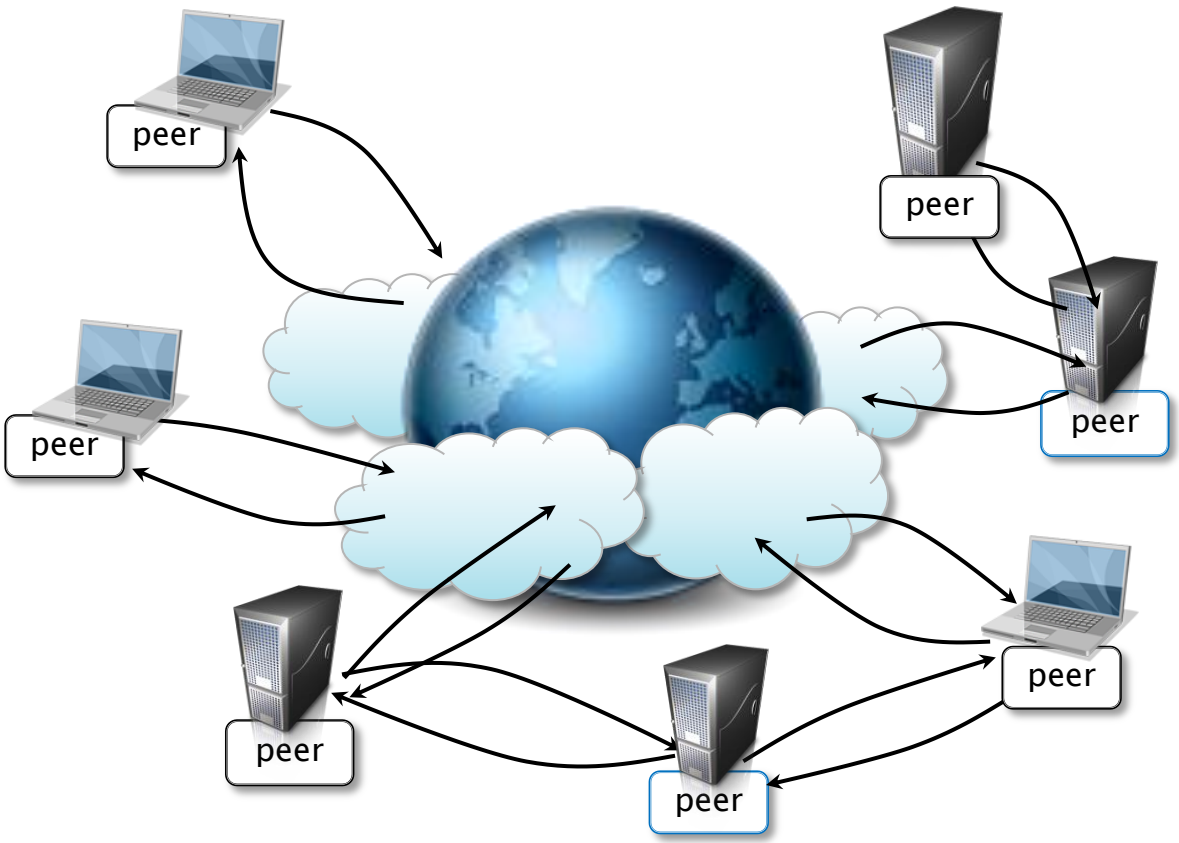
Three Tier



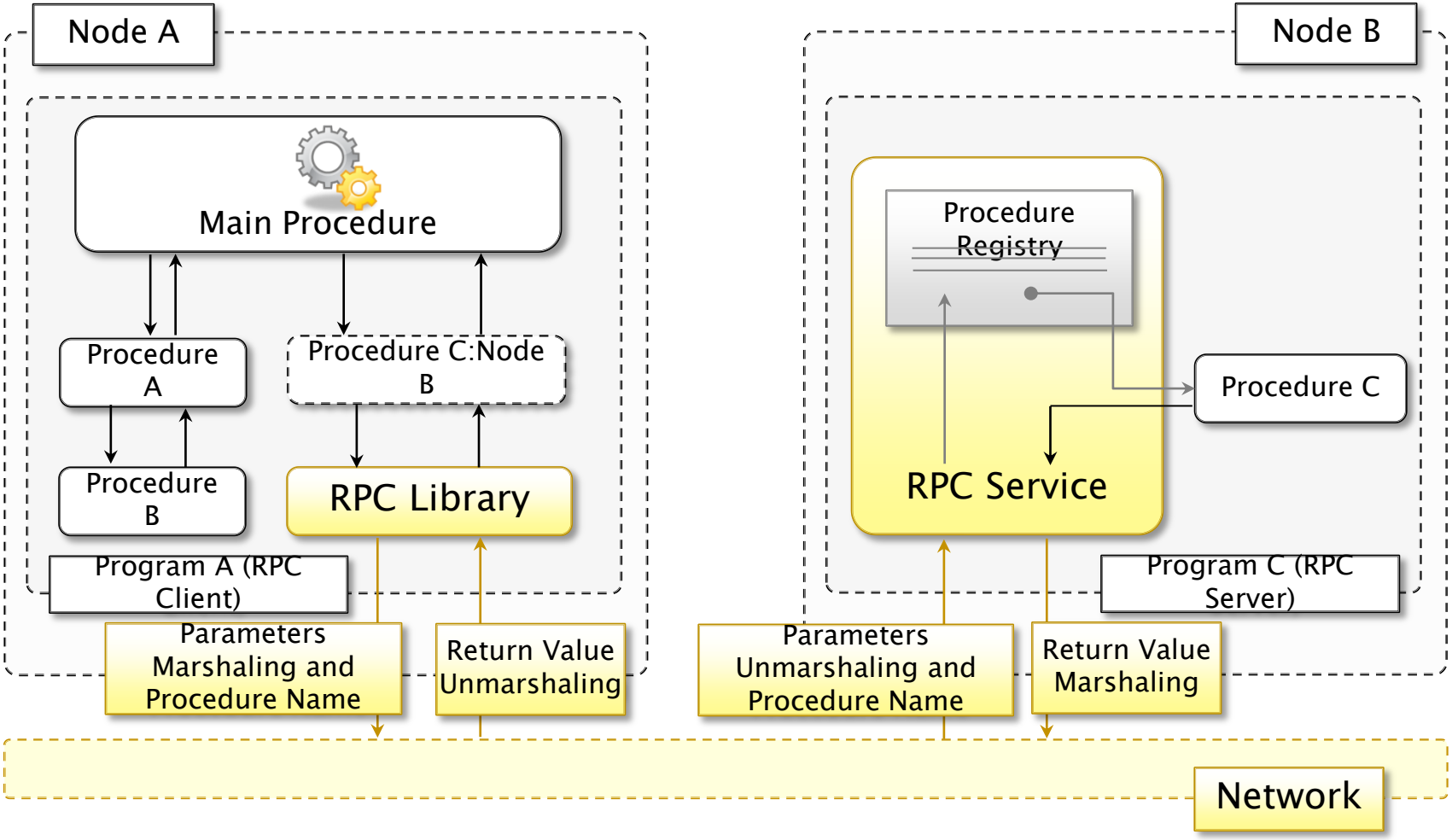
N Tier



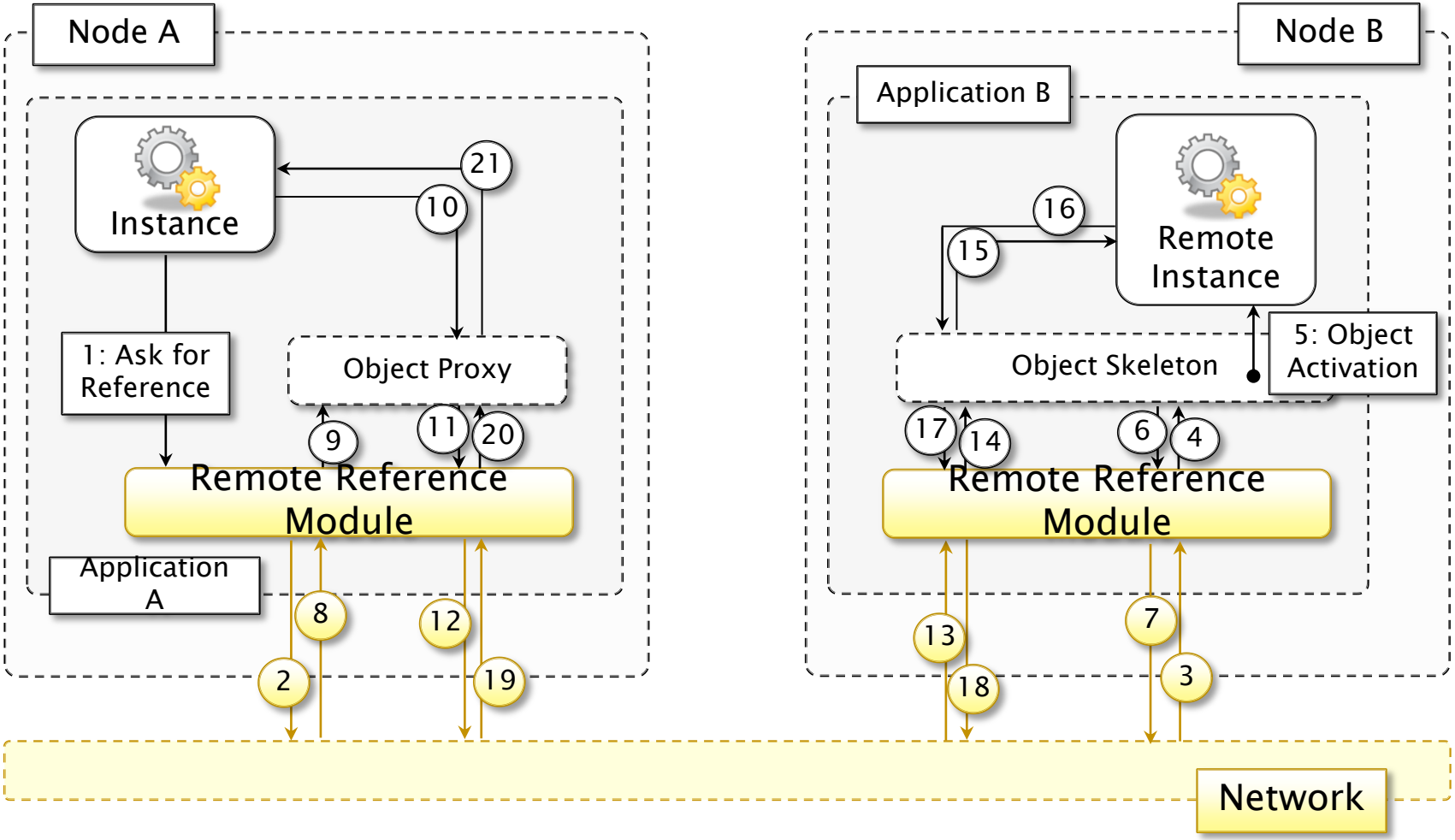
P2P

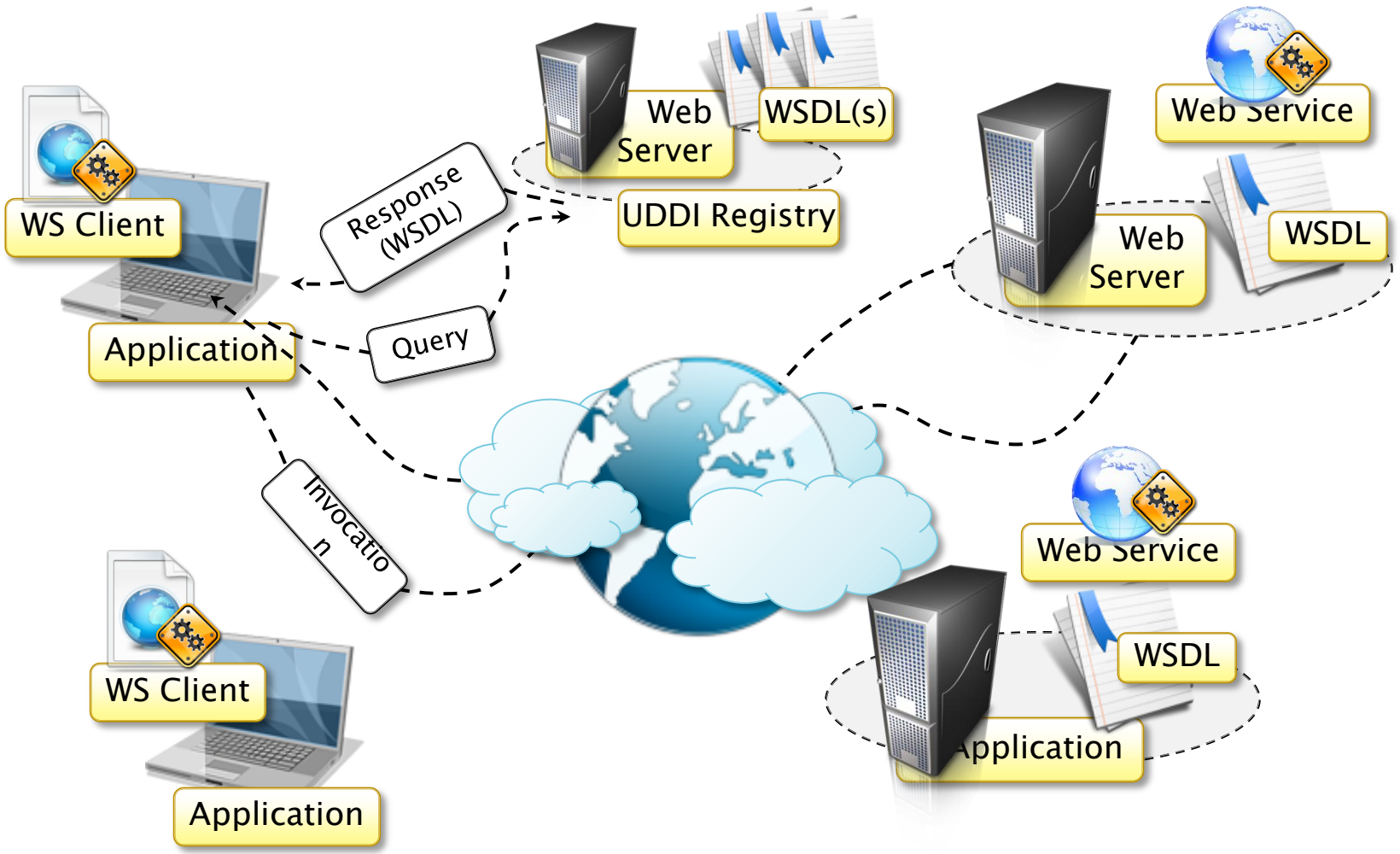


RPC

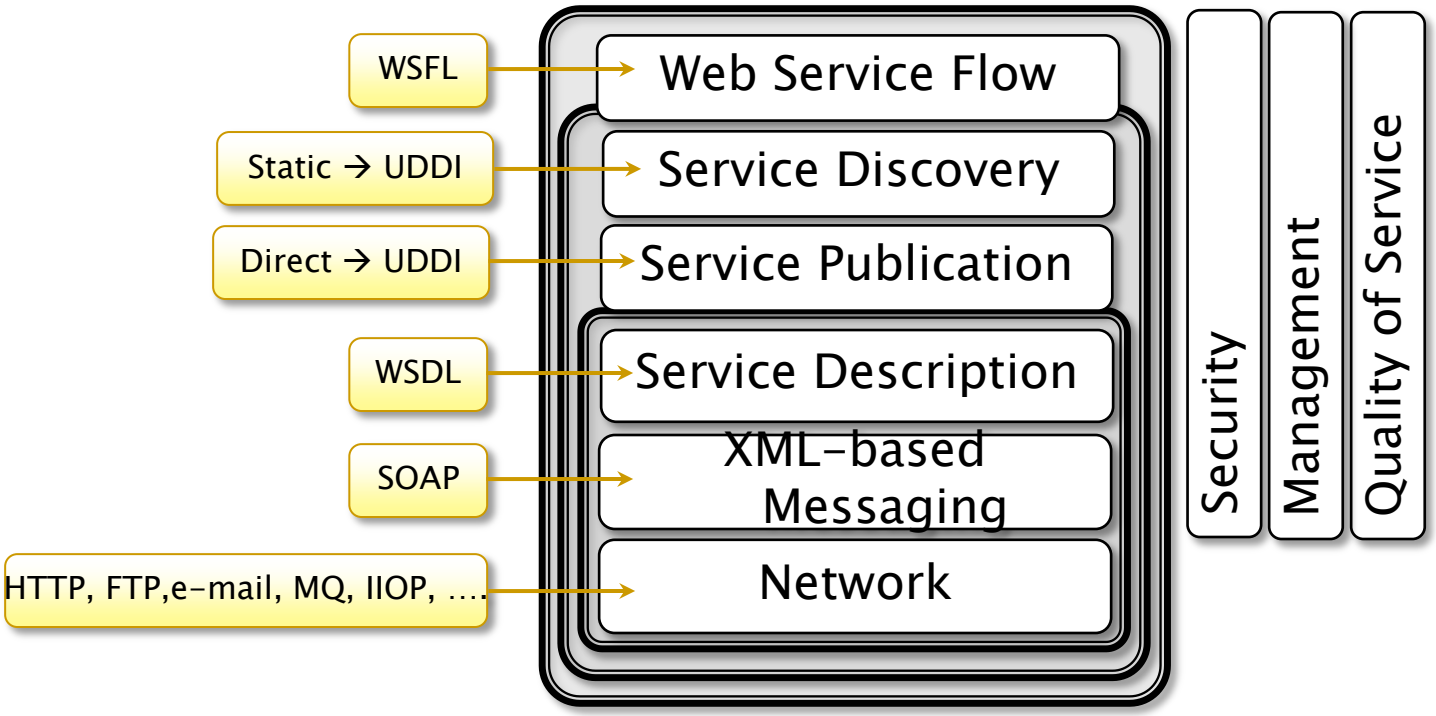

Node A
Main Procedure
Procedure A
Procedure C:Node B
Procedure B
RPC Library
Program A (RPC Client)
Parameters Marshaling and Procedure Name
Return Value Unmarshaling
Node B
Procedure Registry
Procedure C
RPC Service
Program C (RPC Server)
Parameters Unmarshaling and Procedure Name
Return Value Marshaling
Network

DOF interaction




WS Client
Application
Response (WSDL)
Query
Web Server
WSDL(s)
UDDI Registry
Web Service
Web Server
WSDL
Invocation
WS Client
Application
Web Service
WSDL
Application

WS technology Stack



SOAP Messages

```
POST /InStock HTTP/1.1
Host: www.stocks.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: <Size>

<?xml version="1.0">
<soap:Envelope xmlns:soap="http//www.w3.org/2001/12/soap-envelope"
 soap:encondingStyle="http//www.w3.org/2001/12/soap-encoding">
 <soap:Header></soap:Header>
 <soap:Body xmlns:m=http://www.stocks.org/stock>
  <m:GetStockPrice>
   <m:StockName>IBM<m:StockName>
  </m:GetStockPrice>
 </soap:Body>
</soap:Envelope>
```

Envelope

Header: Metadata & Assertions

Body: Method Call

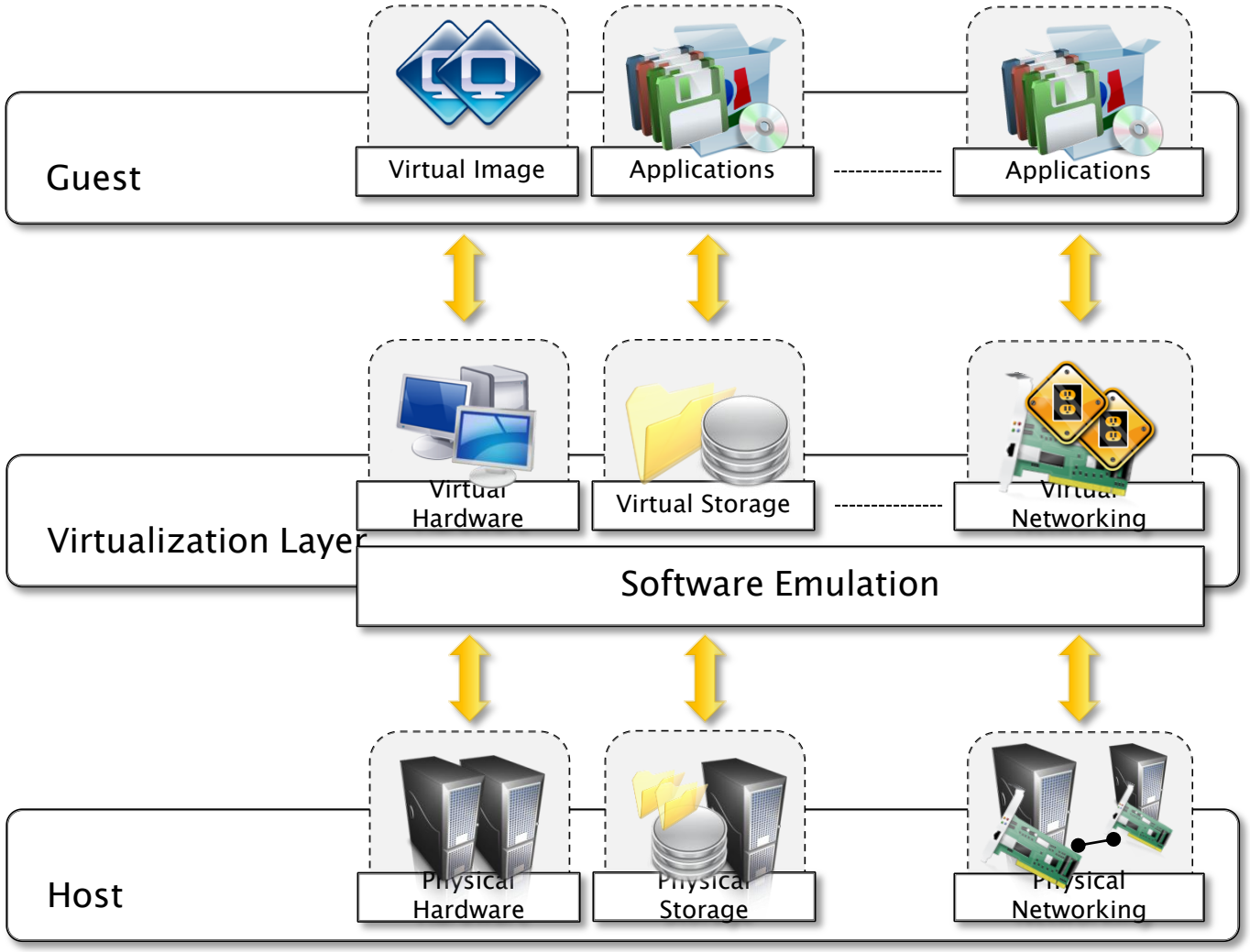
SOAP Messages

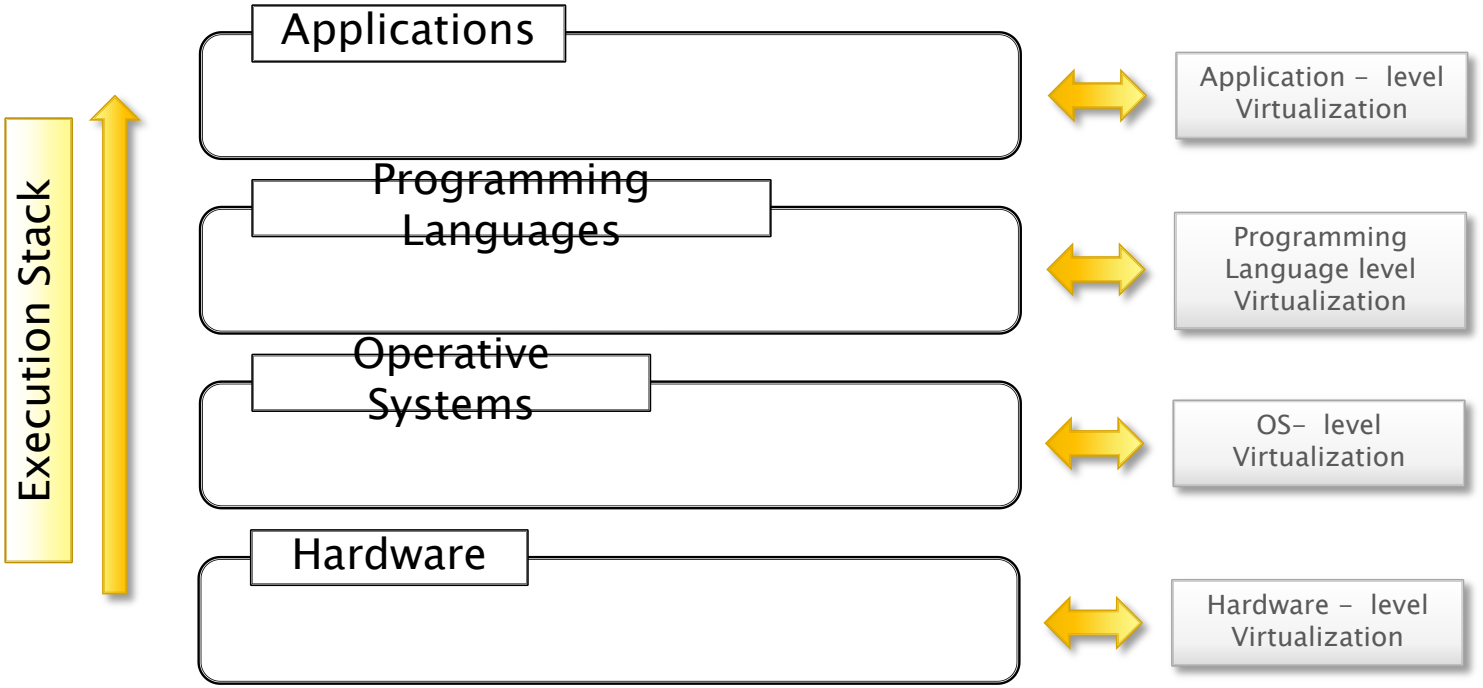
```
POST /InStock HTTP/1.1
Host: www.stocks.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: <Size>

<?xml version="1.0">
<soap:Envelope xmlns:soap="http//www.w3.org/2001/12/soap-envelope"
 soap:encondingStyle="http//www.w3.org/2001/12/soap-encoding">
 <soap:Header></soap:Header>

</soap:Envelope>
```

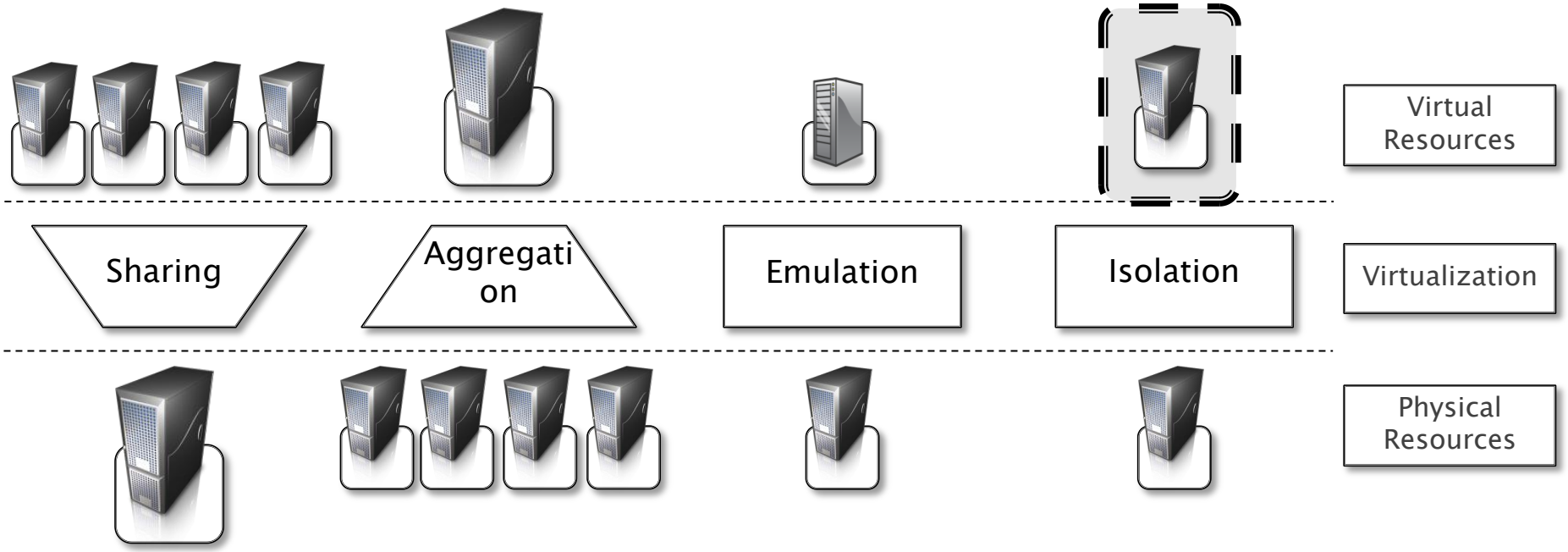
Envelope

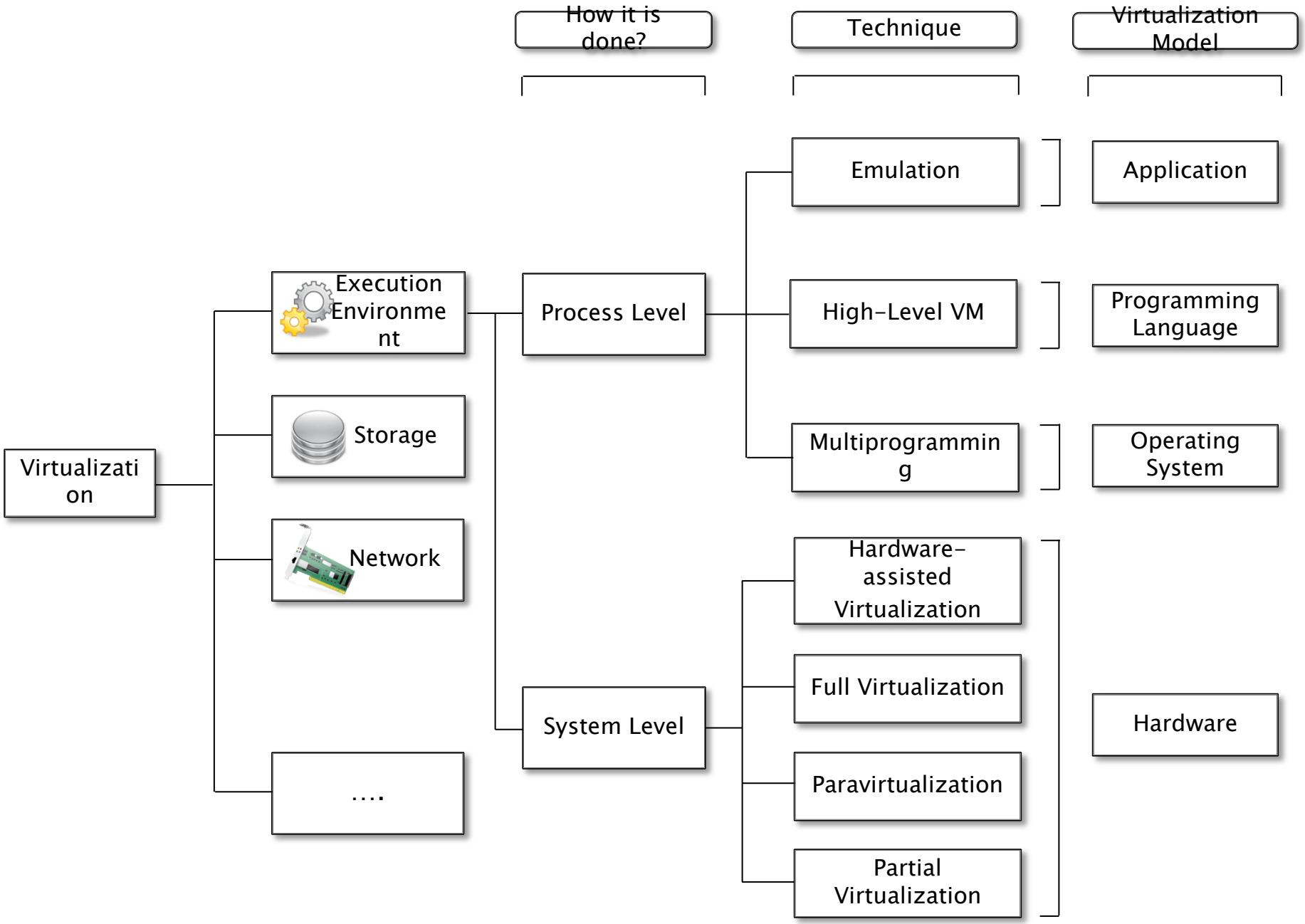
Header: Metadata & Assertions

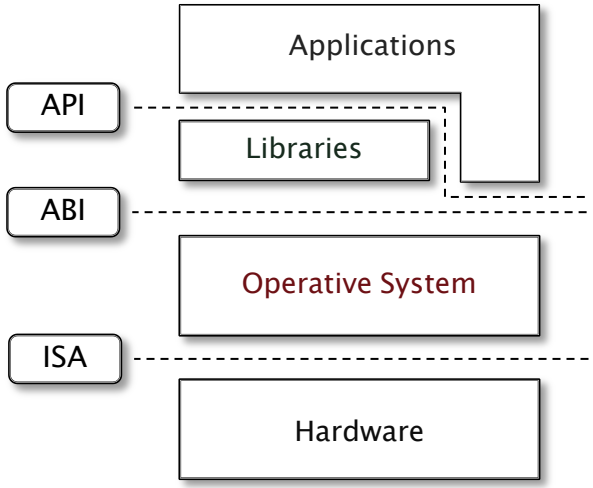
Body: Execution Result

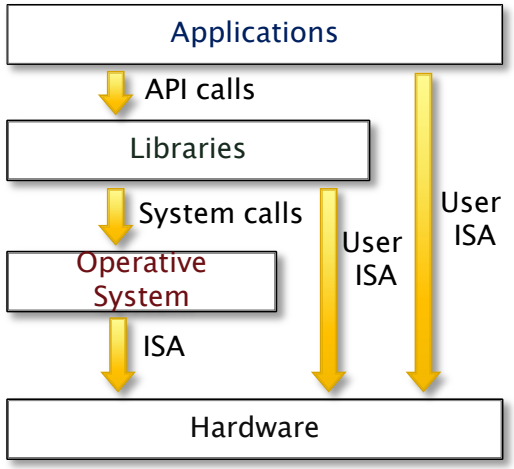

Guest
Virtual Image
Applications
Applications
Virtualization Layer
Virtual Hardware
Virtual Storage
Virtual Networking
Software Emulation
Host
Physical Hardware
Physical Storage
Physical Networking

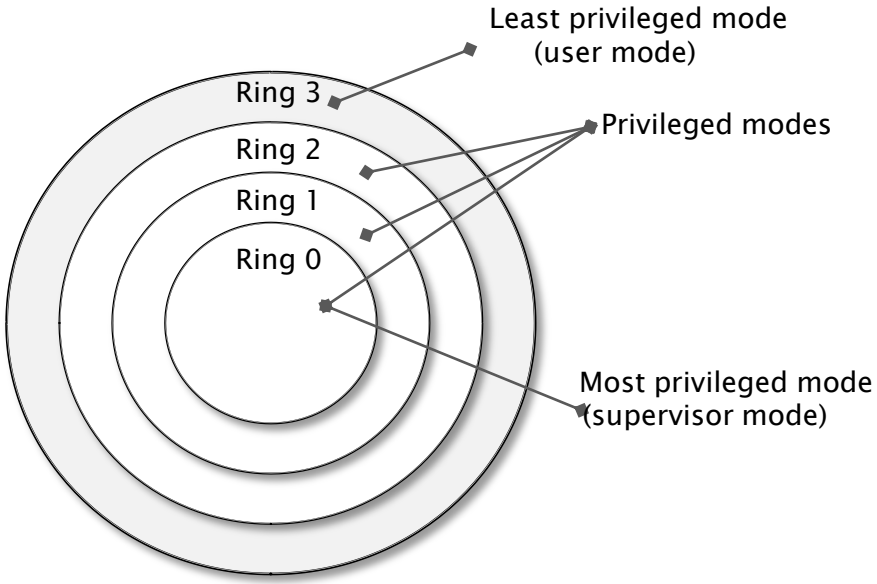

Execution Stack
Applications
Programming Languages
Operative Systems
Hardware
Application - level Virtualization
Programming Language level Virtualization
OS- level Virtualization
Hardware - level Virtualization

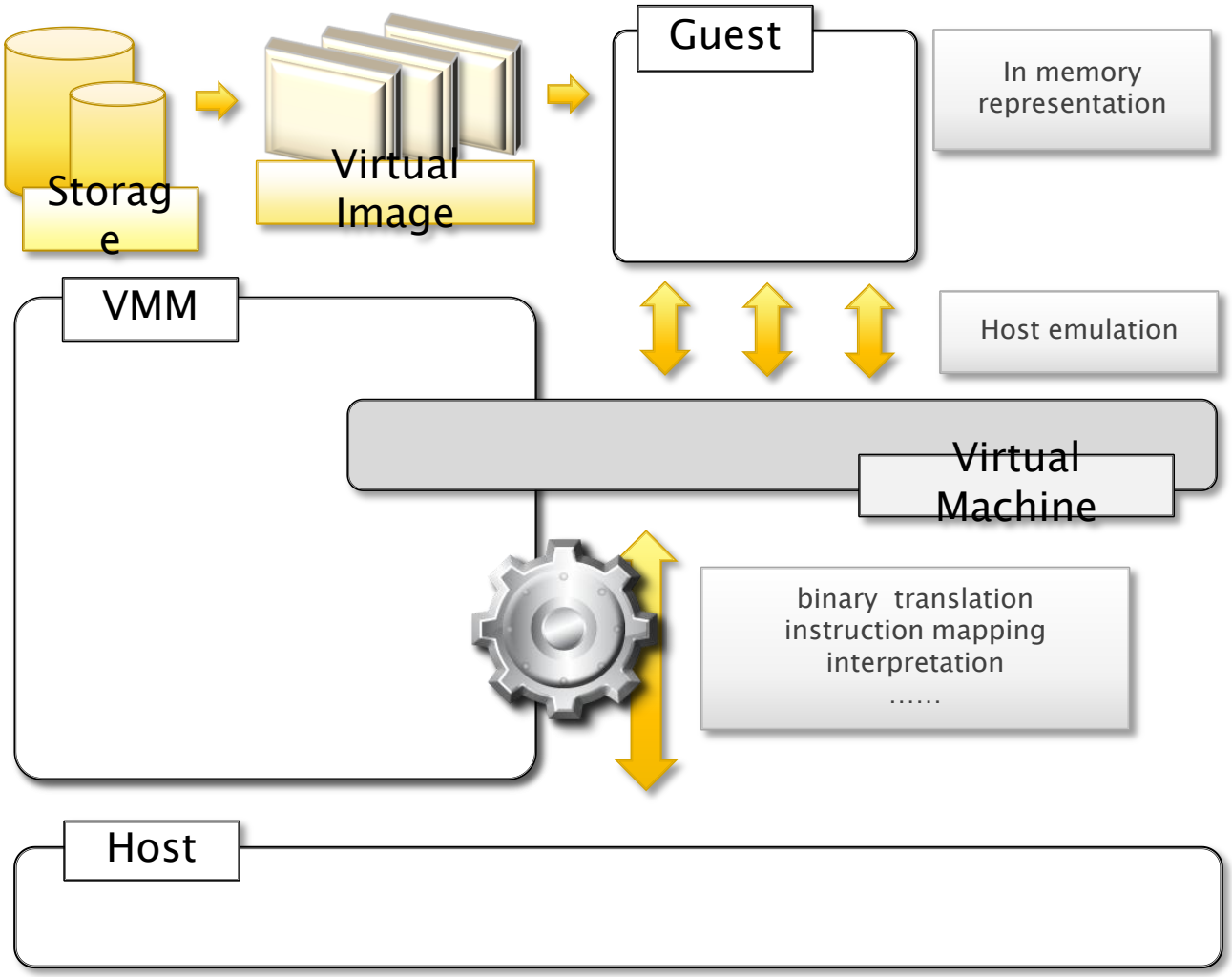

Virtual Resources
Sharing
Aggregation
Emulation
Isolation
Virtualization
Physical Resources

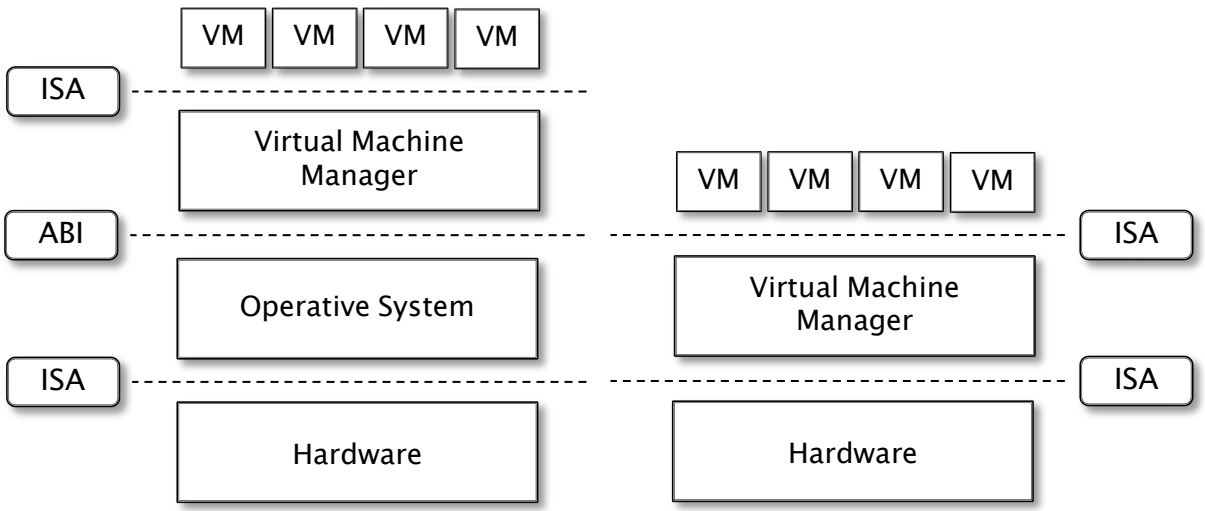

How it is done?
Technique
Virtualization Model
Virtualization
Execution Environment
Storage
Network
....
Process Level
System Level
Emulation
High-Level VM
Multiprogramming
Hardware-assisted Virtualization
Full Virtualization
Paravirtualization
Partial Virtualization
Application
Programming Language
Operating System
Hardware

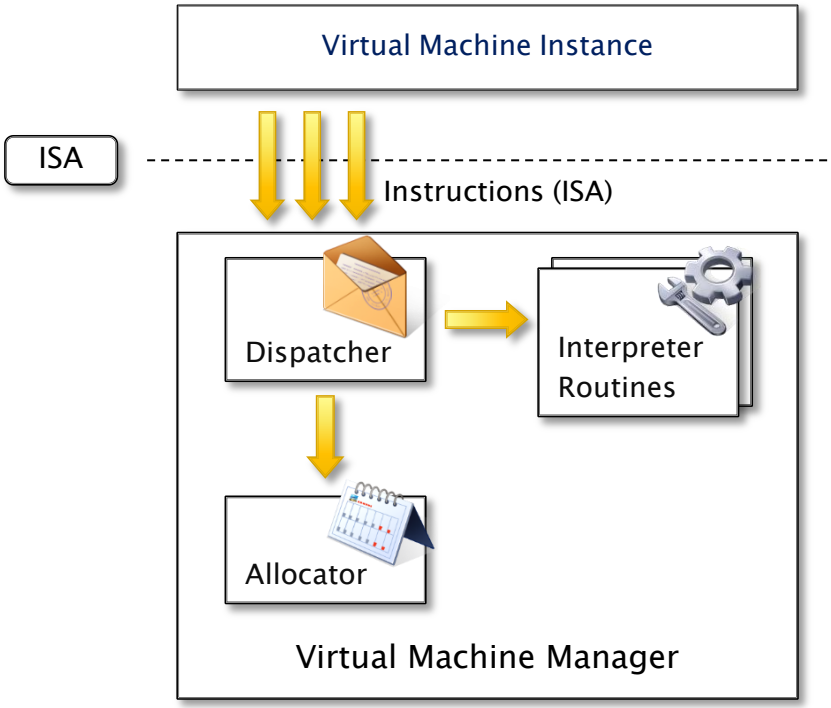

Applications
API
Libraries
ABI
Operative System
ISA
Hardware

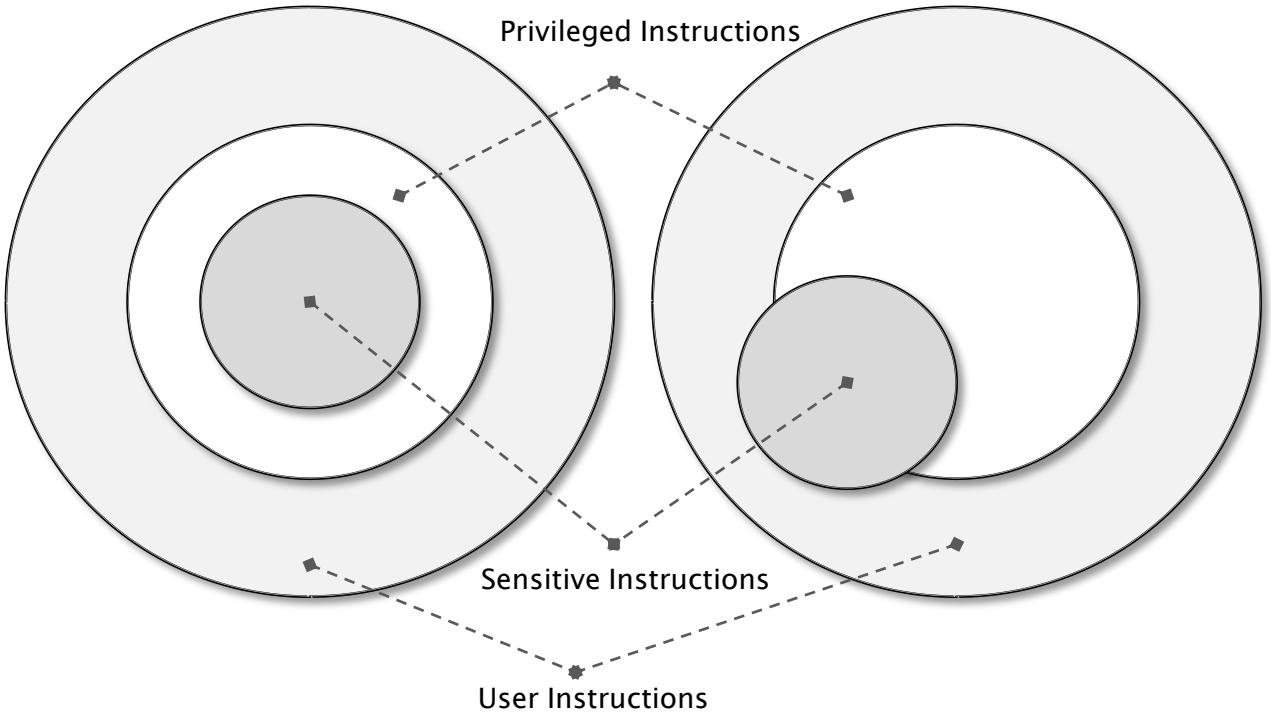

Applications
API calls
Libraries
System calls
Operative System
ISA
Hardware
User ISA
User ISA

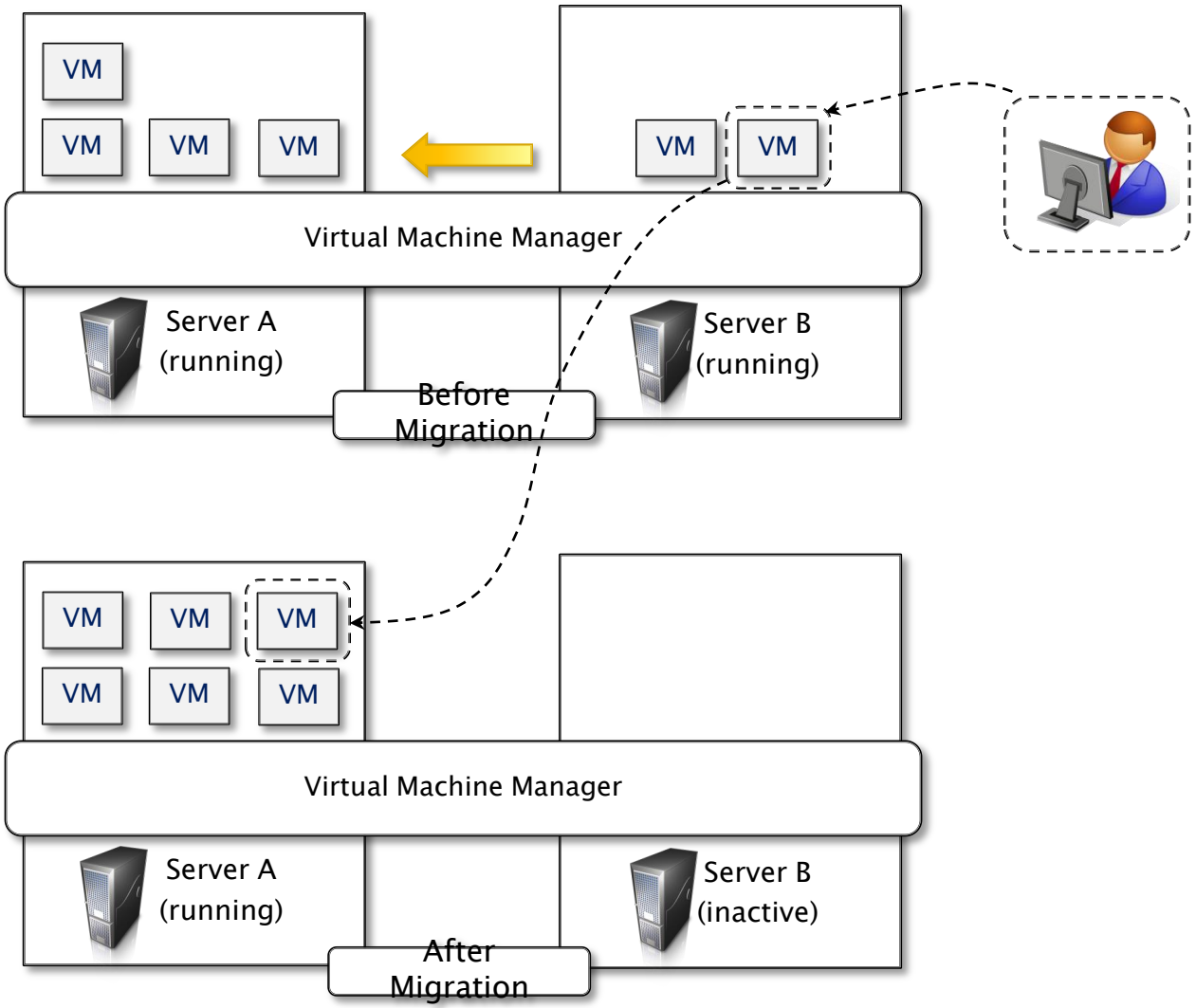

Least privileged mode
(user mode)
Ring 3
Ring 2
Ring 1
Ring 0
Privileged modes
Most privileged mode
(supervisor mode)

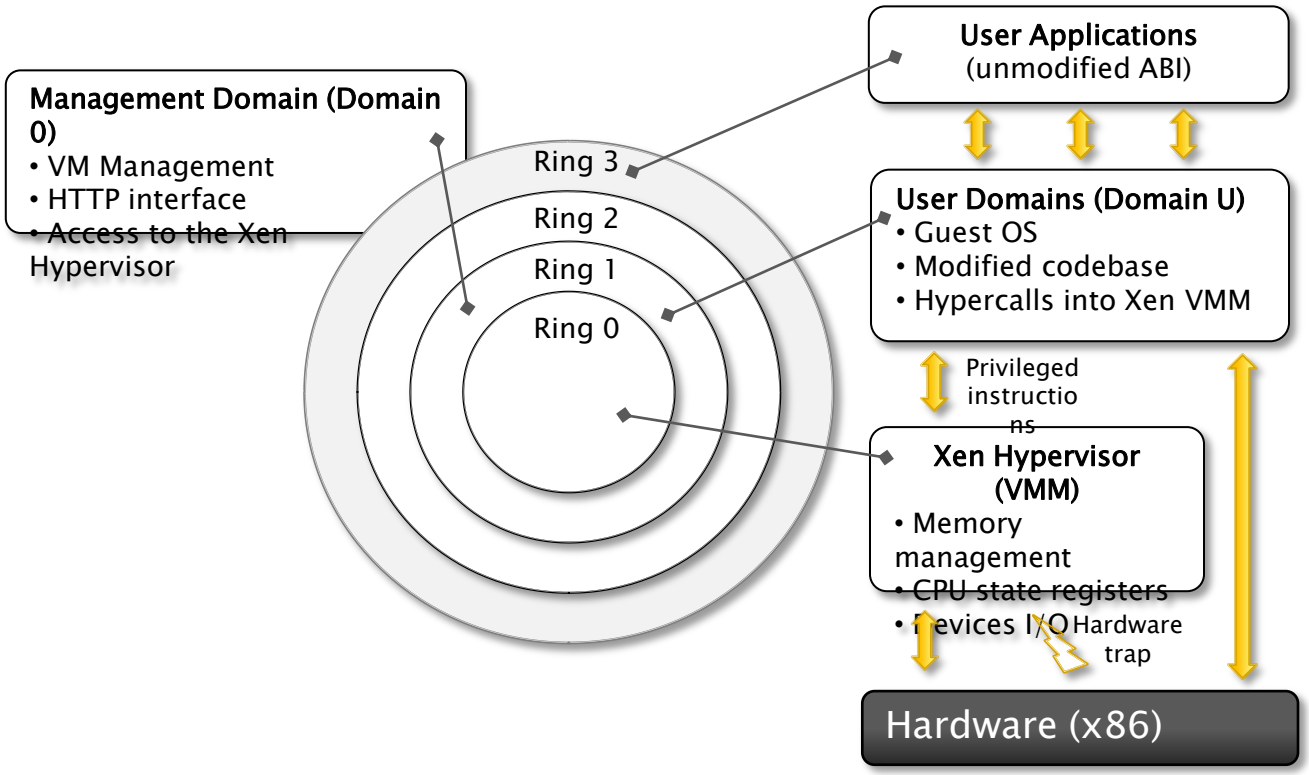

Storag
e
Virtual
Image
Guest
In memory
representation
VMM
Host emulation
Virtual
Machine
binary translation
instruction mapping
interpretation
……
Host

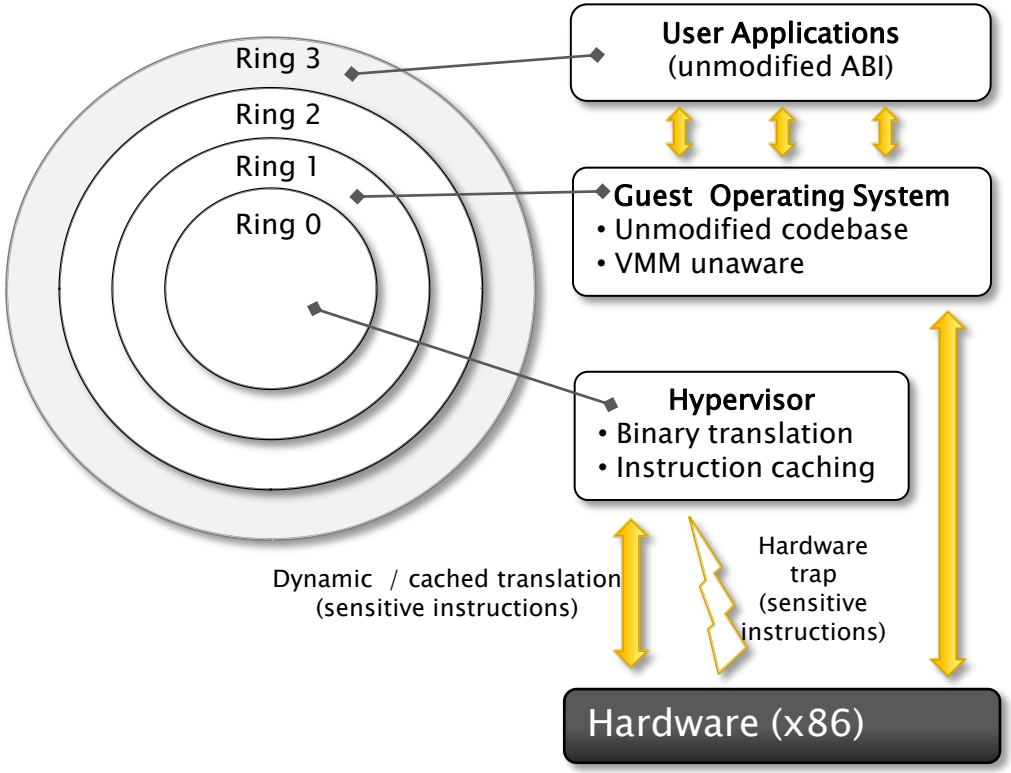

VM
VM
VM
VM
ISA
Virtual Machine Manager
ABI
Operative System
ISA
Hardware
VM
VM
VM
VM
ISA
Virtual Machine Manager
ISA
Hardware

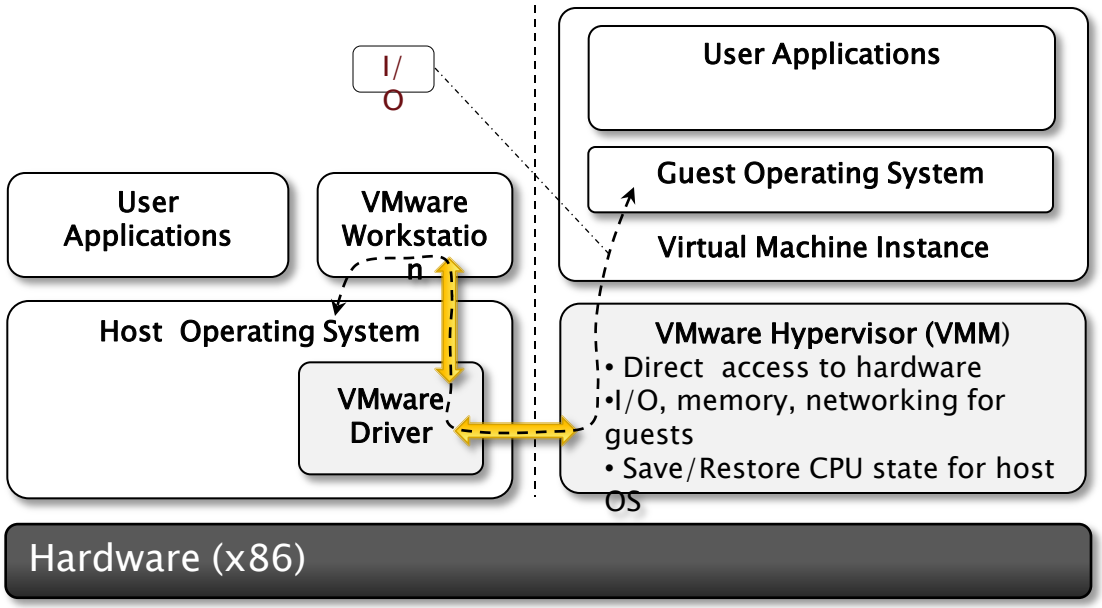

Virtual Machine Instance
ISA
Instructions (ISA)
Dispatcher
Interpreter
Routines
Allocator
Virtual Machine Manager

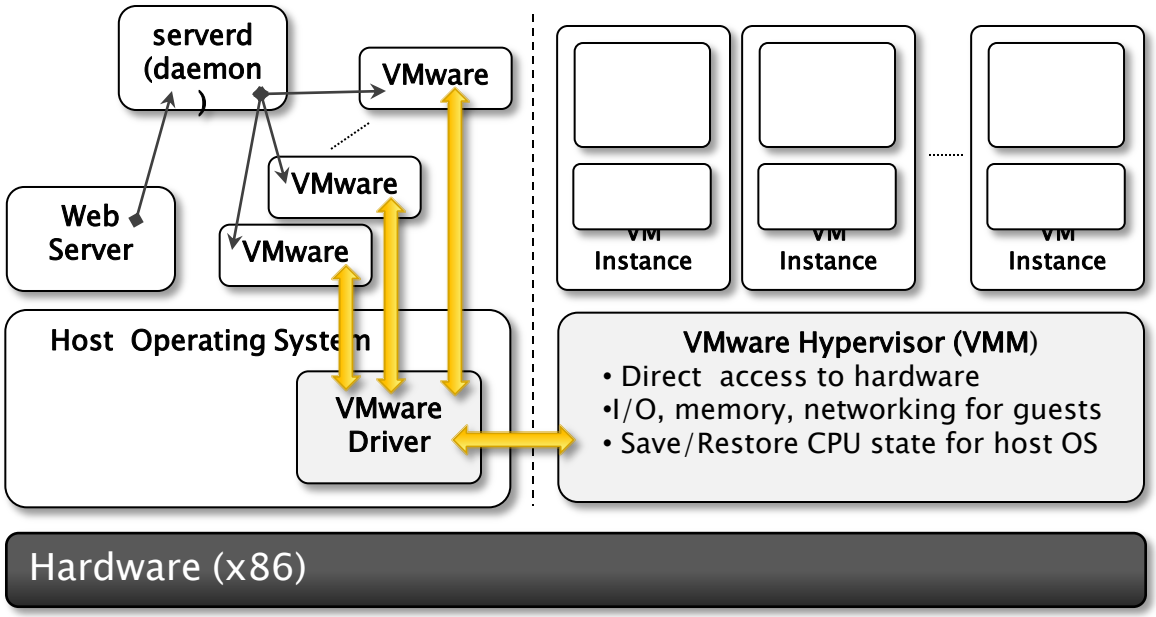

Privileged Instructions
Sensitive Instructions
User Instructions

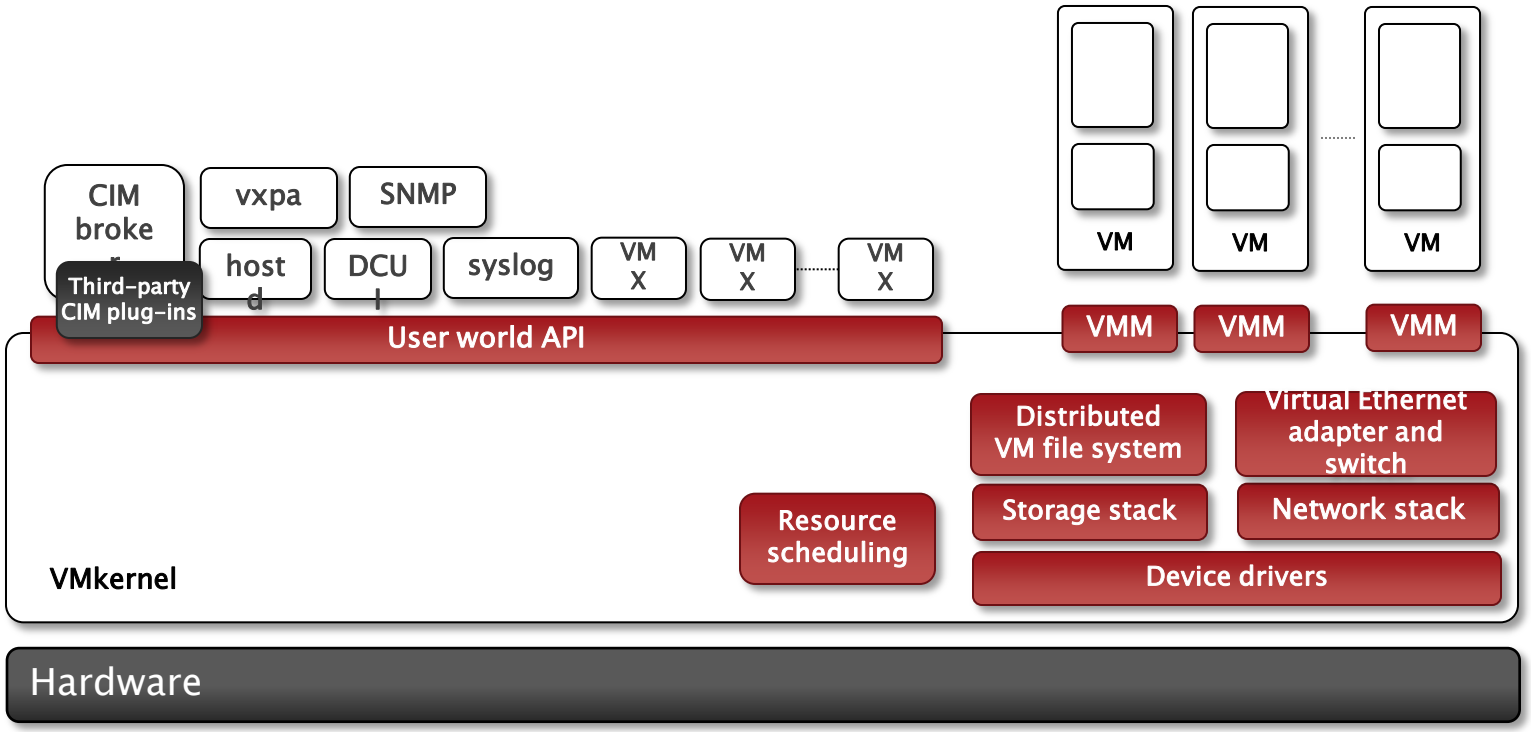

VM
VM
VM
VM
VM
VM
Virtual Machine Manager
Server A
(running)
Server B
(running)
Before
Migration
VM
VM
VM
VM
VM
VM
Virtual Machine Manager
Server A
(running)
Server B
(inactive)
After
Migration

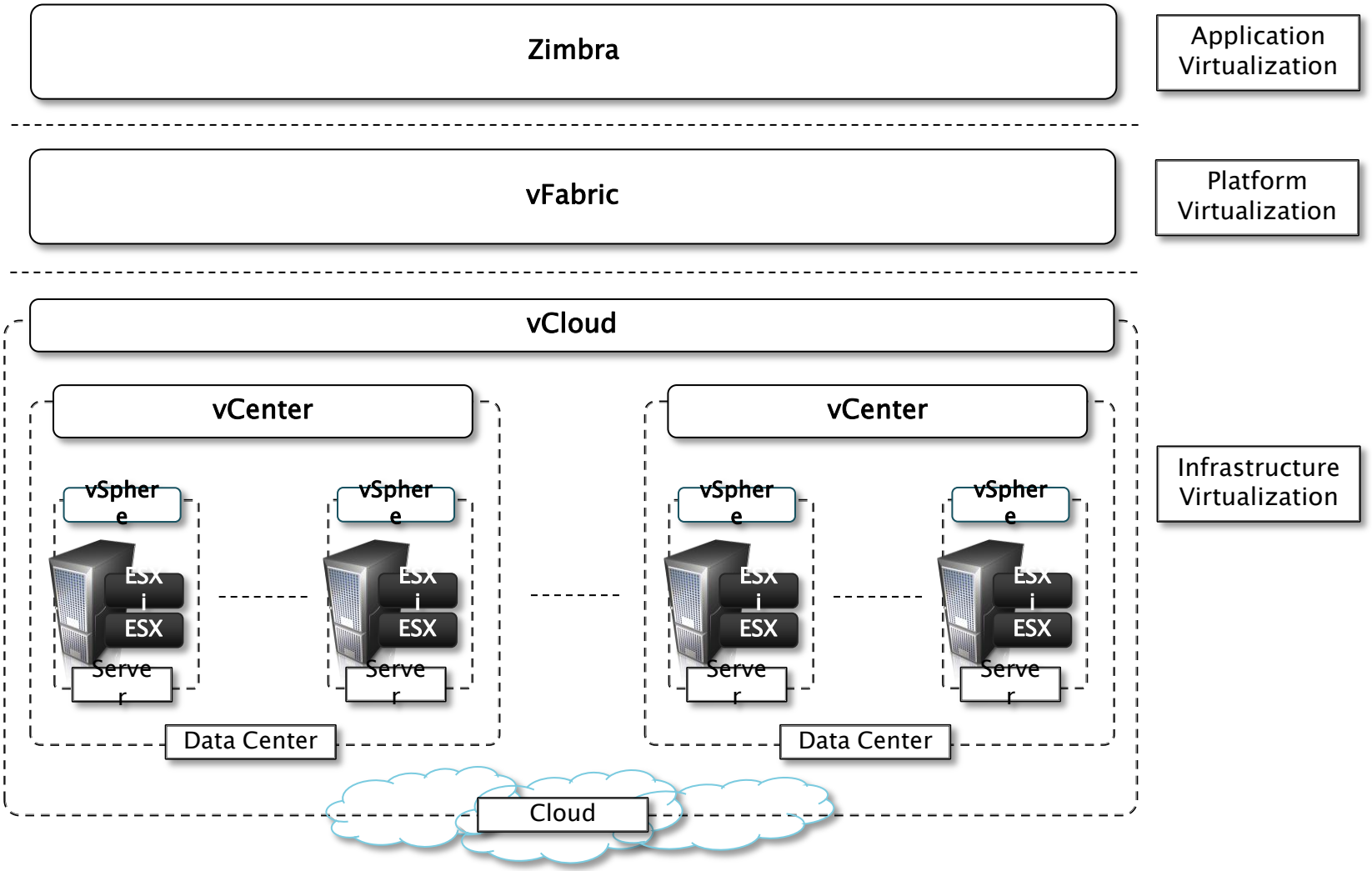

Management Domain (Domain 0)
• VM Management
• HTTP interface
• Access to the Xen Hypervisor
Ring 3
Ring 2
Ring 1
Ring 0
User Applications
(unmodified ABI)
User Domains (Domain U)
• Guest OS
• Modified codebase
• Hypercalls into Xen VMM
Privileged instructio ns
Xen Hypervisor (VMM)
• Memory management
• CPU state registers
• Devices I/O
Hardware trap
Hardware (x86)

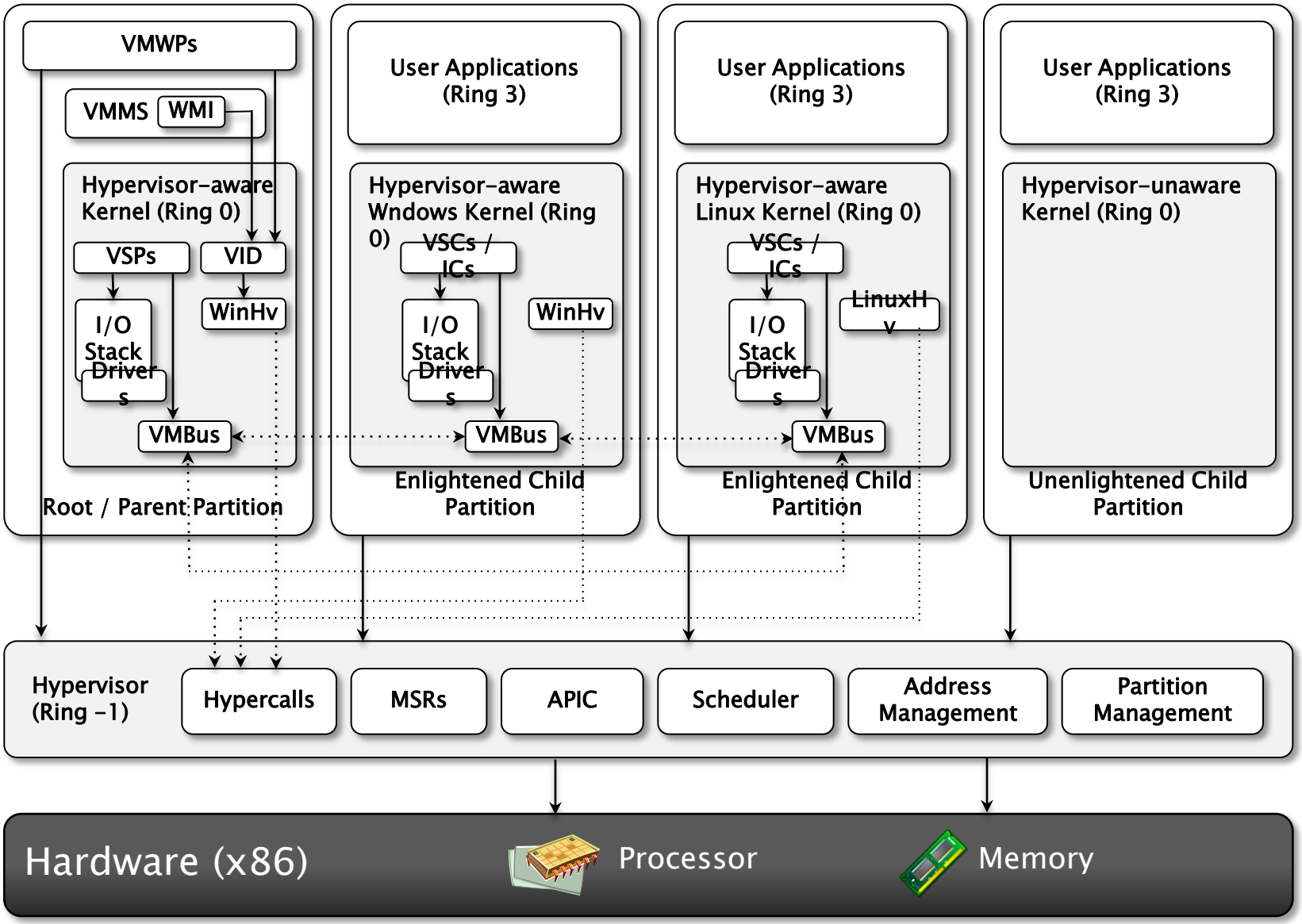

Ring 3
Ring 2
Ring 1
Ring 0
User Applications
(unmodified ABI)
Guest Operating System
• Unmodified codebase
• VMM unaware
Hypervisor
• Binary translation
• Instruction caching
Dynamic / cached translation
(sensitive instructions)
Hardware
trap
(sensitive
instructions)
Hardware (x86)


I/O
User Applications
Guest Operating System
Virtual Machine Instance
User Applications
VMware Workstation
Host Operating System
VMware Driver
VMware Hypervisor (VMM)
• Direct access to hardware
•I/O, memory, networking for guests
• Save/Restore CPU state for host OS
Hardware (x86)


serverd
(daemon
)
VMware
VMware
VMware
Web
Server
Host Operating System
VMware
Driver
VM
Instance
VM
Instance
VM
Instance
VMware Hypervisor (VMM)
• Direct access to hardware
•I/O, memory, networking for guests
• Save/Restore CPU state for host OS
Hardware (x86)


CIM broke
vxpa
SNMP
Third-party CIM plug-ins
host d
DCU I
syslog
VM X
VM X
VM X
VM
VM
VM
User world API
VMM
VMM
VMM
Distributed VM file system
Virtual Ethernet adapter and switch
Storage stack
Network stack
Resource scheduling
Device drivers
VMkernel
Hardware


Zimbra
Application Virtualization
vFabric
Platform Virtualization
vCloud
vCenter
vCenter
vSphere
vSphere
vSphere
vSphere
ESXi
ESX
Server
ESXi
ESX
Server
ESXi
ESX
Server
ESXi
ESX
Server
Infrastructure Virtualization
Data Center
Data Center
Cloud


VMWPs
VMMS
WMI
Hypervisor-aware Kernel (Ring 0)
VSPs
VID
I/O Stack
Drivers
WinHv
VMBus
Root / Parent Partition
User Applications (Ring 3)
Hypervisor-aware Wndows Kernel (Ring 0)
VSCs / ICs
I/O Stack
Drivers
WinHv
VMBus
Enlightened Child Partition
User Applications (Ring 3)
Hypervisor-aware Linux Kernel (Ring 0)
VSCs / ICs
I/O Stack
Drivers
LinuxHv
VMBus
Enlightened Child Partition
User Applications (Ring 3)
Hypervisor-unaware Kernel (Ring 0)
Unenlightened Child Partition
Hypervisor (Ring -1)
Hypercalls
MSRs
APIC
Scheduler
Address Management
Partition Management
Hardware (x86)
Processor
Memory

Migration into a cloud

Introduction

- disruptive techno-commercial model
- Answer the following questions
  - when and how to migrate one's application into a cloud?
  - what part or component of the IT application to migrate into a cloud and what not to migrate into a cloud?
  - what kind of customers really benefit from migrating their IT into the cloud?
- Definition
  - It is a techno-business disruptive model of using distributed large-scale data centers either private or public or hybrid offering customers a scalable virtualized infrastructure or an abstracted set of services qualified by service-level agreements (SLAs) and charged only by the abstracted IT resources consumed.

Promise of the cloud

Cloudonomics

- 'Pay per use' – Lower Cost Barriers
- On Demand Resources –Autoscaling
- Capex vs OPEX – No capital expenses (CAPEX) and only operational expenses OPEX.
- SLA driven operations – Much Lower TCO
- Attractive NFR support: Availability, Reliability

Technology

- 'Infinite' Elastic availability – Compute/Storage/Bandwidth
- Automatic Usage Monitoring and Metering
- Jobs/Tasks Virtualized and Transparently 'Movable'
- Integration and interoperability 'support' for hybrid ops
- Transparently encapsulated & abstracted IT features.

The Cloud Service Offerings and Deployment Models

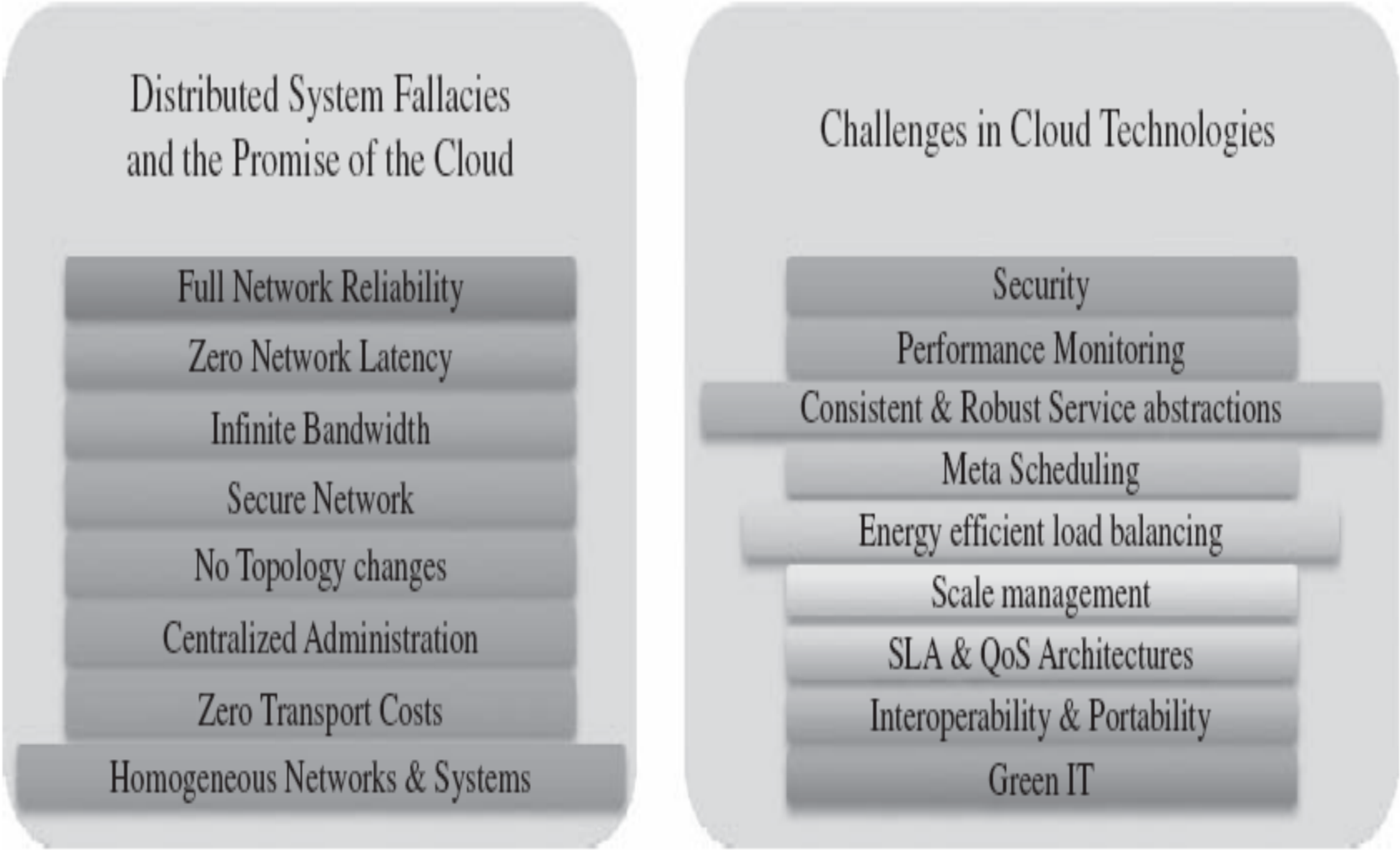
IaaS IT Folks	• Abstract Compute/Storage/Bandwidth Resources • Amazon Web Services[10,9] – EC2, S3, SDB, CDN, CloudWatch
PaaS Programmers	• Abstracted Programming Platform with encapsulated infrastructure • Google Apps Engine(Java/Python), Microsoft Azure, Aneka[13]
SaaS Architects & Users	• Application with encapsulated infrastructure & platform • Salesforce.com; Gmail; Yahoo Mail; Facebook; Twitter

Cloud Application Deployment & Consumption Models		
Public Clouds	Hybrid Clouds	Private Clouds

Examples

- IaaS
  - Amazon services
  - Elastic Cloud Compute (EC2)
    - small-instance
    - large-instance
    - extra-large instance
    - high-cpu instance
    - high-cpu medium instance
    - high-cpu extra-large instance
- SaaS
  - Gmail
  - Scalable storage space

Challenges in cloud



Why Migrate

- Reasons
  - Economic
  - Business
  - Technologic
- Five level of migration
  - Application
  - Code
  - Design
  - Architecture
  - Usage

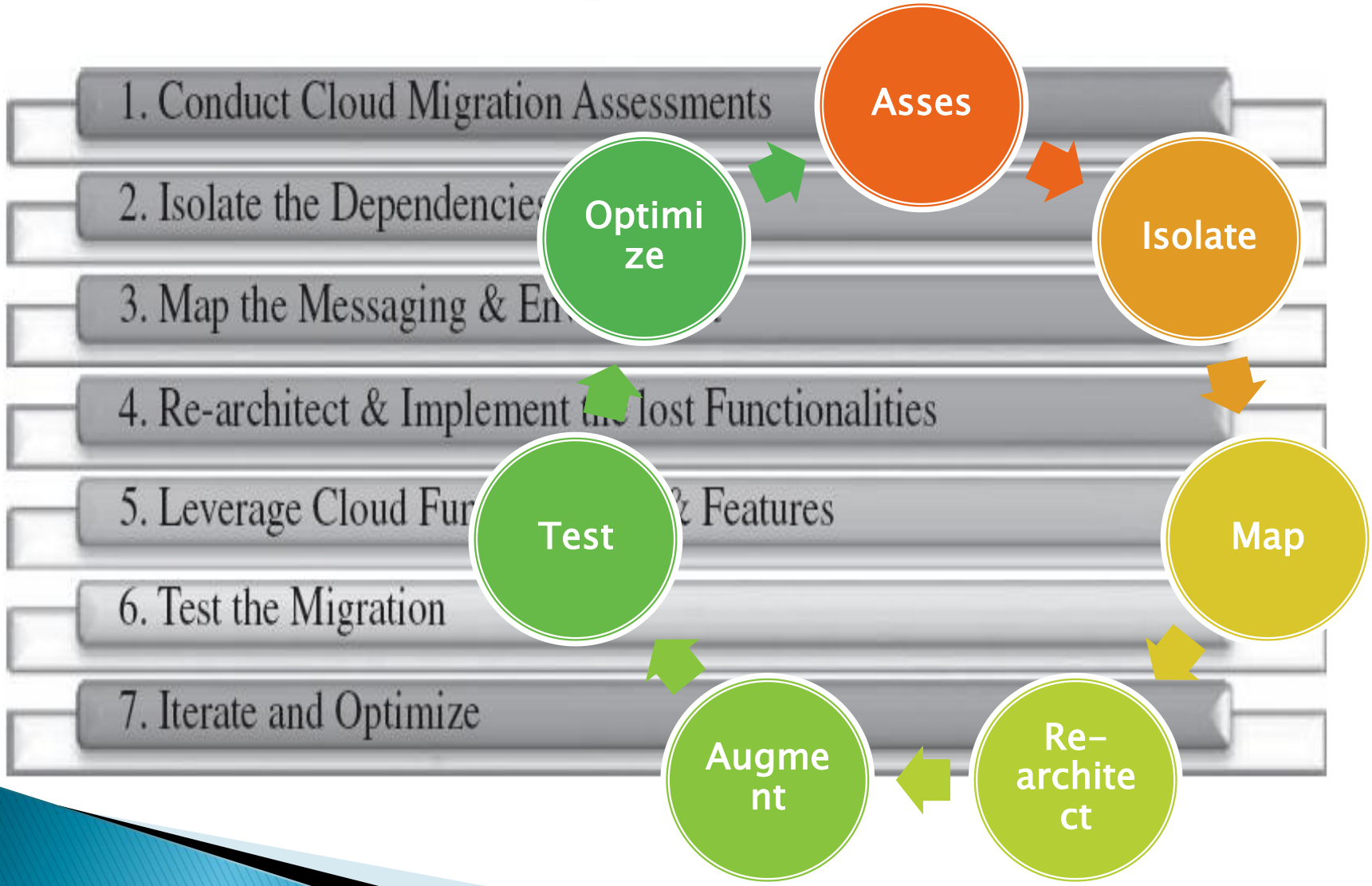
$$P \rightarrow P'_C + P'_l \rightarrow P'_{\mathrm{OFC}} + \mathrm{P}'_l$$

- Clean and independent application
- Code(design) needs to be modified and adapted
- Usage of application needs to be modified and adapted
- Hybrid Cloud

Cloudonomics

- economics and the associated trade-offs, of leveraging the cloud computing services
- Factors to migrate
  - Economic
    - CaPex
    - Opex
    - When?
      - cost of using cloud + cost of migration < cost of using captive dc
  - Licensing issues
  - SLA compliances
  - pricing of the cloud service offerings
    - Elasticity and pricing variability

Model of Migration



Migration in AWS

- Assessment
  - Isolate dependency
- Reference migration architecture
- Data migration
- Application migration
- Leveraging AWS features
- Optimize for Cloud

Migration risk

- Is a challenge
- Identify in test phase
- Mitigate in optimization phase
- Types
  - General
    - Performance monitoring & tuning
    - Disaster recovery
    - Compliance with standards and governance issues
    - Licensing issues
    - QoS
    - Portability and interoperability
    - …
  - Security-related
    - Issues of security at various level of app
    - issues of trust and issues of privacy
    - Right execution logs
    - Consistent identity management
    - …

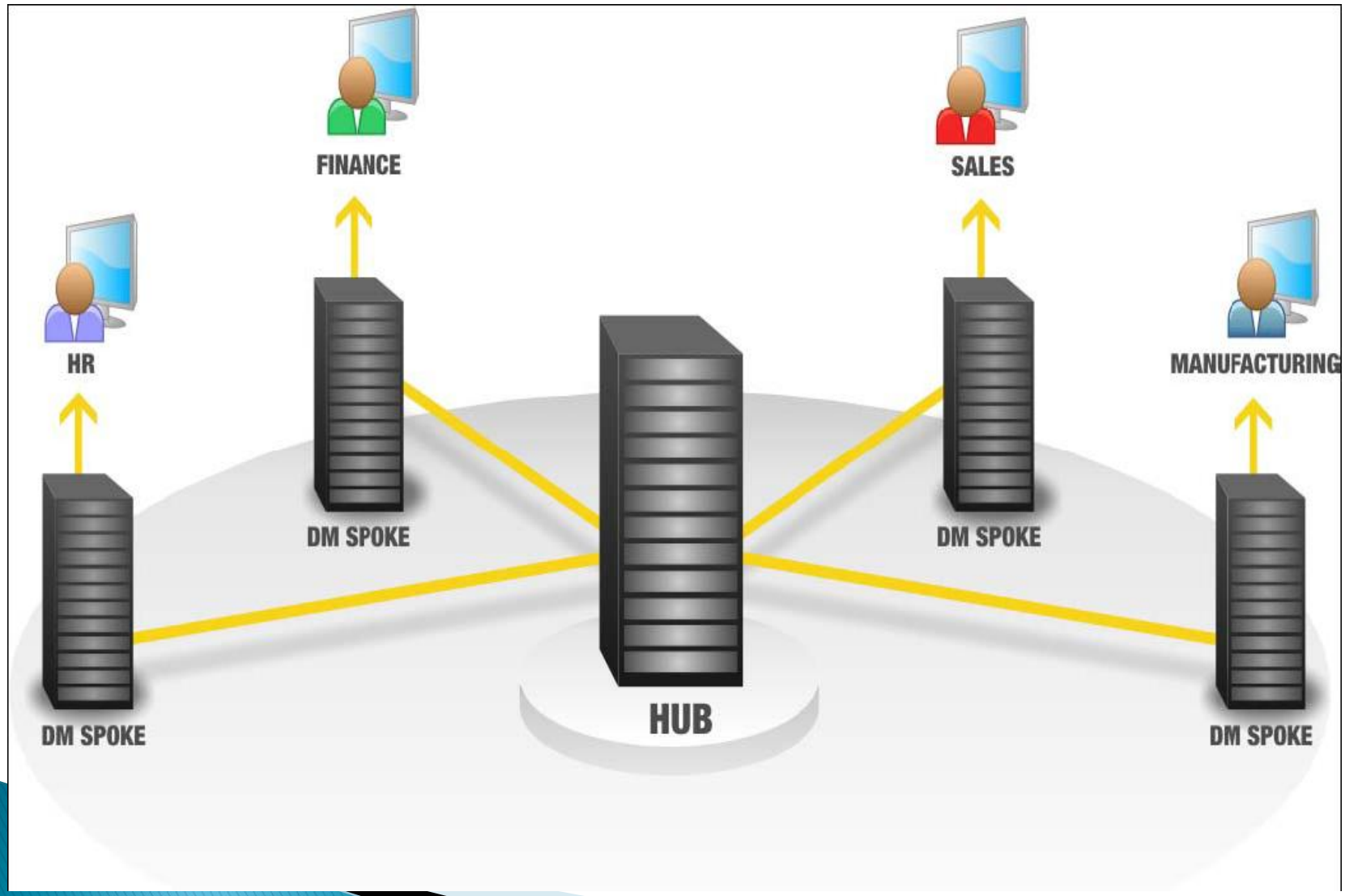
Enriching the integration as a service paradigm for the cloud era

Introduction

- Integration
  - E2E (Enterprise to Enterprise)
  - E2C (Enterprise to Cloud)
  - C2C (Cloud to Cloud)

- Systems
  - One-to-many
  - One-to-one
  - Many-to-one

Evolution of SaaS



Challenges of SaaS paradigm

- Challenges
  1. Controllability
  2. Visibility & flexibility
  3. Security and Privacy
  4. High Performance and Availability
  5. Integration and Composition
  6. Standards
- Private cloud,…
- Integration
  - Real time data and functionality
- API
  - Too coding
- Data Transmission security
- Impact of the clouds

Approaching the SaaS integration

- Integration middleware
  - EAI
    - Enterprise Application Integration
  - ESB
    - Enterprise Service Bus for service integration
    - loosely coupled, in a cloud
  - EDB
    - Enterprise Data Bus for data integration
  - MOM
    - Message Oriented Middleware for integration application via Message passing
  - CEP
    - Complex Event Processing engines
    - decoupled

Why SaaS Integration is hard?

- Data synchronization
- Constraint of SaaS
  - Dynamic nature of the SaaS interfaces
  - Dynamic nature of the metadata
  - Managing assets outside of the firewall
  - Move Massive amounts of information
- Complicated integration
  - New integration scenarios
  - Limited access
    - Controllability, Flexibility, Visibility
  - Dynamic resources
    - Tightly coupled
  - Performance

Integration Scenario

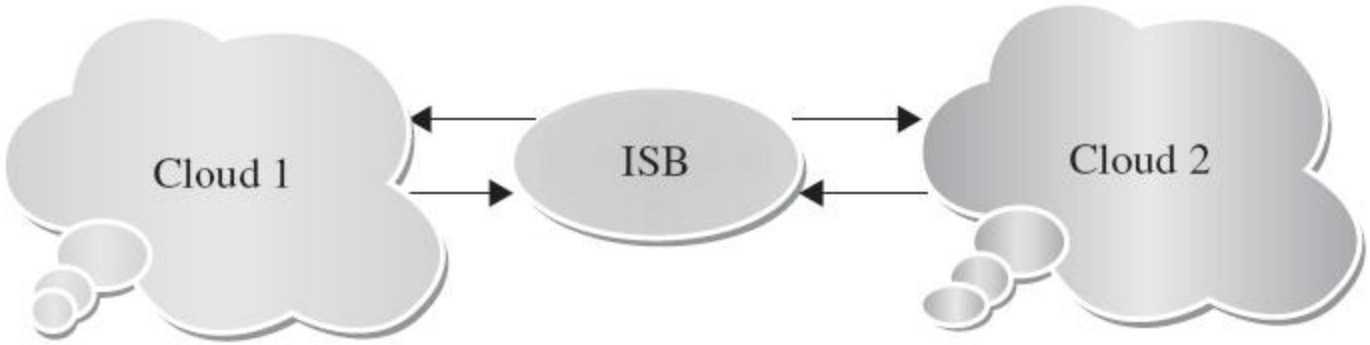
- Integration model
  - Local to local
  - Local to cloud
  - Cloud to cloud
- Three major scenario
  - Public cloud
  - Homogeneous cloud
  - Heterogeneous cloud

Integration with public cloud

- In one cloud
- Integration middleware
  - ESB or ISB
- Two app owned by different companies
- May be in single server

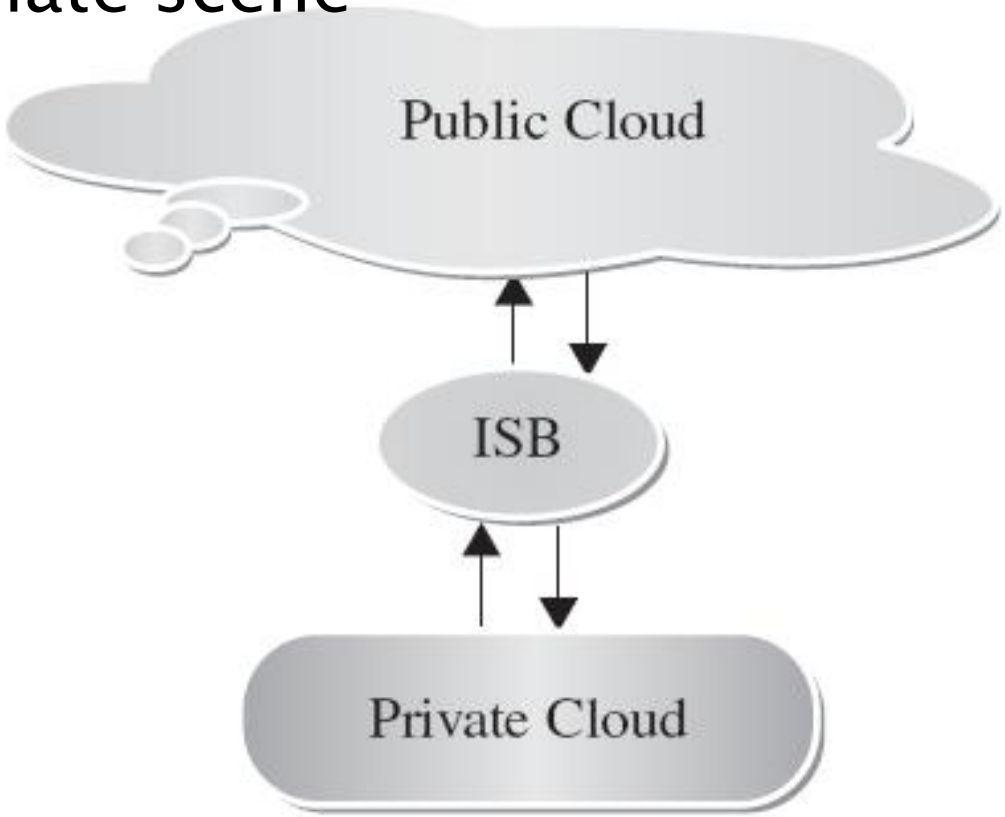
Integration homogeneous clouds

- Two geographically separate
- Middleware in 1 or 2 or another cloud



Integration Heterogeneous clouds

- Public and private
- Dominate scene



Integration Methodology

- Three type cloud integration
  i. Traditional Enterprise Integration Tools can be empowered with special connectors to access Cloud-located Applications
     - the most likely approach for IT organizations
  ii. Traditional Enterprise Integration Tools are hosted in the Cloud
     - Good for C2C
  iii. Integration-as-a-Service (IaaS) or On-Demand Integration Offerings
     - On-premise to cloud, cloud to cloud and on-premise to on-premise
     - Informatica on demand is an example

Characteristic of integration

- Connectivity
- Semantic mediation
- Data mediation
  - Data transformation
- Data Migration
- Data Integrity
- Data Security
- Governance

Integration Engineering Lifecycle

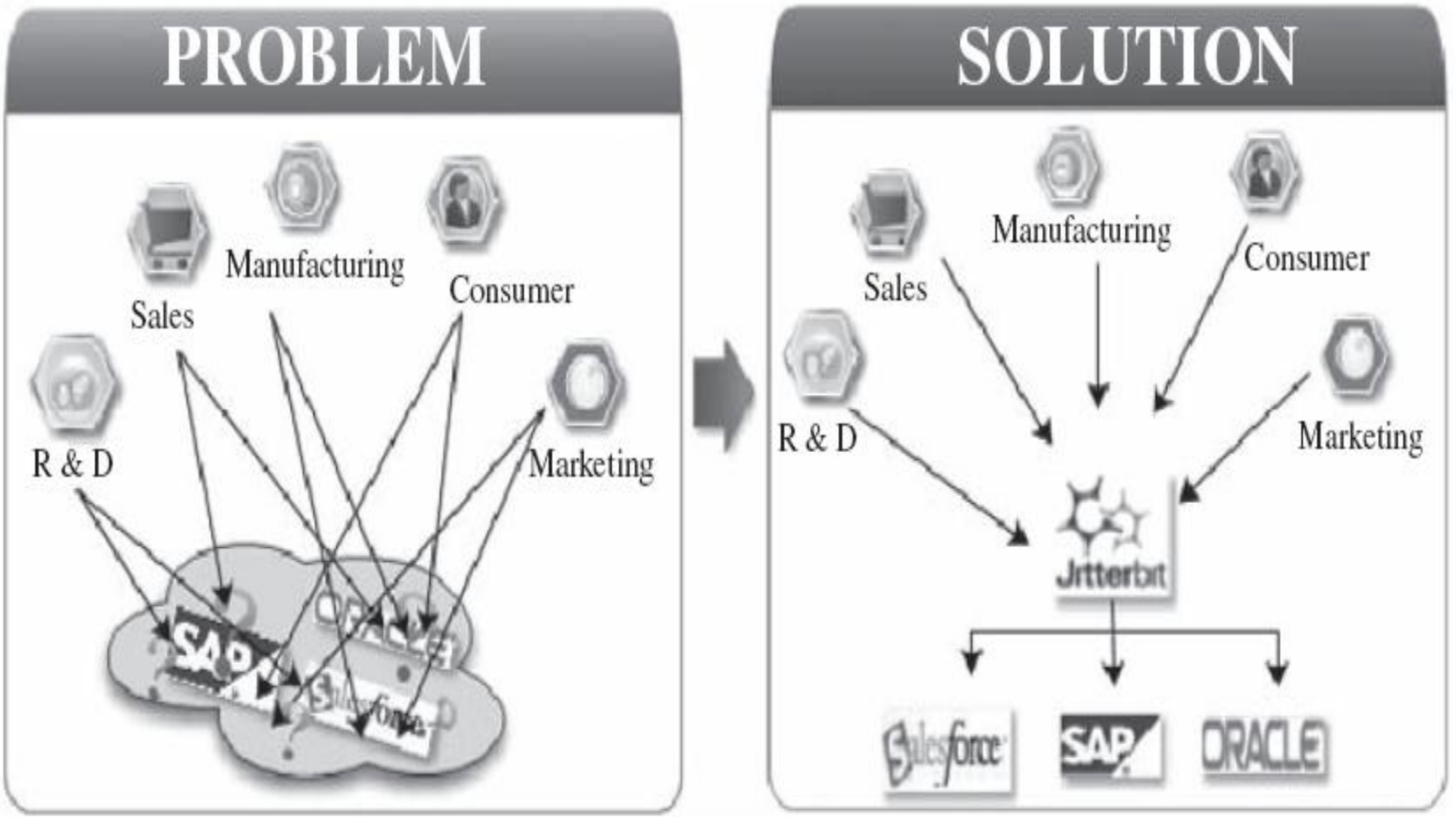
- Understanding
  - Semantic understanding of source and target system
- Definition
  - Information represent, ownership, physical attributes
- Design
  - Visual mapping technology
- Implementation
  - Connecting source and target systems
- Test
  - integration is properly designed and implemented

Products and platform

- Jitterbit
- Bommi software
  - On demand
- Bungee connect
- OpenSource connect
- SnapLogic
  - Free community
  - Professional
- Pervasive DataCloud
- Bluewolf
  - Proactive monitoring and consulting services
- Online MQ
- CloudMQ
- Linxter

Jitterbit

- Enable integration among data, apps, web services



Pervasive DataCloud

- Multi-tenant platform
- Deliver
  - Integration as a Service
  - Package turnkey integration
  - Support every integration scenario
  - Connectivity to hundreds of different application and data stores
- Is platform for deploy applications that are
  - Scalable
    - multi-tenant architecture
  - Flexible
    - SaaS-to-SaaS, SaaS to on-premise ,…
  - Easy to access and configure
    - Via web browser
  - Robust
  - Secure
    - automatic update, monitoring,...
  - Affordable
    - Pay-as-you-go model

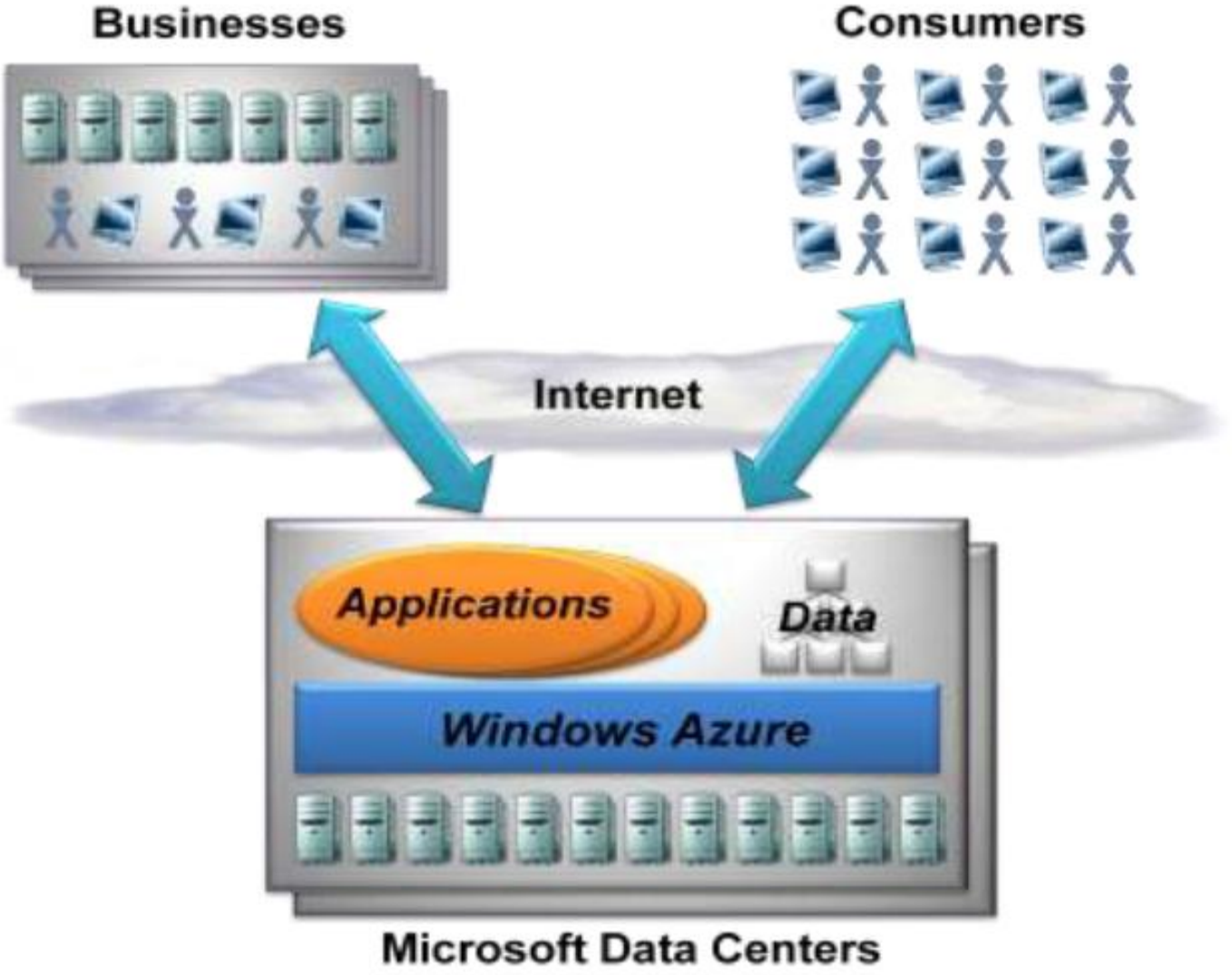
Online MQ

- Internet based queuing system
- Send/Receive message over network
- Cloud messaging queuing service
- Advantages
  - Ease of use
  - No maintenance
  - Load balancing
    - Multiple instance
  - High availability
    - clustering
  - Easy integration
    - SOAP, JMS-compatible

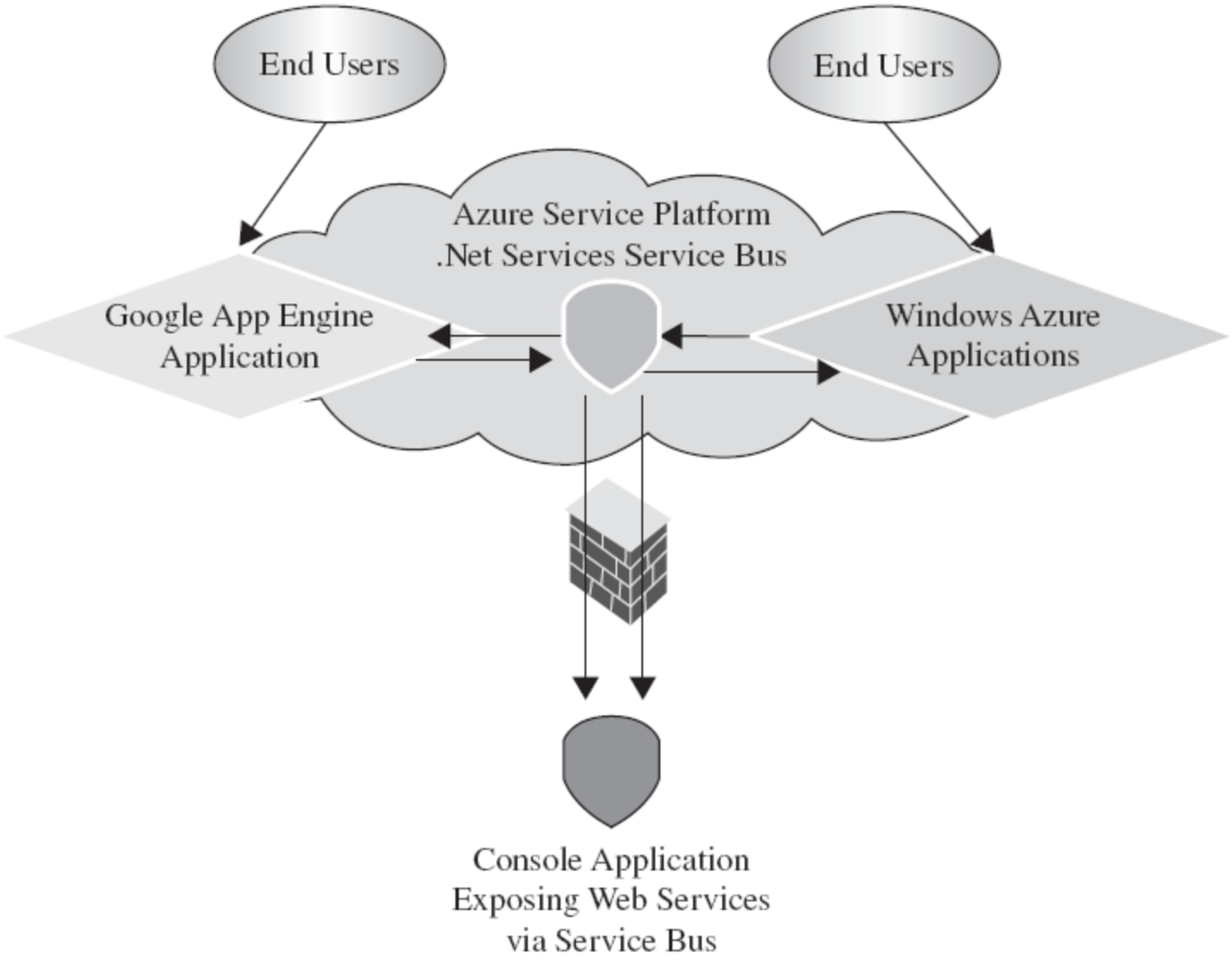
Informatica on-demand service

- innovative on-demand data integration solutions
- As-a-service delivery model
- Benefits
  - Rapid development and deployment
    - with zero maintenance of the integration technology
  - Automatically upgraded and continuously enhanced by vendor
  - Proven SaaS integration solutions
  - Proven data transfer and translation technology
- No complex software update
- No additional fee
- Patching, versioning has no cost

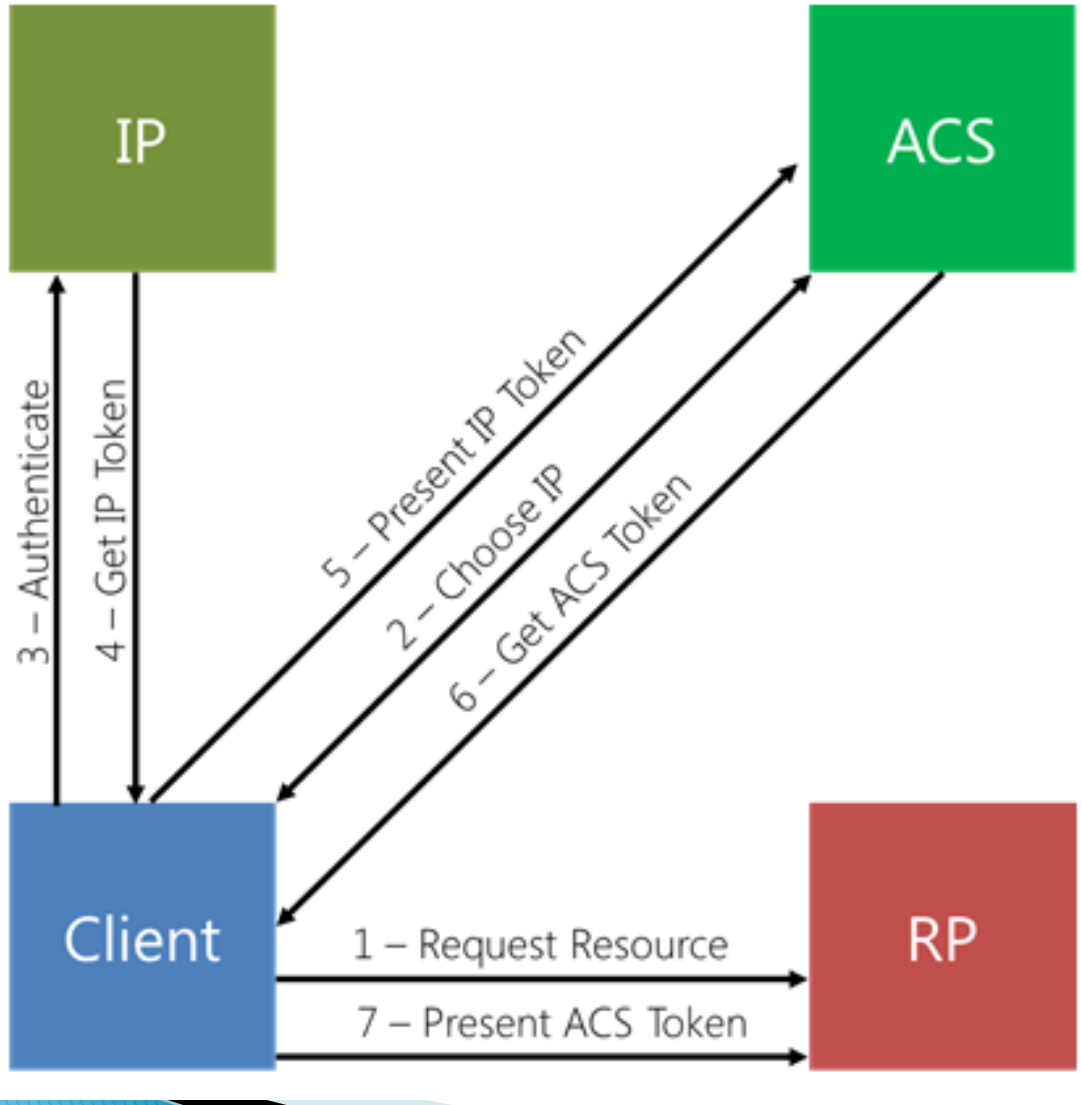
Microsoft ISB



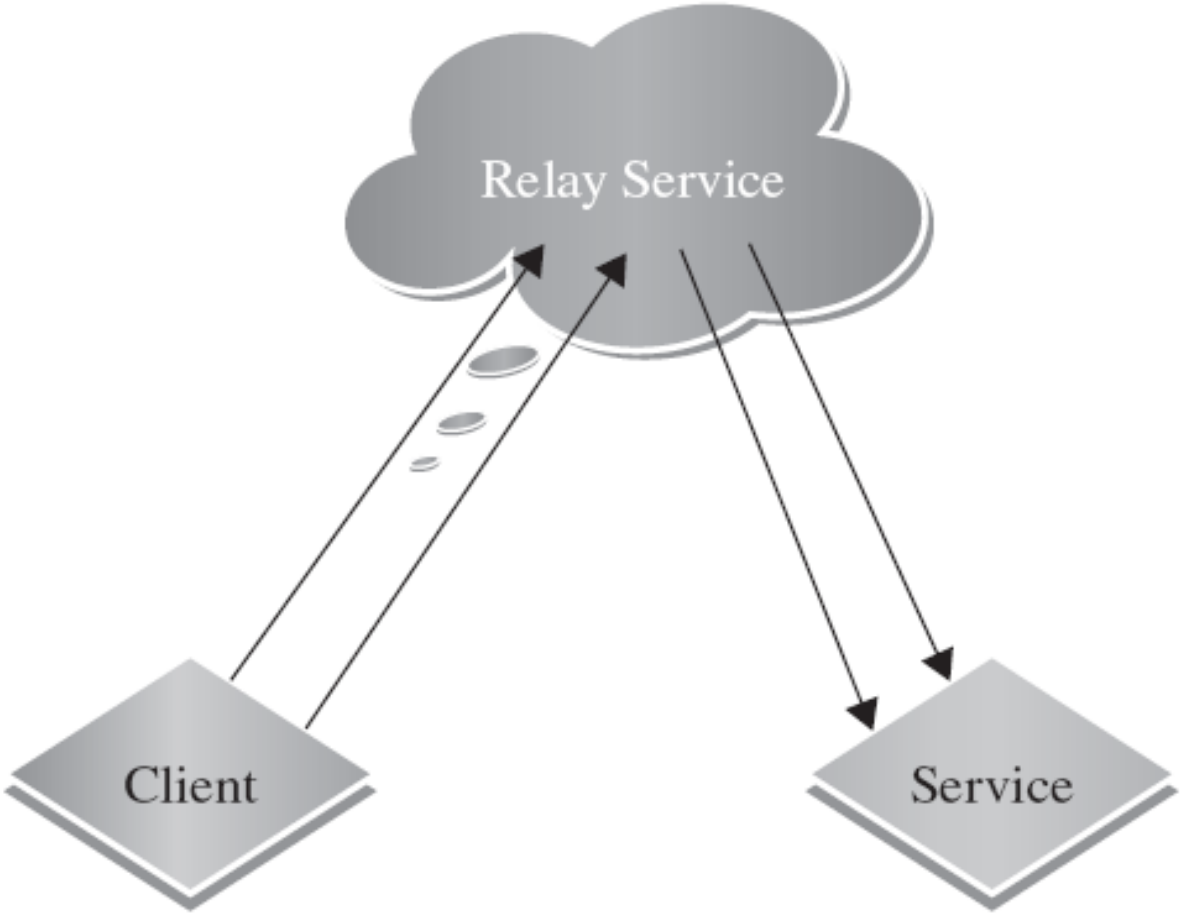
.Net Service Bus



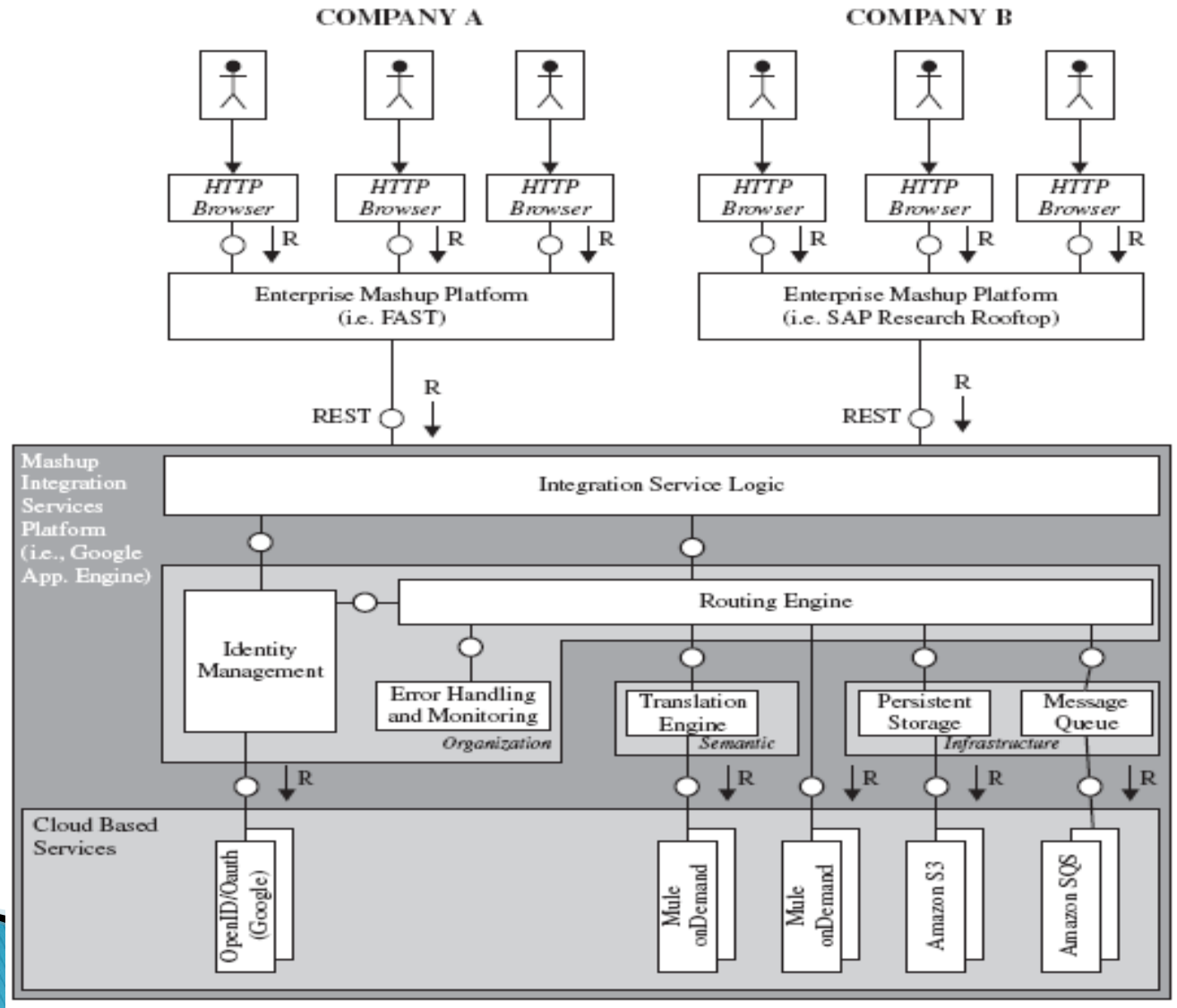
.Net Access Control Service



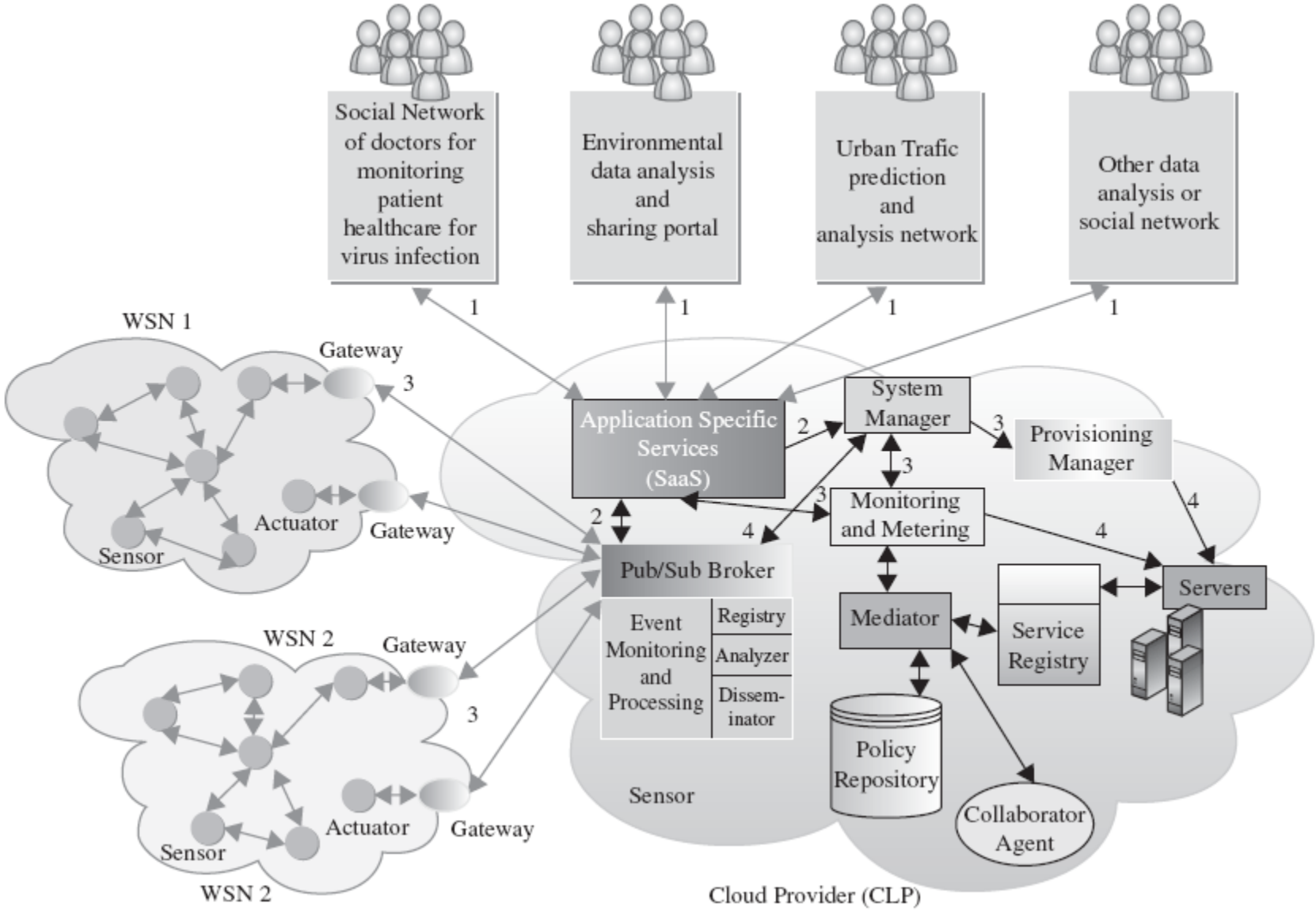
Relay Service



Enterprise mashup platform



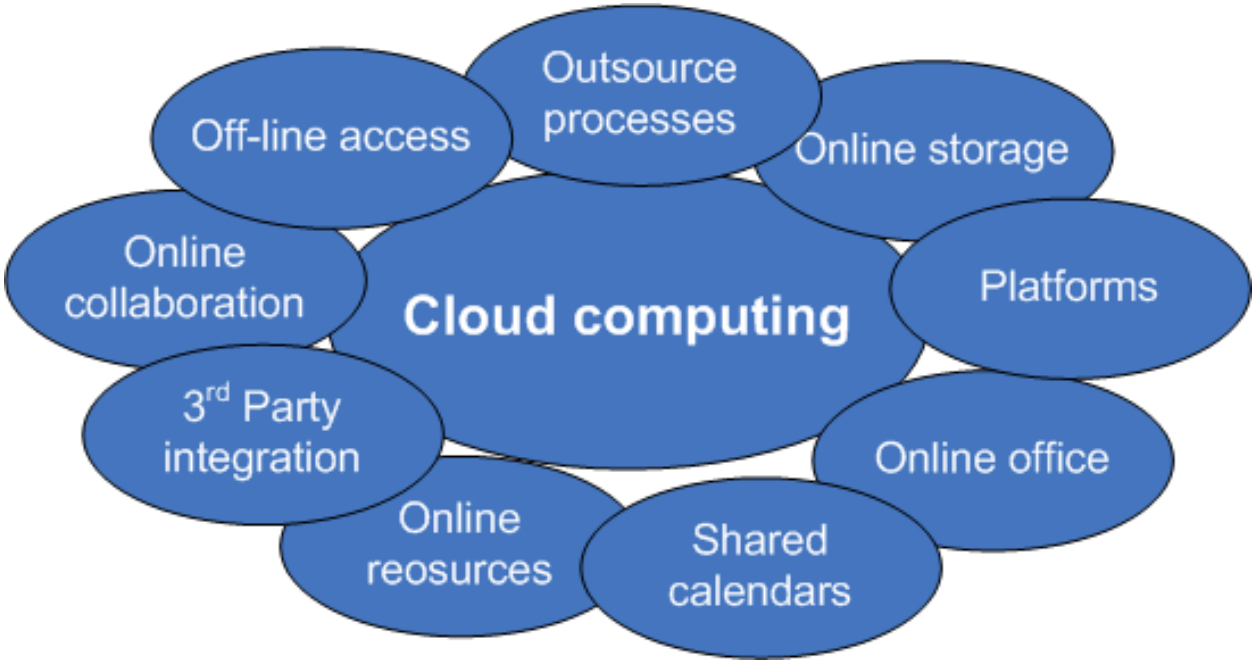
Sensor-Cloud Integration



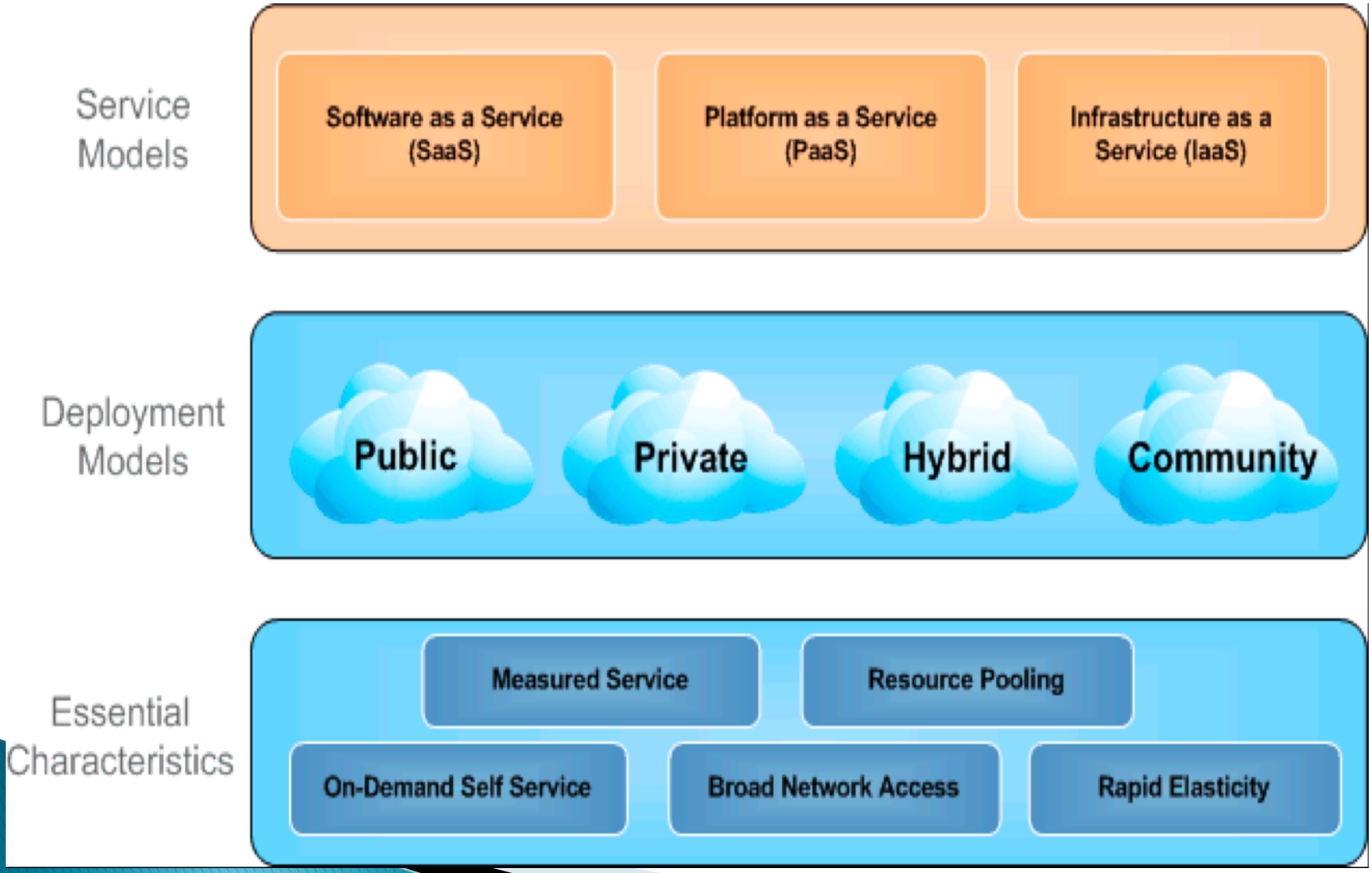
The Enterprise Cloud Computing Paradigm

Introduction

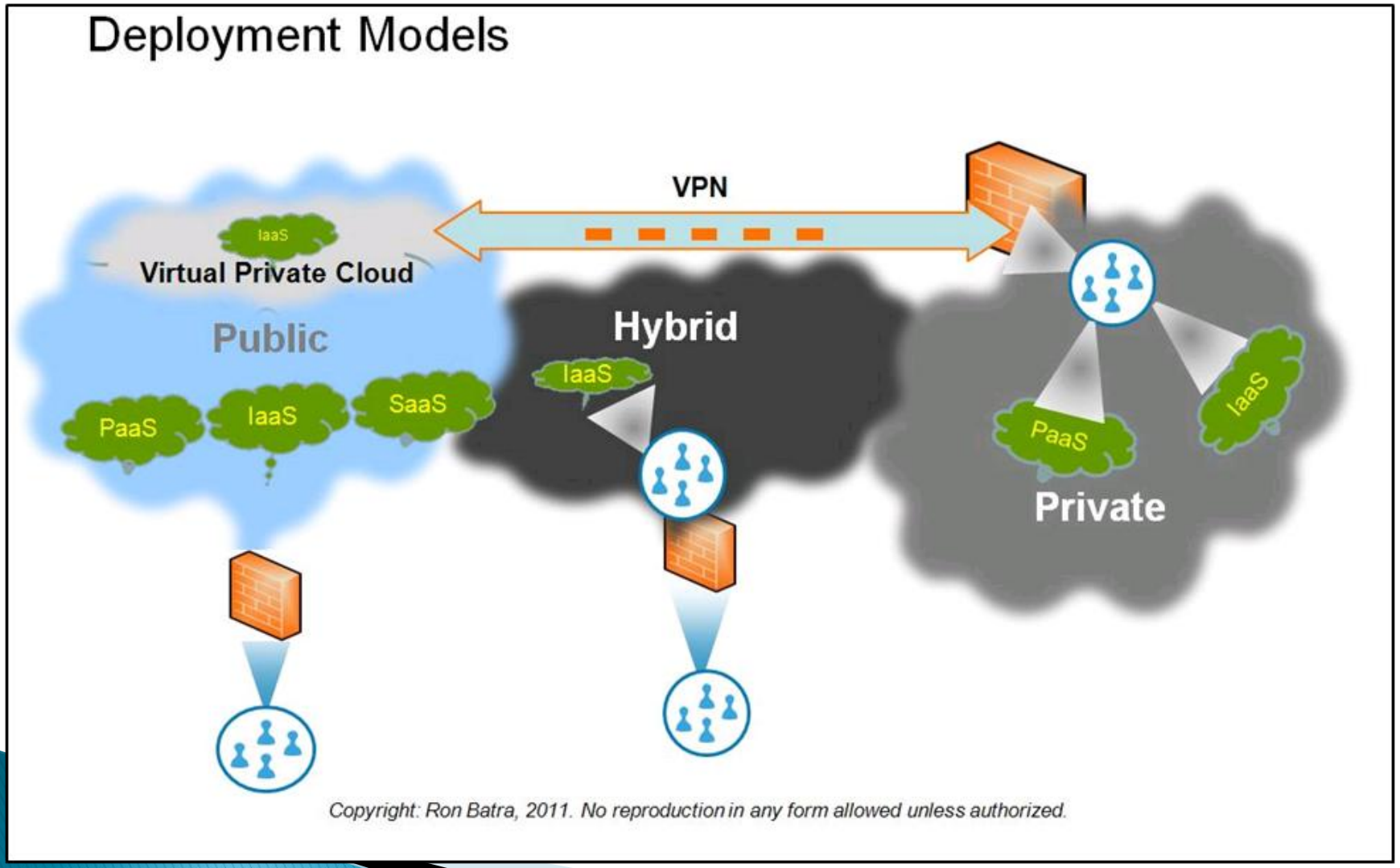
- Enterprise Cloud Computing Paradigm



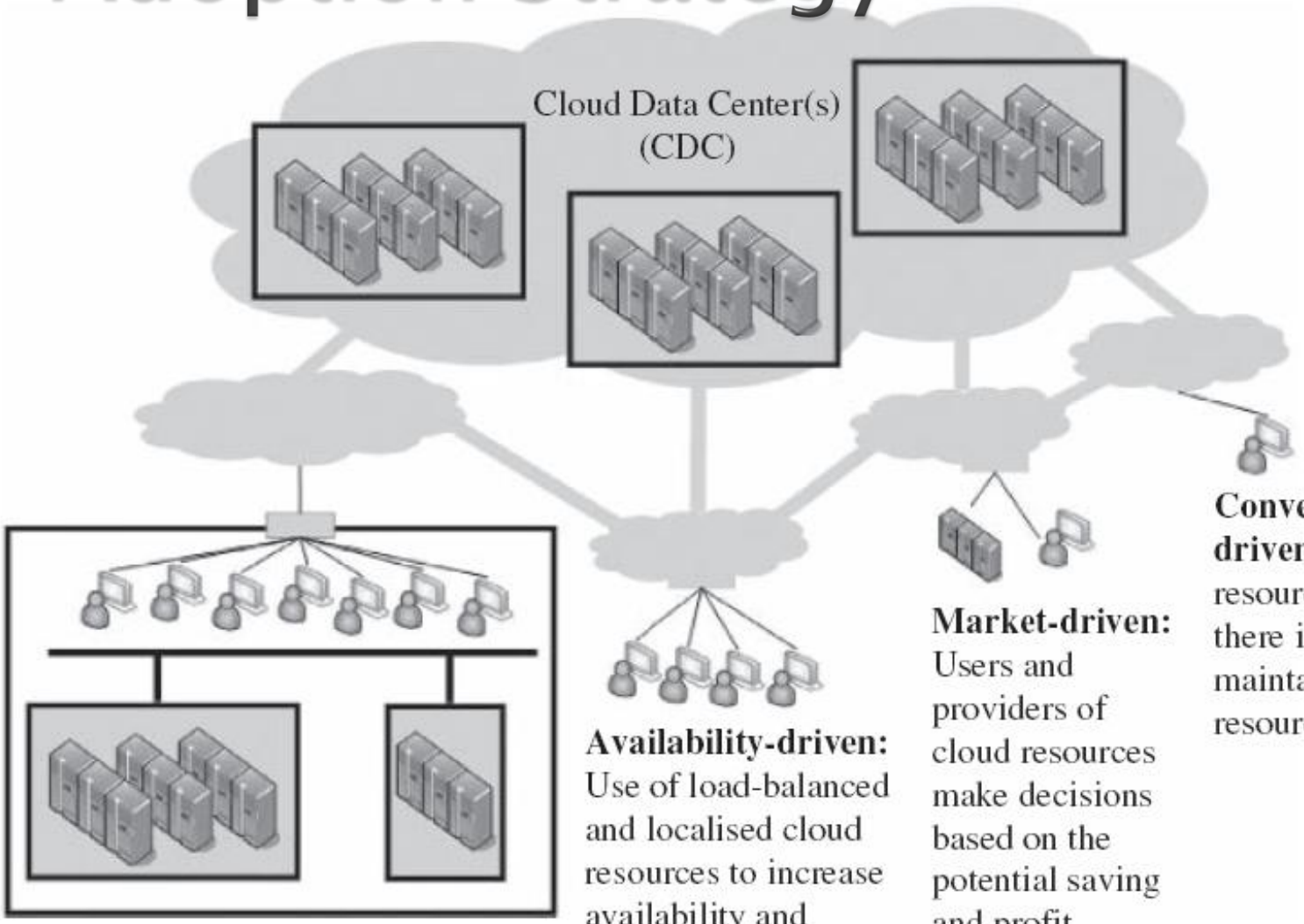
Deployment Models



Deployment Models



Adoption Strategy



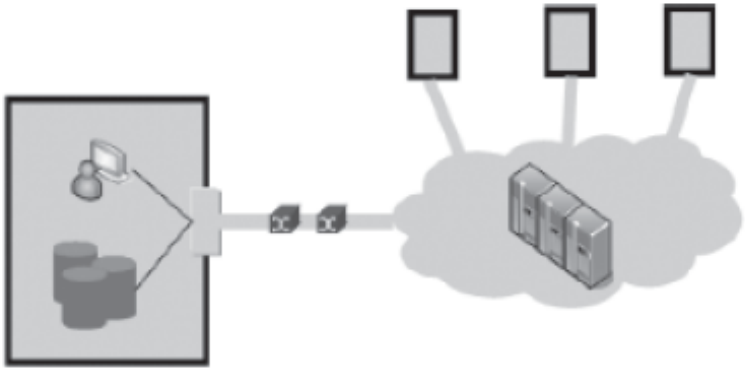
Scalability-driven: Use of cloud resources to support additional load or as back-up.

Availability-driven: Use of load-balanced and localised cloud resources to increase availability and reduce response time

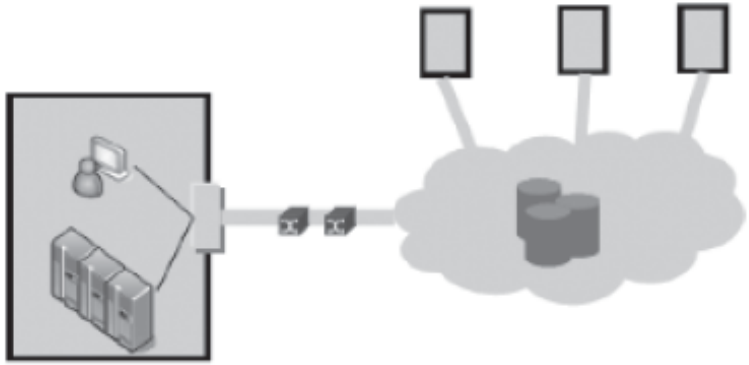
Market-driven: Users and providers of cloud resources make decisions based on the potential saving and profit

Convenience-driven: Use cloud resources so that there is no need to maintain local resources.

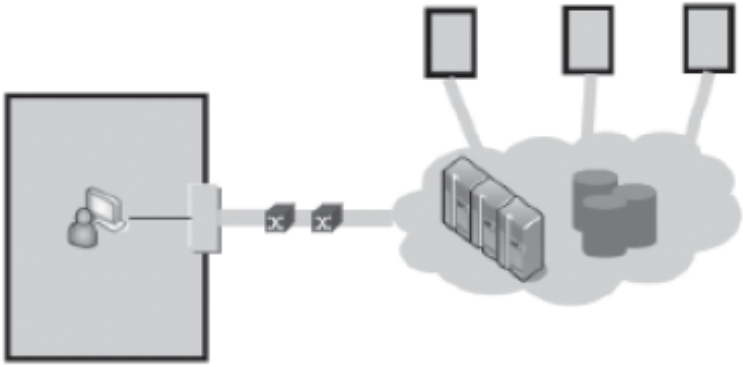
Consumption Strategy



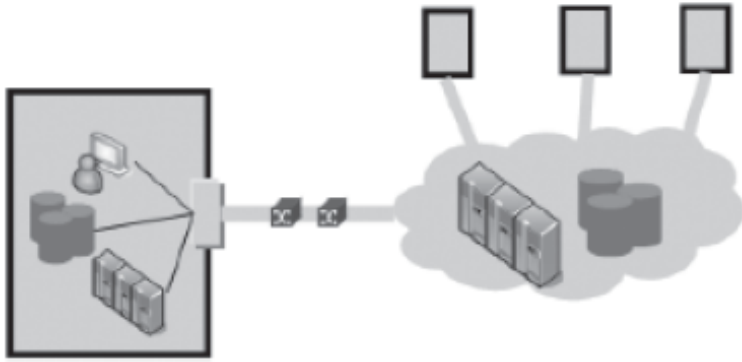
(1) Software Provision: Cloud provides instances of software but data is maintained within user's data center



(2) Storage Provision: Cloud provides data management and software accesses data remotely from user's data center



(3) Solution Provision: Software and storage are maintained in cloud and the user does not maintain a data center



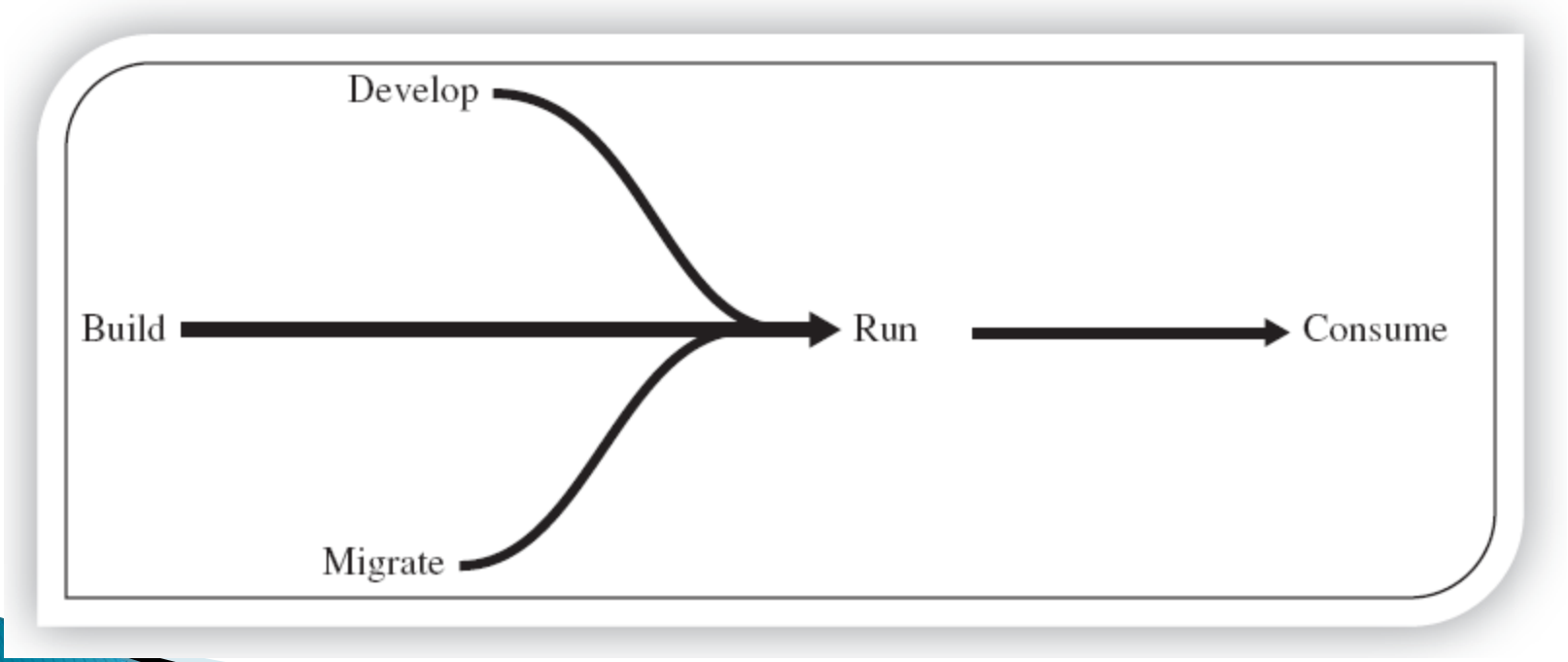
(4) Redundancy Services: Cloud is used as an alternative or extension of user's data center for software and storage

Issues for Enterprise Applications in cloud

- ERP
- Capabilities of ERP
  - Transactional Capabilities
    - OLTP (Online Transaction Processing)
    - manage transaction oriented applications(relational databases)
    - ACID properties, write/update-intensive
    - CRM (Customer Relationship Management)
  - Analytical Capabilities
    - OLAP (Online Analytical Processing)
    - Analysis, reporting, decision support
    - Read only
    - Data-intensive
    - BI (Business Intelligence)

(ERP) Transition Challenges

- Five stage of the cloud



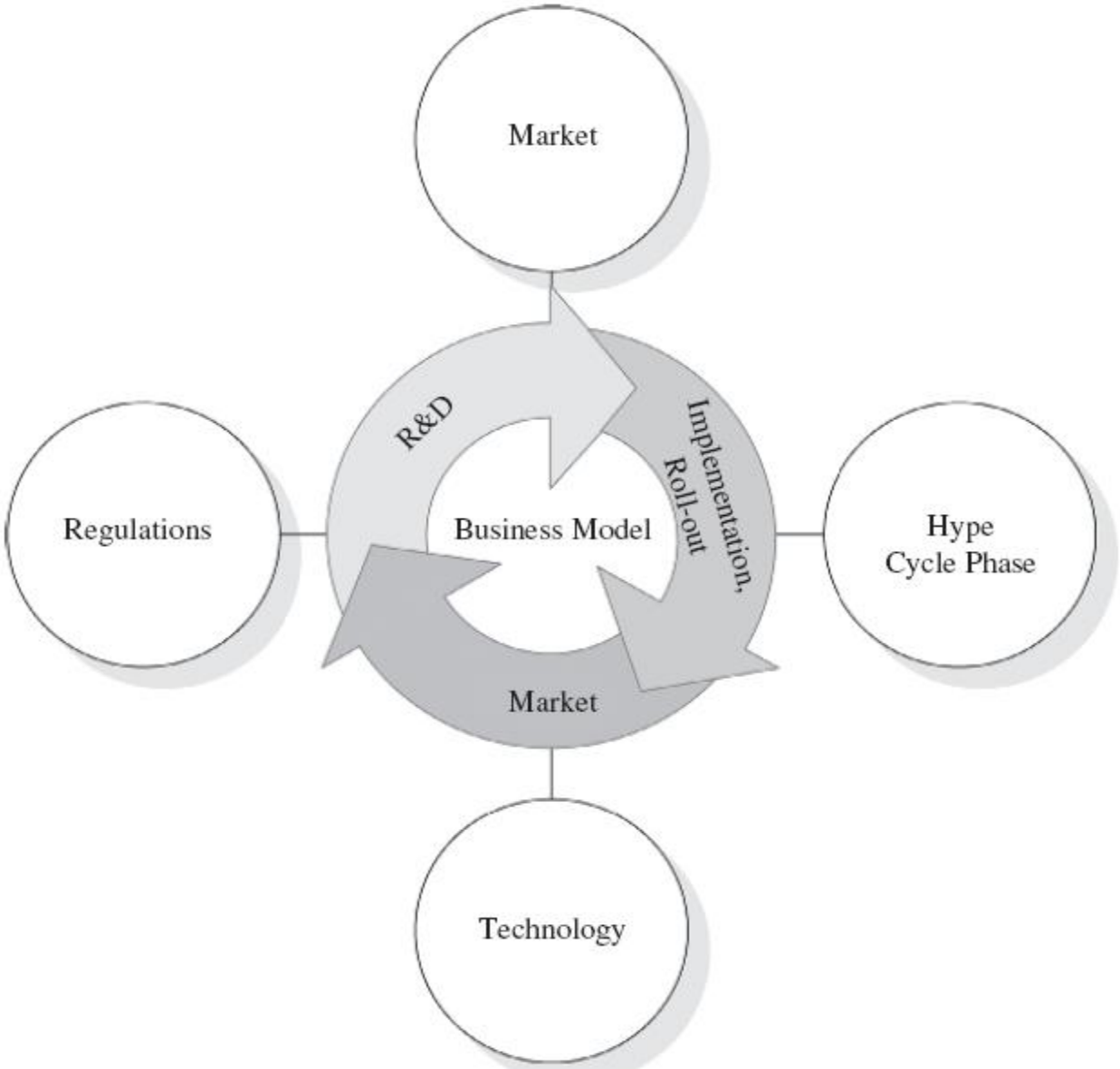
Enterprise Cloud Technology & Market Evolution

- Standard
  - Vendor lock-in
  - OGF OCCI for compute clouds
  - SNIA CDMI for storage and data management
  - DMTF Virtualization Management (VMAN)
  - DMTF Cloud Incubator
  - Drives adoption, Drives the market, Third party vendor
- SLA
  - Lack of control
  - Primitive vs. Sophisticated
- Cloud Service Brokerage (CSB)
  - Cloud Service Intermediation
  - Aggregation
  - Cloud service Arbitrage

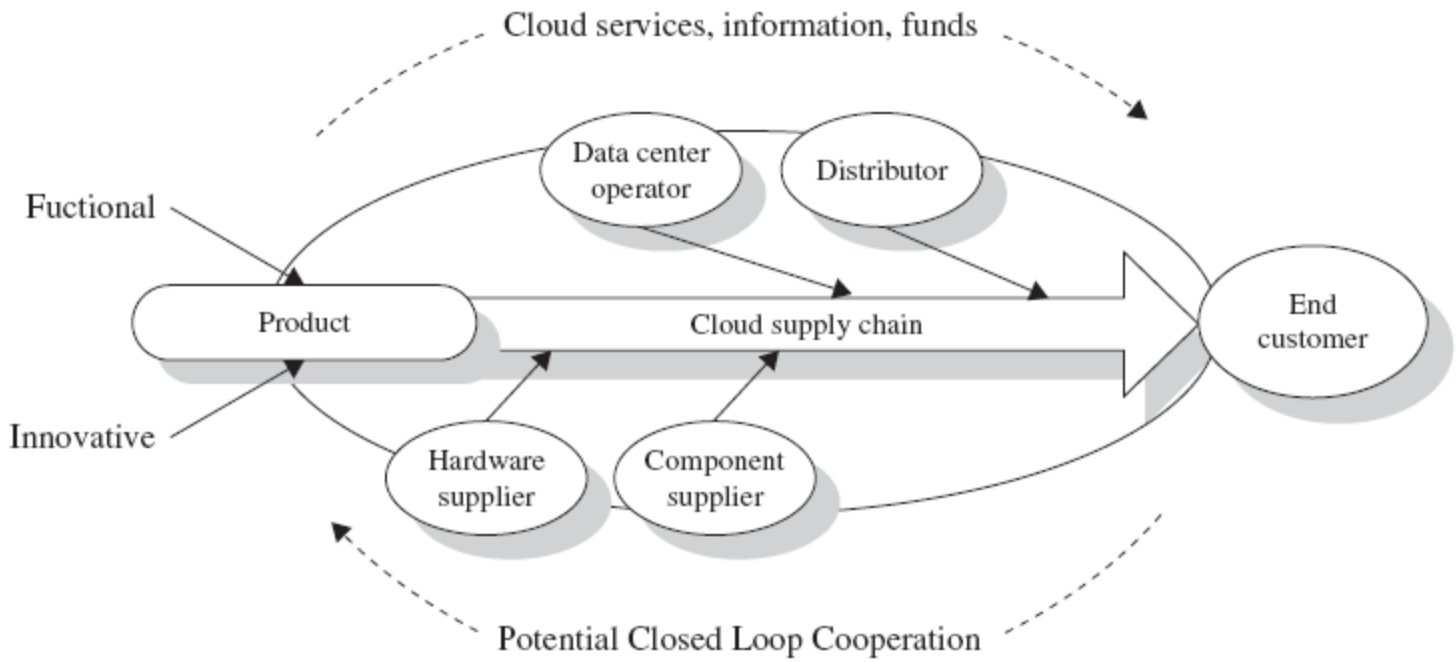
Marketplace

- industry-specific characteristics
  - Rivalry
  - Comparable Products
    - Market share
  - Federation
    - Small companies
  - Switching cost
    - Standardization

Dynamic Business Model



Cloud Supply Chain



ON THE MANAGEMENT OF VIRTUAL MACHINES FOR CLOUD INFRASTRUCTURES

IaaS Anatomy

- IaaS provider characteristic
  1. on-demand provisioning of computational resources
  2. Virtualization technologies to lease resources
  3. Provide public and simple remote interfaces to manage resources
  4. use a pay-as-you-go cost model
  5. “infinite capacity” or “unlimited elasticity”
- Private and Public difference
- Role of Virtualization
  - Key of these characteristic
  - Allocating resources efficiently
  - Taking into account an organization’s goals
  - Reacting to changes in the physical infrastructure

IaaS Anatomy

- Problems In VM Solutions
  - Distributed management of virtual machines
  - Reservation-based provisioning of virtualized resource
  - Provisioning to meet SLA commitments
- RESERVOIR project
  - Resources and Services Virtualization without Barriers
  - Addressed above problems

Distributed Management

- Manage the virtual infrastructures themselves
- Efficiently selecting or scheduling computational resources
- VM-based resource scheduling
  - Static approach
  - Efficiency approach

- Solution
  - Virtual Infrastructure Manager
    - Managing VMs in a pool of distributed physical resources
- Case Study
  - OpenNebula

VM Model and Life Cycle (OpenNebula)

- VM model attributes
  - A capacity in terms of memory and CPU
  - A set of NICs attached to one or more virtual networks
  - A set of disk images
  - A state file (optional) or recovery file

- Life Cycle
  - Resource Selection
  - Resource Preparation
    - Contextualization
  - VM Creation
  - VM Migration
  - VM Termination

VM Management
(OpenNebula)

- Management Areas
  - Virtualization
    - physical resource
  - Image management
  - Networking

Virtualization

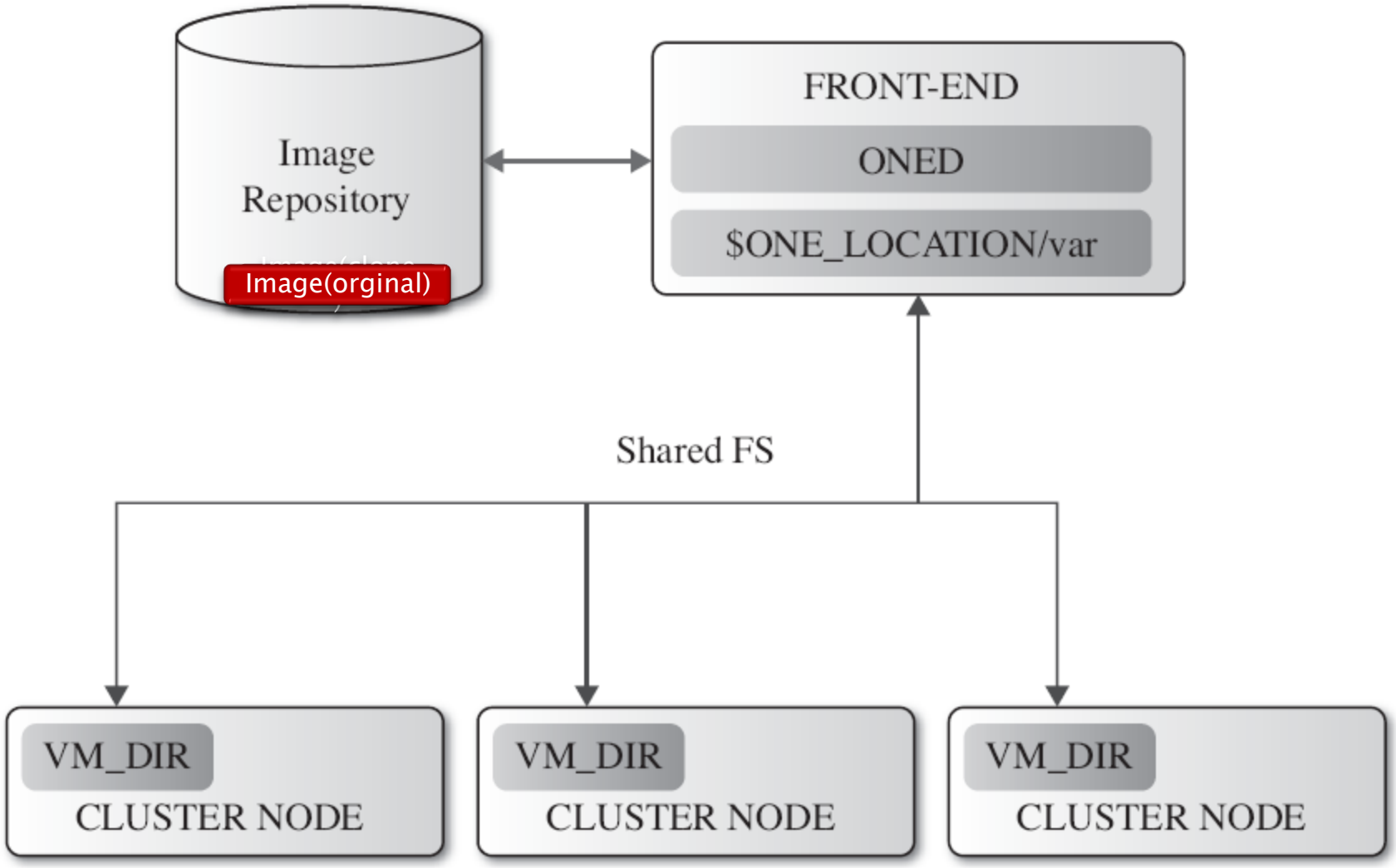
- How?
  - Interfacing with the physical resource virtualization technology (hypervisors like Xen, KVM)
- More detail
  - Pluggable drivers
    - Decouple the managing process from the underlying technology
  - High-level command
    - start VM, stop VM
  - Driver-based architecture
    - Adding support VIMs by writing drivers

Image Management

- How?
  - Transferring the VM images from an image repository to the selected resource and by creating on-the-fly temporary images
- More detail
  - What is image?
    - Virtual disk contains the OS and other additional software
  - Image management model

Image Management Model

(openNebula)

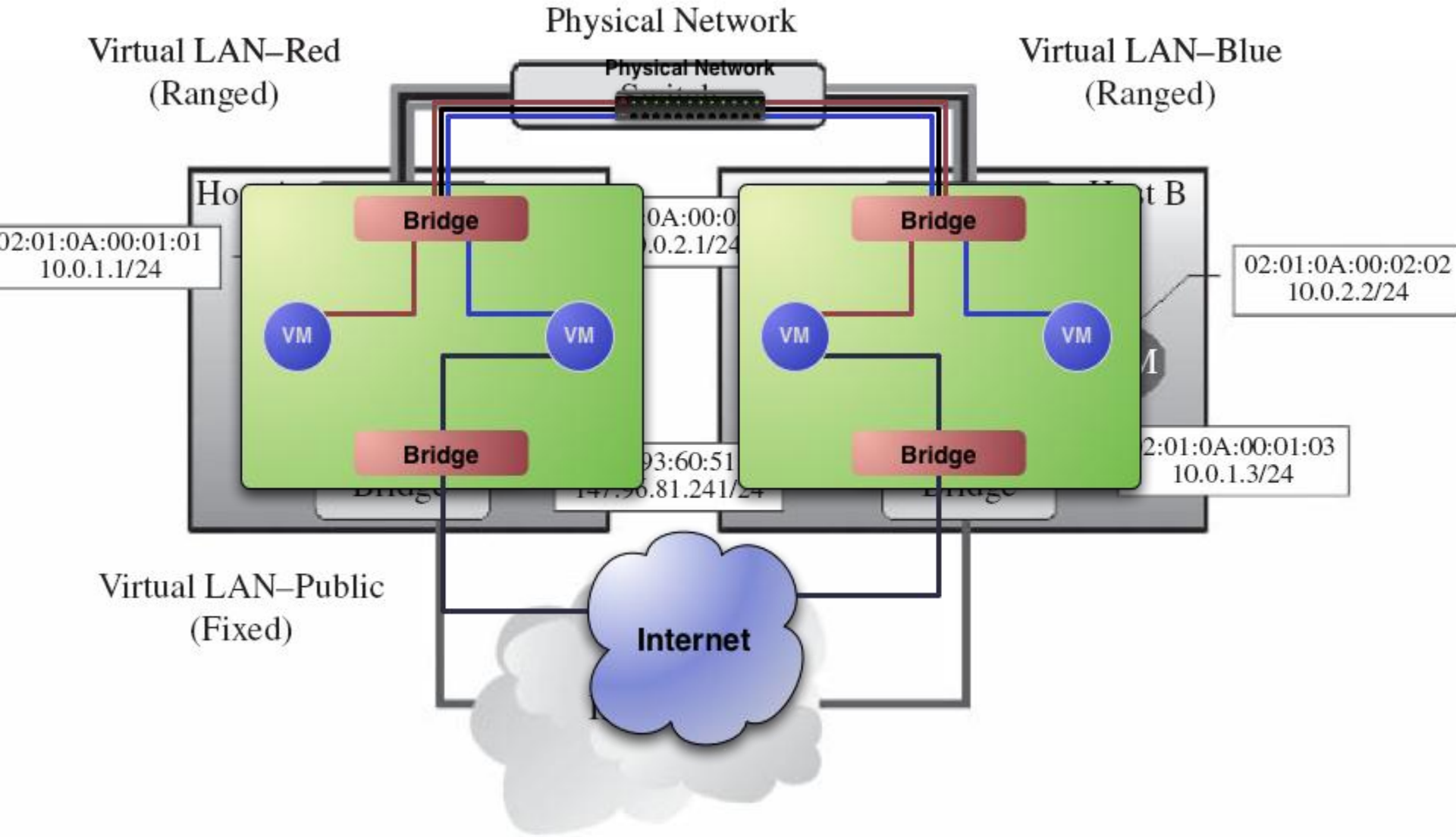


Networking

- How?
  - creating local area networks (LAN) to interconnect the VMs and tracking the MAC addresses leased in each network.

- More detail
  - virtual application network (VAN)
    - the primary link between VMs
  - OpenNebula dynamically creates VANs
  - physical cluster
    - set of hosts with one or more network interfaces
    - each of them connected to a different physical network
  - Networking Model

Networking Model (OpenNebula)



Advance Reservation

- Demand for resources is known beforehand
- Example
  - an experiment depending on some complex piece of equipment is going to run from 2 pm to 4 pm
- Commercial Providers
  - Infinite capacity
- Private clouds
  - Finite capacity
  - Reservation lead resource to be underutilized
- Haizea
  - Lease manager
  - Scheduling backend by openNebula to support provisioning models

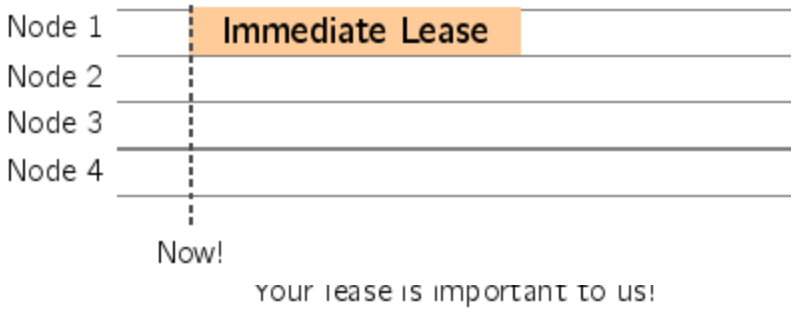
Existing Approach

- Preemption
  - Checkpointing
  - Checkpointable applications
  - OS-level checkpointing
- VARQ
  - Virtual advance reservation for queues
  - Queuing based approach
  - Wait time prediction
- Planning based approach
  - Immediately planned by making a reservation

Reservation with VMs

- Challenges
  - Preparation overhead
  - Runtime overhead

- Haizea
  - Leases
    - Advance reservation
    - Best-effort
    - Immediate



Haizea Lease Scheduling

- Backfilling
- How to address preparation and runtime Overhead?
  - Disk image transfer before start
  - Caching
- How does best-effort lease?
  - Scheduling using queue
  - Backfilling algorithm
  - Depend on required disk image
- VM suspension/resumption
- How does advance reservation lease?
  - EDF algorithm for preparation overhead
  - Without preemption for Runtime overhead
- Pluggable policy
- Combine best-effort and advance reservation
  - Overcome utilization problems

SLA Commitment

- Cloud consumer vs. End users
- SLA between Service owner and end user
  - High-Level SLA
- SLA between Cloud provider and Service owner
- Cloud provider task
  - Elasticity on demand
- Problem
  - Application specific metric for resource allocation
- Solution
  - Elasticity of the application should be contracted and formalized as part of capacity availability SLA between the cloud provider and service owner (RESERVOIR)
- Research issues

Infrastructure SLAs

- Main approaches:
  - No SLAs
    - Premises
      - Spare capacity
      - QoS-insensitive
    - Suitable for best-effort workloads
  - Probabilistic SLAs
    - Availability percentile
    - Less stringent commitment
    - Lower availability = cheaper cost
    - Suitable Small and medium business
  - Deterministic SLAs
    - 100% availability percentile
    - Most stringent guarantee
    - Suitable for Critical services

Elasticity rules

- Definition
  - Scaling and de-scaling policies
- Motivation
  - Pay-as-you-go billing
- Types
  - Time driven
    - Timer event
    - Predictable workload
  - OS Level Metric driven
    - OS parameter, auto scaling
    - Not precise
  - Application Metric driven
    - Application specific policies

Policy-Driven Probabilistic Admission Control

- ARL (Acceptable Risk Level)
  - Control over-subscribing of capacity
- BSM-aligned admission control
- Equivalent Capacity
  - resource of the service applications
  - Representation $(r_1, r_2, \ldots r_n) \rightarrow (10, 13, \ldots 4)$
  - Physical capacity matching
    - Knapsack problem (no capacity augmentation)
    - Bin-packing problem (with capacity augmentation)
  - We have abstract equivalent capacity
  - Rejection policy
    - Reject service
    - Increase capacity and accept service
    - Increase ARL and accept service
  - Service providers influence

Policy-Driven Placement Optimization

- Aspects
  - Penalization for Nonplacement
    - Penalty for SLA violation
  - Selection Constraints
    - No Partial placement
  - Repeated Solution
    - Minimize the cost of replacement
    - Minimize the cost of reassignments of VMs to hosts
    - Reassignment entail migration
  - ICT-Level Management Policies
    - power conservation
    - load balancing
    - migration minimization

Management Policies and Management Goals

- Policy-driven Management
  - if-then rules
  - Management goals
- Placement optimization
  - Phase I : low effort placement
  - Phase II : management policy
- Levels of management(abstraction)
  - Business
  - Service-induced
  - Infrastructure (ICT level)

Enhancing Cloud Computing Environments Using a Cluster as a Service

Introduction

- Elements to create cloud
  - Large-scale clusters
  - Virtualization
  - Service Oriented Architecture (SOA)
  - Web Services
- CaaS
  - Cluster as a Service
- Related Work
  - Amazon EC2
  - Google AppEngine
  - Microsoft Azure
  - Salesforce

RVWS Design

- Problem
  - To know if the resource(s) behind the Web service is (are) ready
- Solution
  - Resource Via Web Service (RVWS) framework
  - A single, effective, service-based framework
  - Combines
    - dynamic attributes
    - stateful Web services (aware of their past activity)
    - stateful and dynamic WSDL documents
    - brokering

Dynamic attributes

- State attributes
  - cover the current activity of the service and its resources, thus indicating readiness.

- Characteristic attributes
  - cover the operational features of the service, the resources behind it, the quality of service (QoS), price and provider information

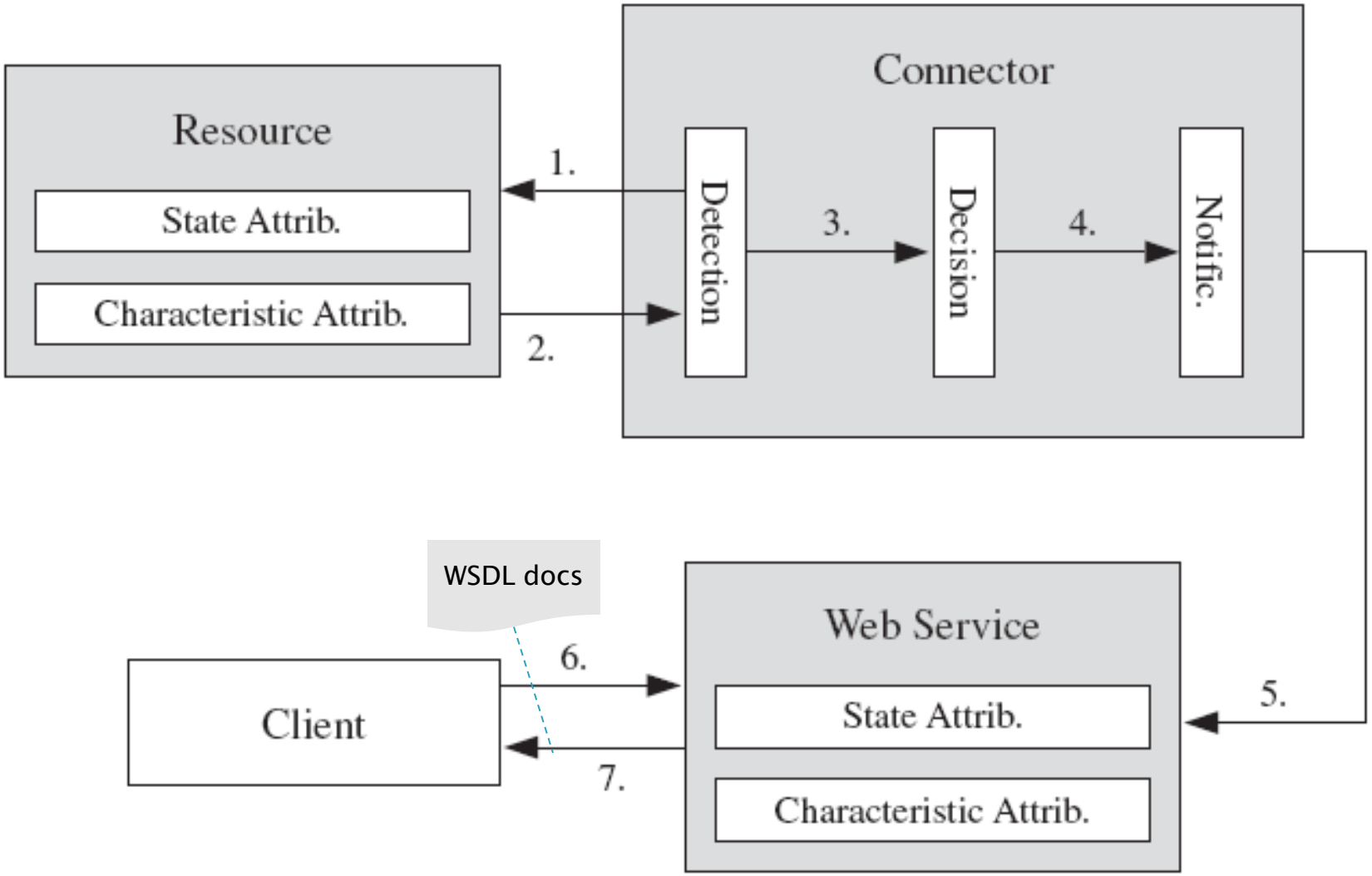
Cluster (state) Attributes

Type	Attribute Name	Attribute Description	Source
State	free-disk	Amount of free disk space	Cluster node
	free-memory	Amount of free memory	
	os-name	Name of the installed operating system	
	os-version	Version of the running operating system	
	processes-count	Number of processes	
	processes-running	Number of processes running	
	cpu-usage-percent	Overall percent of CPU used. As this metric is for the node itself, this value becomes averaged over cluster core	Generated
	memory-free-percent	Amount of free memory on the cluster node	

Cluster (characteristic) Attributes

Type	Attribute Name	Attribute Description	Source
Characteristics	core-count	Number of cores on a cluster node	Cluster node
	core-speed	Speed of each core	
	core-speed-unit	Unit for the core speed (e.g., gigahertz)	
	hardware-architecture	Hardware architecture of each cluster node (e.g., 32-bit Intel)	
	total-disk	Total amount of physical storage space	
	total-disk-unit	Storage amount unit (e.g., gigabytes)	
	total-memory	Total amount of physical memory	
	total-memory-unit	Memory amount measurement (e.g., gigabytes)	
	software-name	Name of an installed piece of software.	
	software-version	Version of a installed piece of software	
	software-architecture	Architecture of a installed piece of software	

Stateful Web Service

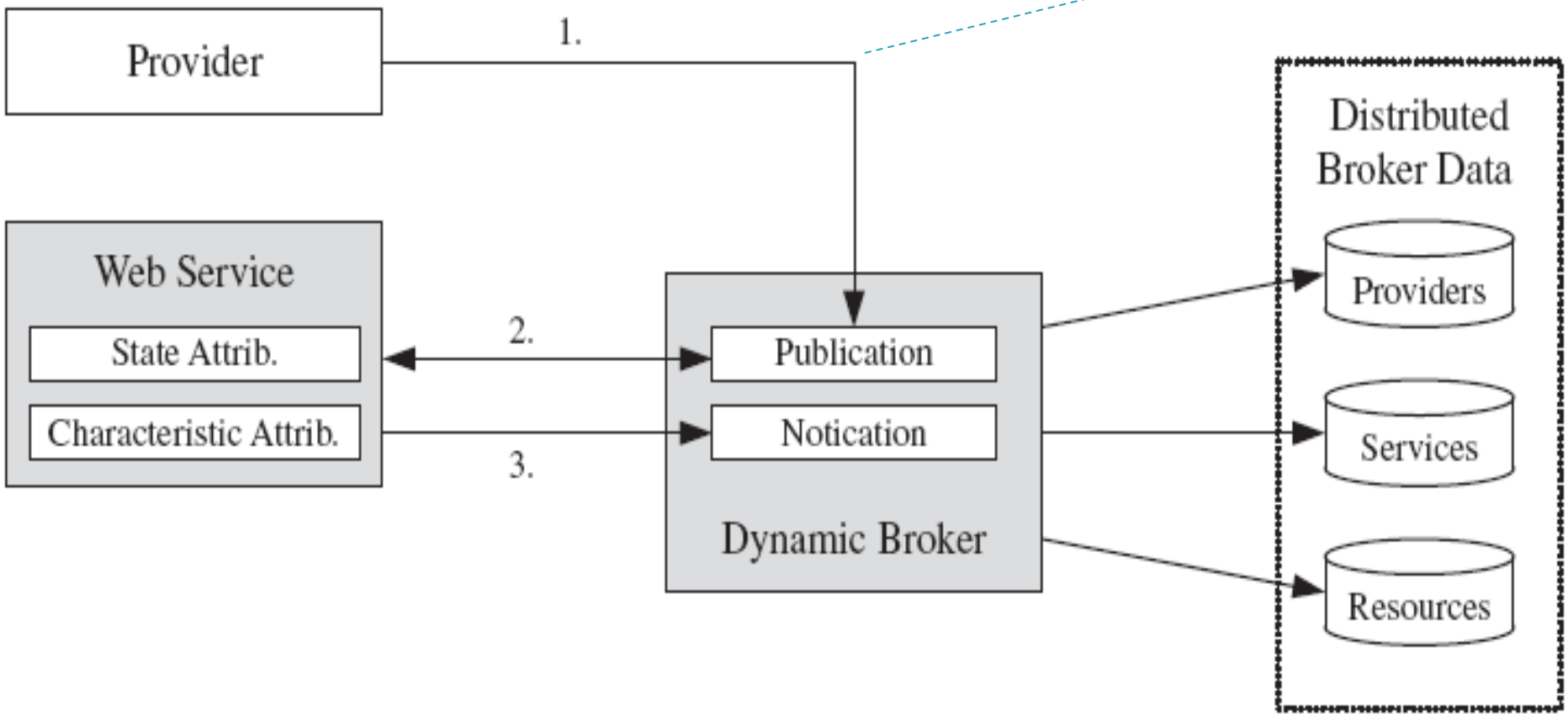


Stateful WSDL Document

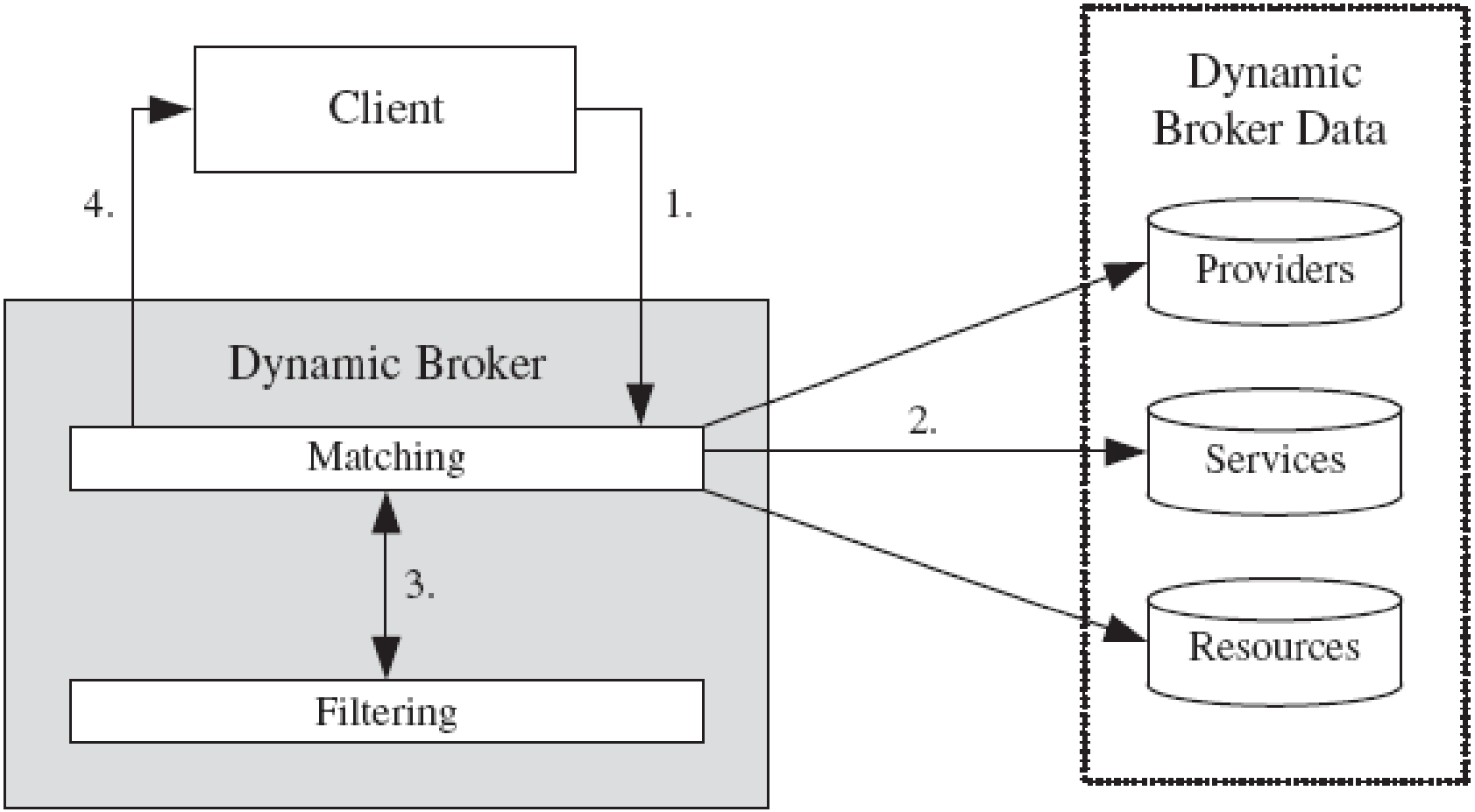
```
<definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <resources>
      <resource-info identifier="resourceID">
          <state>
              <description name="" attribute1="value1" ...
                            attributen="valuen">
                  ...Other description Elements...
              </description>
              ...Other description Elements...
          </state>
          <characteristics>
              <description name="" />
              ...Other description Elements...
          </characteristics>
      </resource-info>
      ...Other resource-info elements
  </resources>
  <types>...</types>
   message name="MethodSoapIn">...</message>
  <message name="MethodSoapOut">...</message>
  <portType name="CounterServiceSoap">...</portType>
  <binding name="CounterServiceSoap"
           type="tns:CounterServiceSoap">...</wsdl:binding>
  <wsdl:service name="CounterService">...</wsdl:service>
```

Dynamic Broker
(Publication in RVWS)

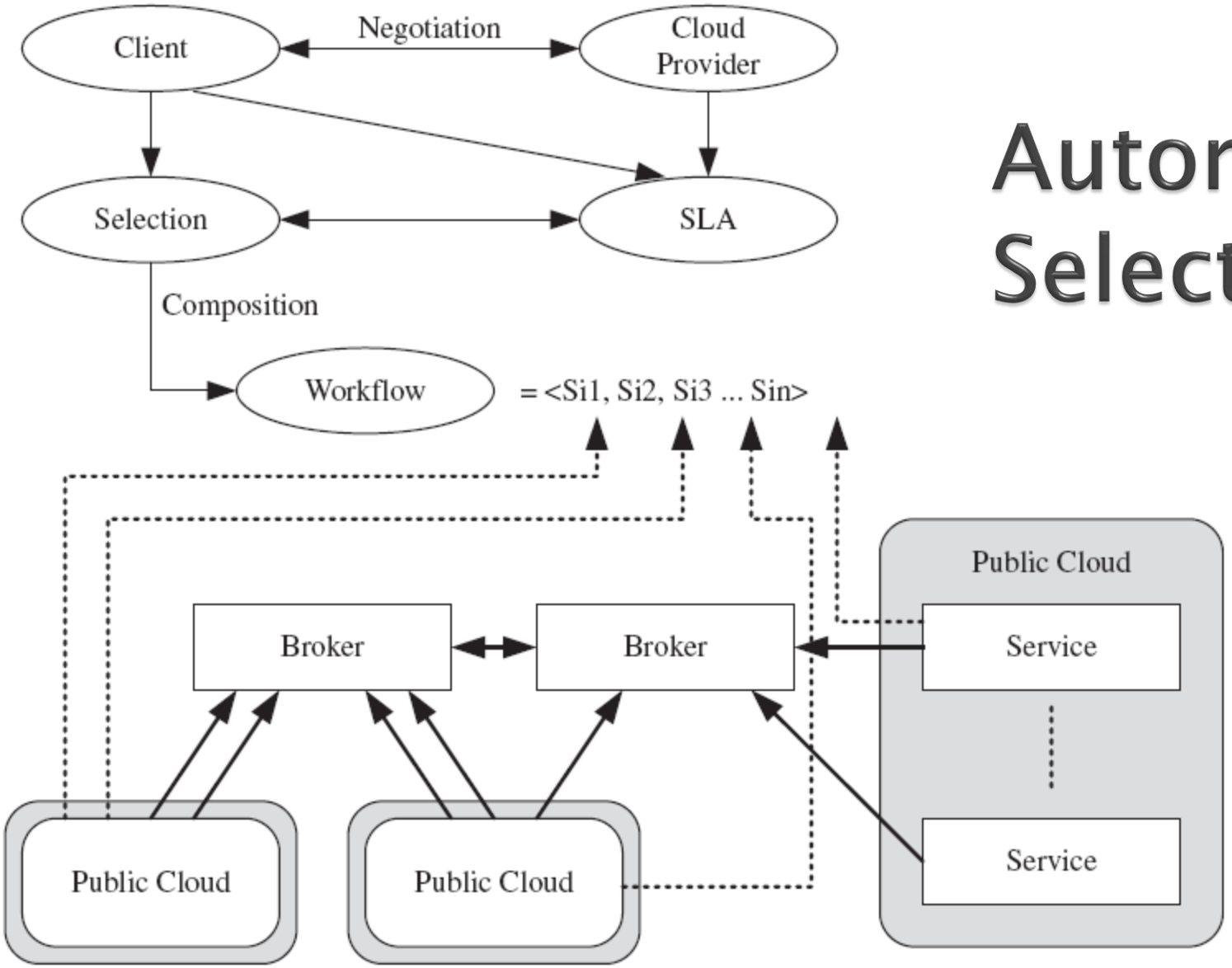
Publication dynamic attributes and provider info..



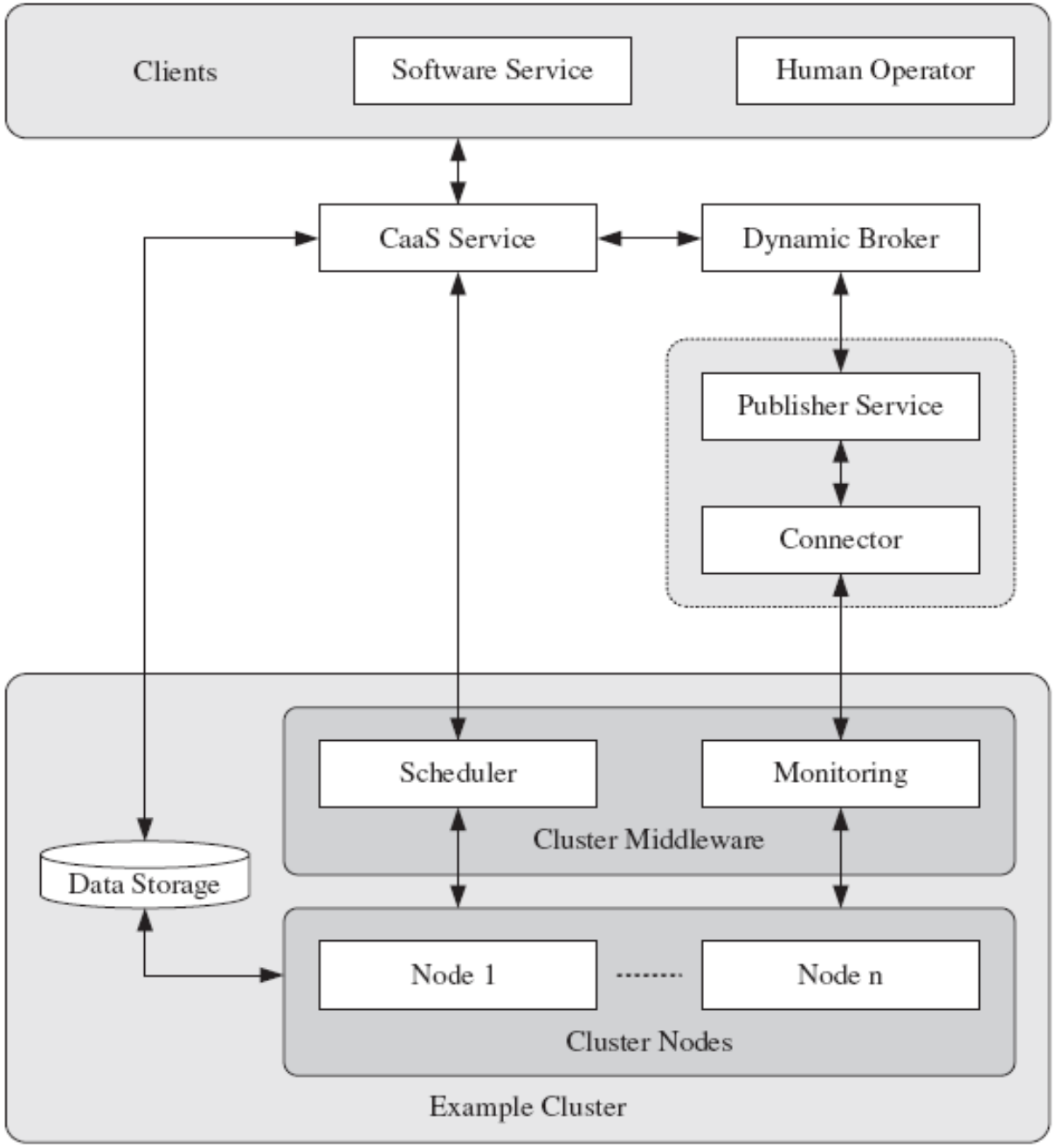
Automatic Discovery



Automatic Selection



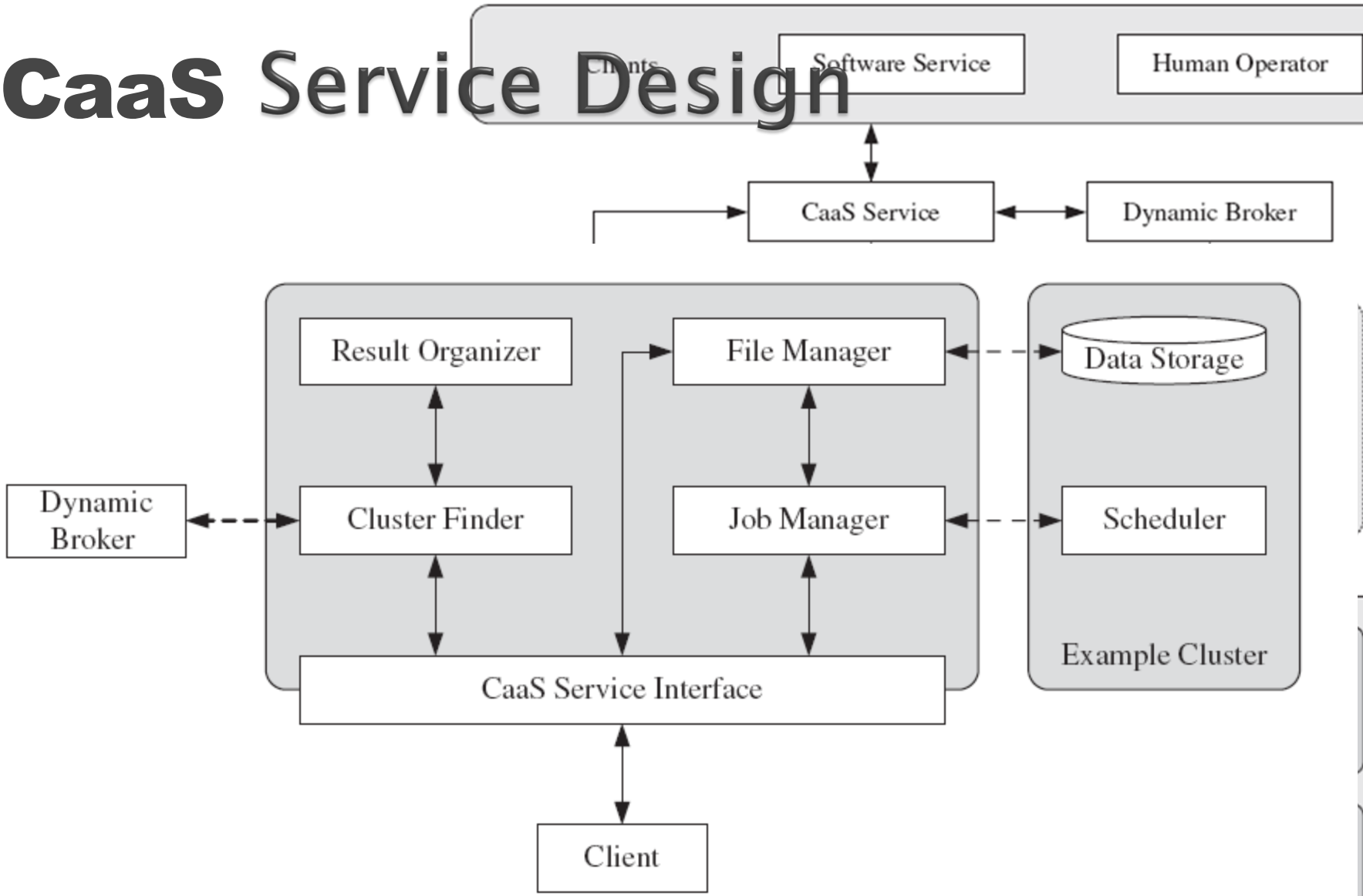
CaaS Overview



Stateful Cluster WSDL

```
<definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <resources>
    <resource-info resource-identifier="resourceId">
      <state element-identifier="elementId">
        <cluster-state element-identifier="cluster-state-root">
          <cluster-node-name free-disk="" free-memory="" os-name="" os-version=""
            os-virtualisation="" processes-count="" processes-running=""
            cpu-usage-percent="" memory-free-percent=""
            element-identifier="stateElementId" />
          ...Other Cluster Node State Elements...
        </cluster-state>
      </state>
      <characteristics element-identifier="characteristicElementId">
        <cluster-characteristics node-count=""
          element-identifier="cluster-characteristics-root">
          <cluster-node-name core-count="" core-speed="" core-speed-unit=""
            hardware-architecture="" total-disk="" total-disk-unit=""
            total-memory="" total-memory-unit=""
            element-identifier="characteristicElementId" />
          ...Other Cluster Node Characteristic Elements...
        </cluster-characteristics>
      </characteristics>
    </resource-info>
  </resources>
  <types>...</types>
  <message name="MethodSoapIn">...</message>
  <message name="MethodSoapOut">...</message>
  <portType name="PublisherServiceSoap">...</portType>
  <binding name="PublisherServiceSoap"
           type="tns:PublisherServiceSoap">...</binding>
  <service name="PublisherService">...</service>
</definitions>
```

CaaS Service Design



Cluster Specification

Section A: Hardware

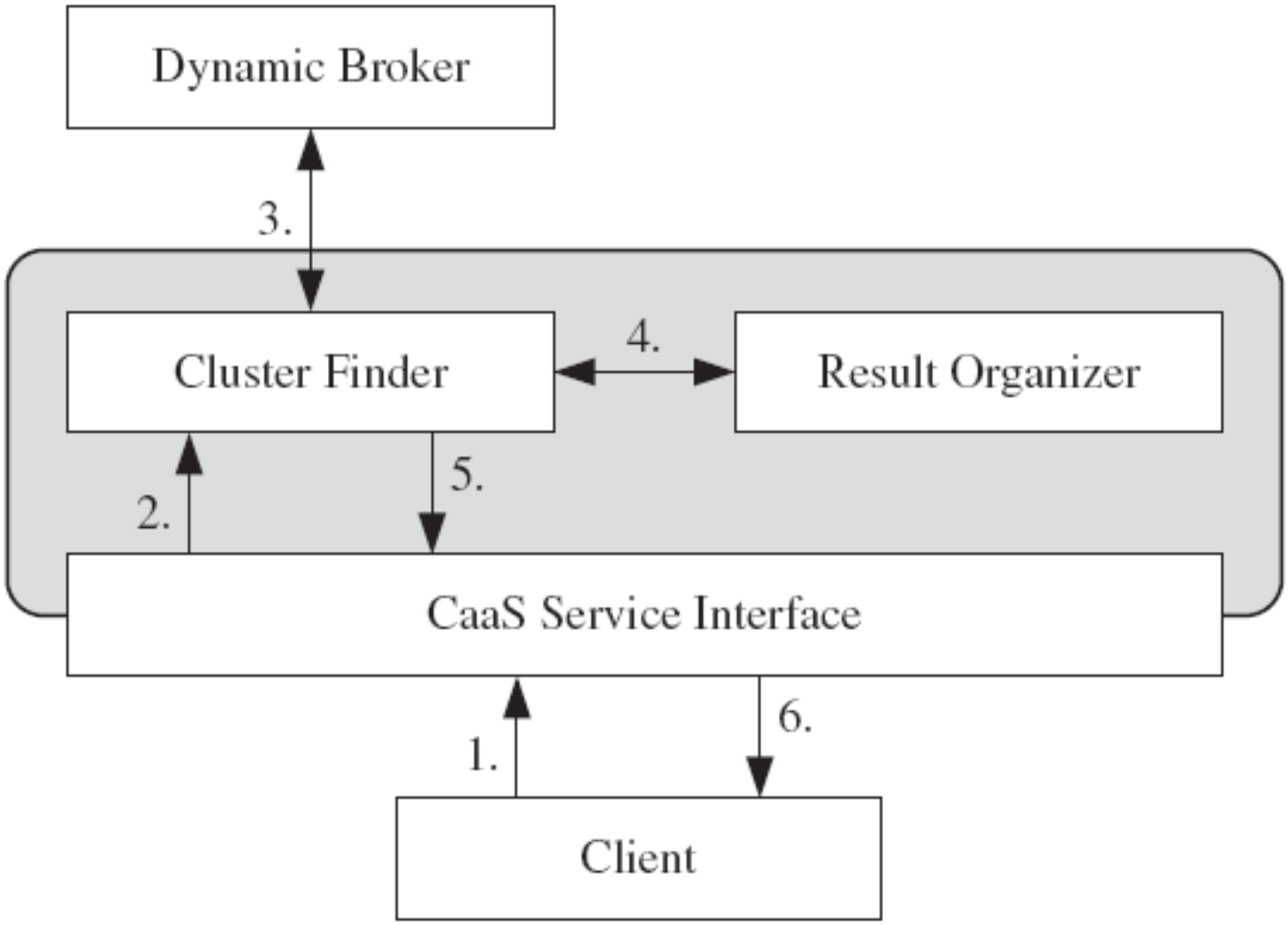
Number of Nodes:	50			
Amount of Memory:	50	GB		
Free Memory:	50	GB		
Disk Free:	50	GB		
CPU:	Pentium 4	64 bit	3.2	GHz

Section B: Software

Operating System:	Windows XP w/Service Pack 2

Discover ->

Cluster Discovery

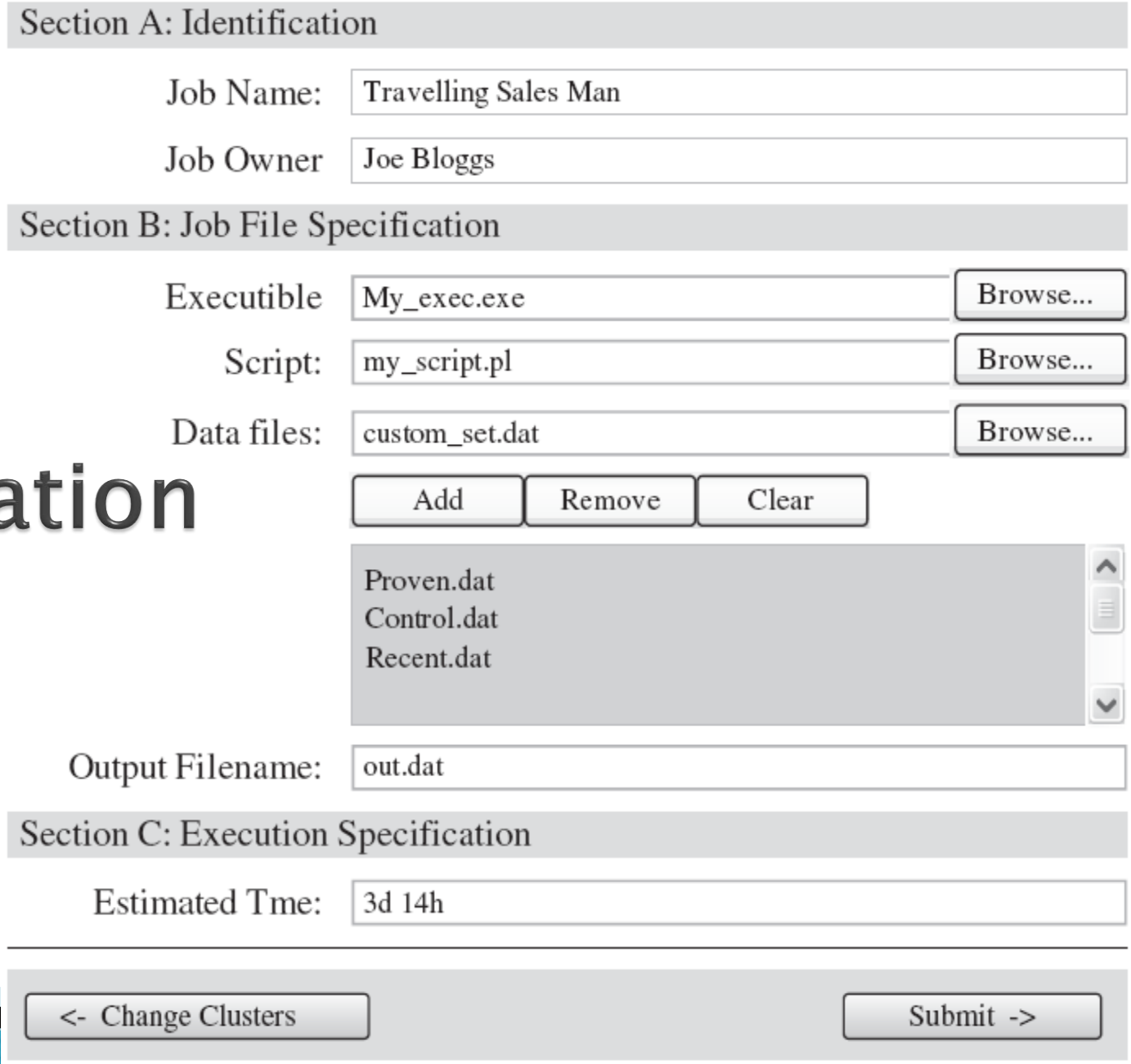


Cluster Selection

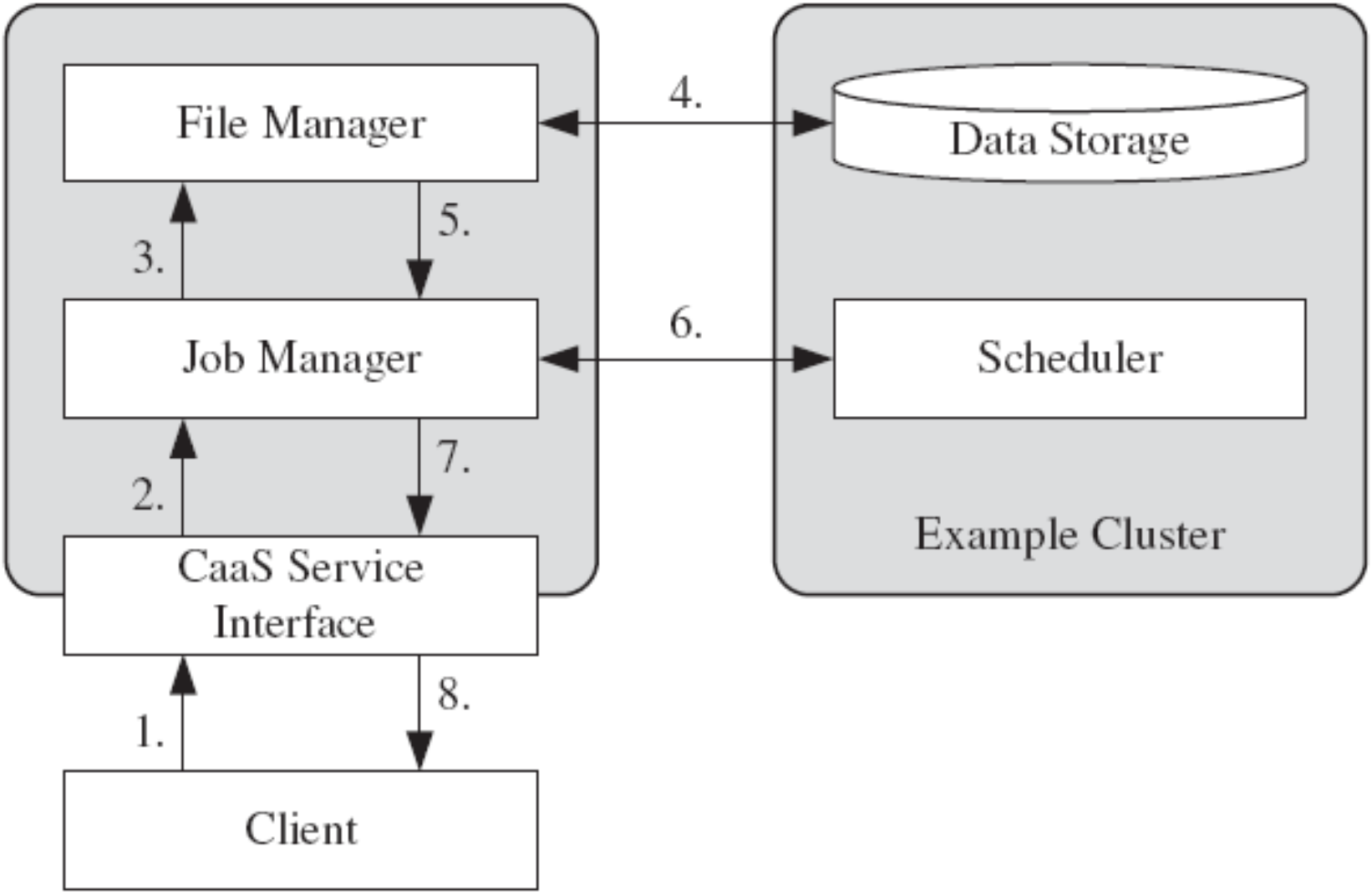
	Cluster A select	Cluster B select
Hardware		
Number of Nodes :	☑	
Amount of Memory :	☑	
Free Memory :	☑	
Disk Free :		☑
CPU :	☑	
Architecture :	☑	
Speed		☑
Software		
Operating System :	☑	
Architecture :	☑	
Version :	☑	

<- Refine Search

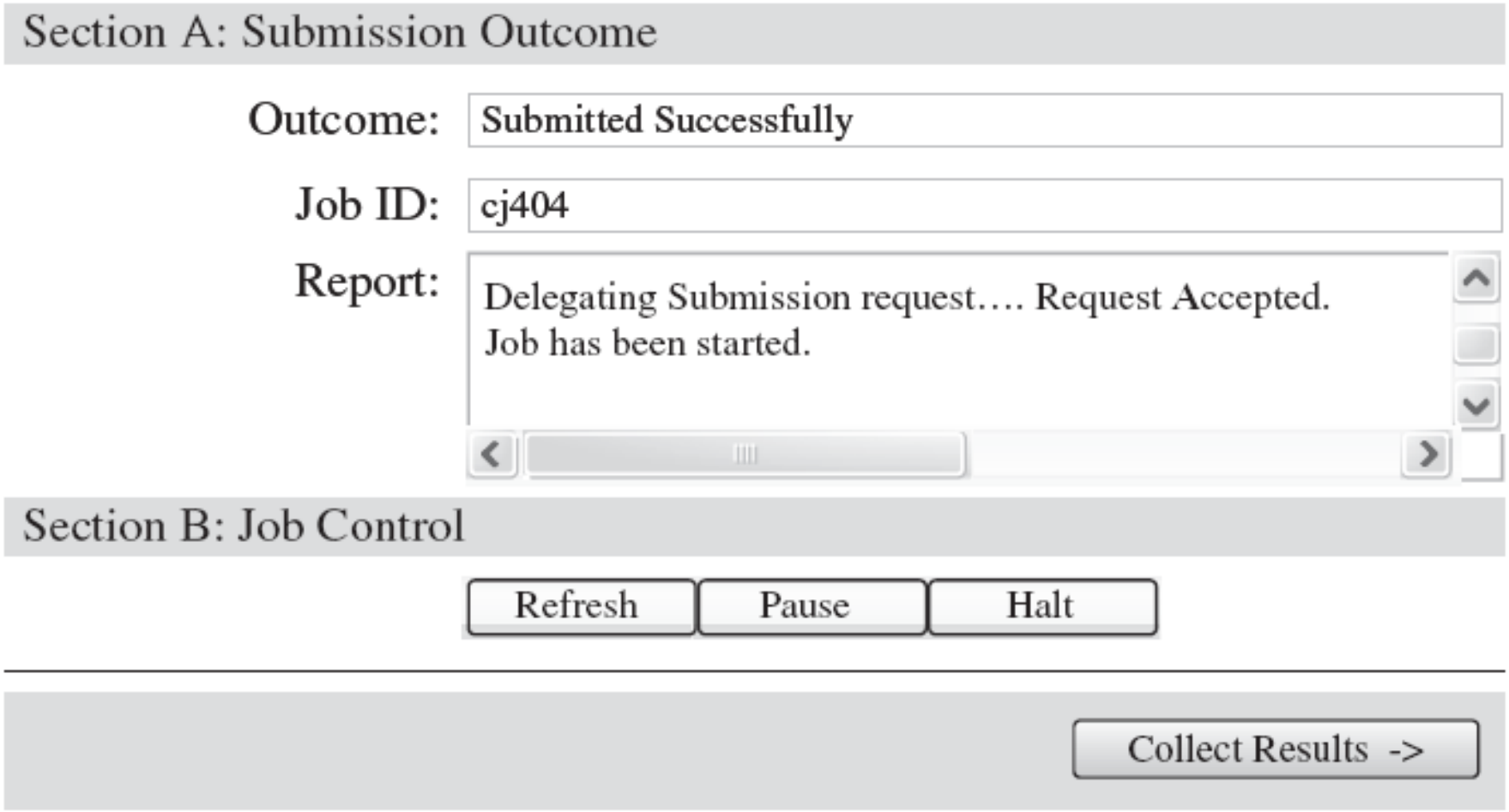
Job Specification



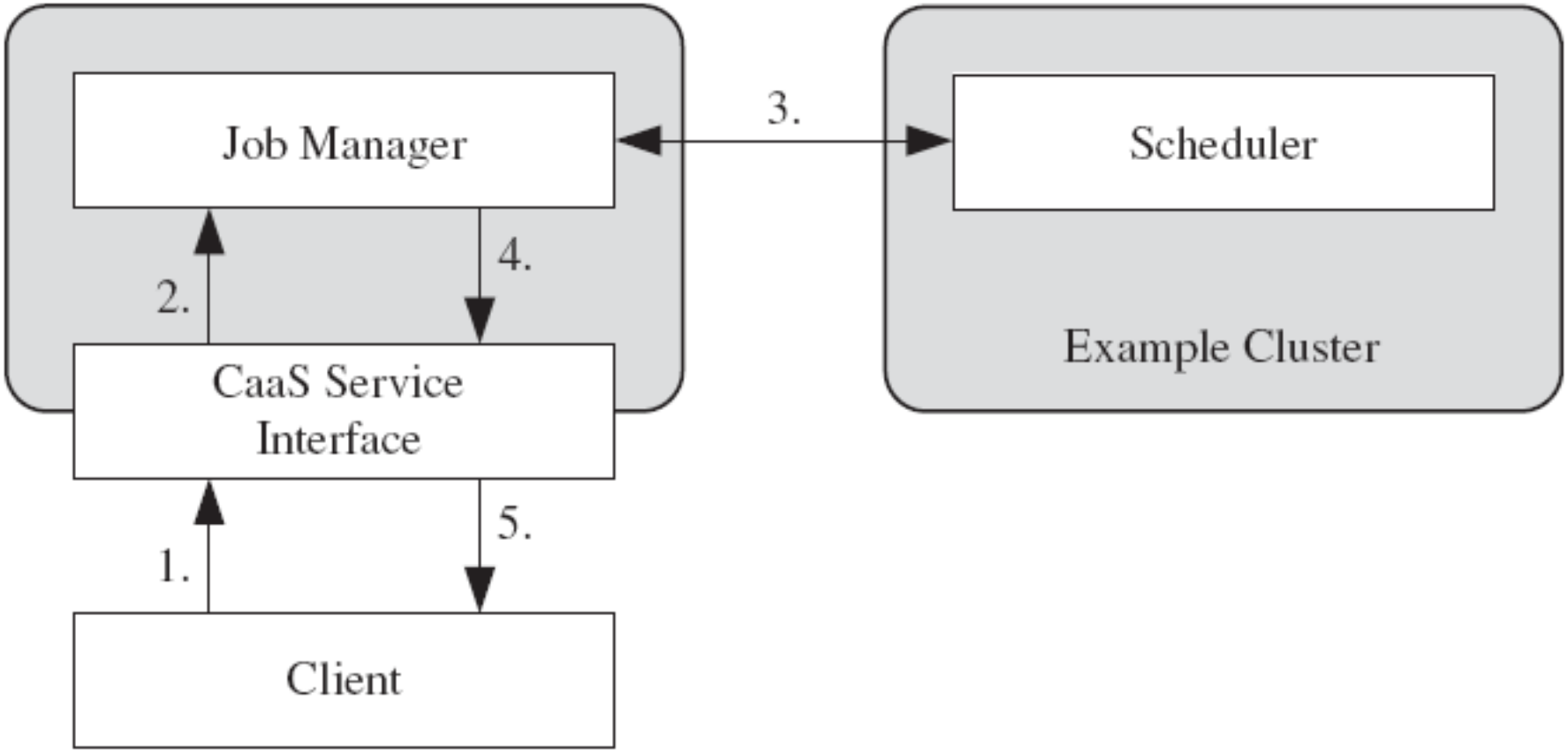
Job Submission



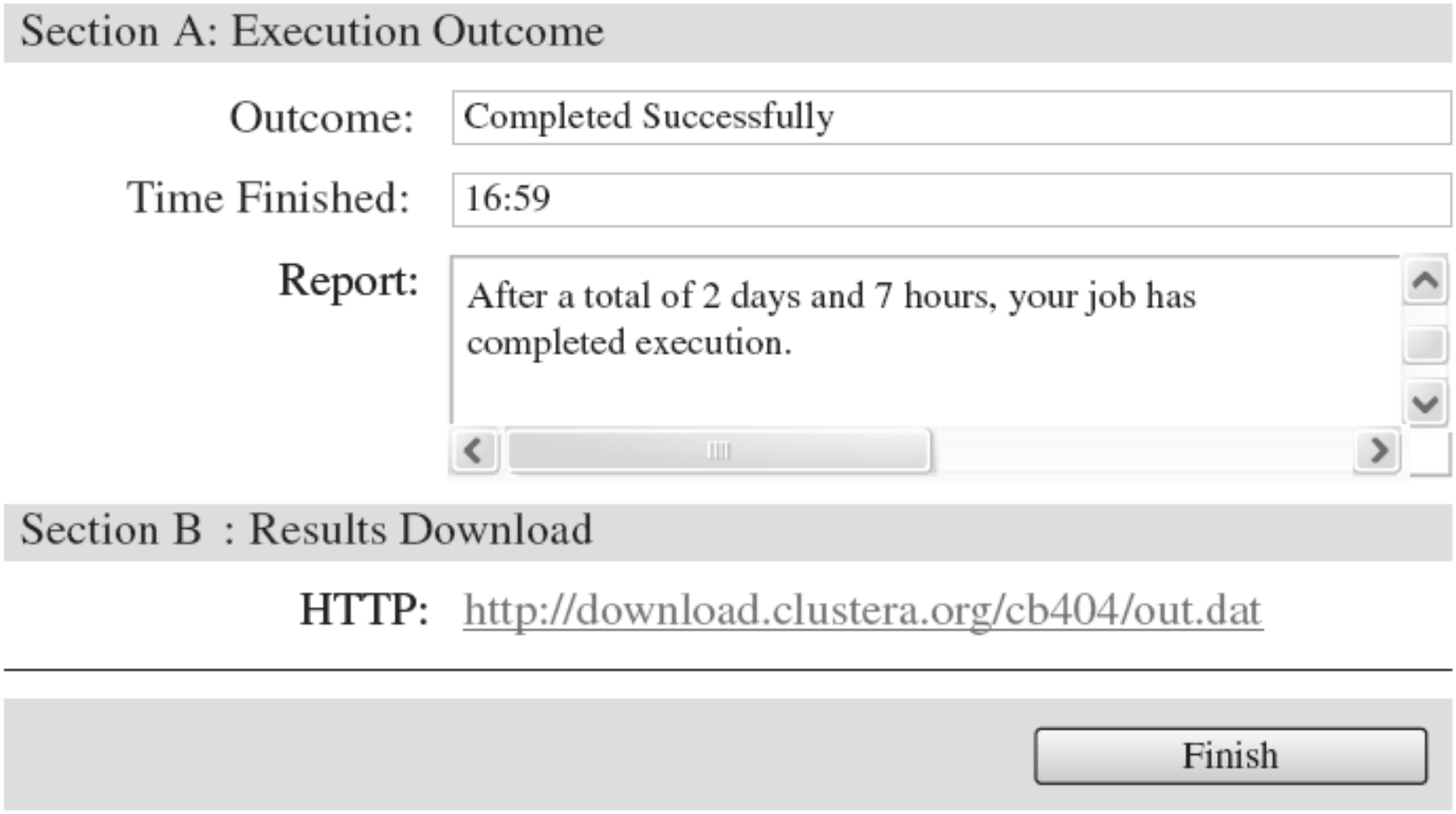
Job Monitoring



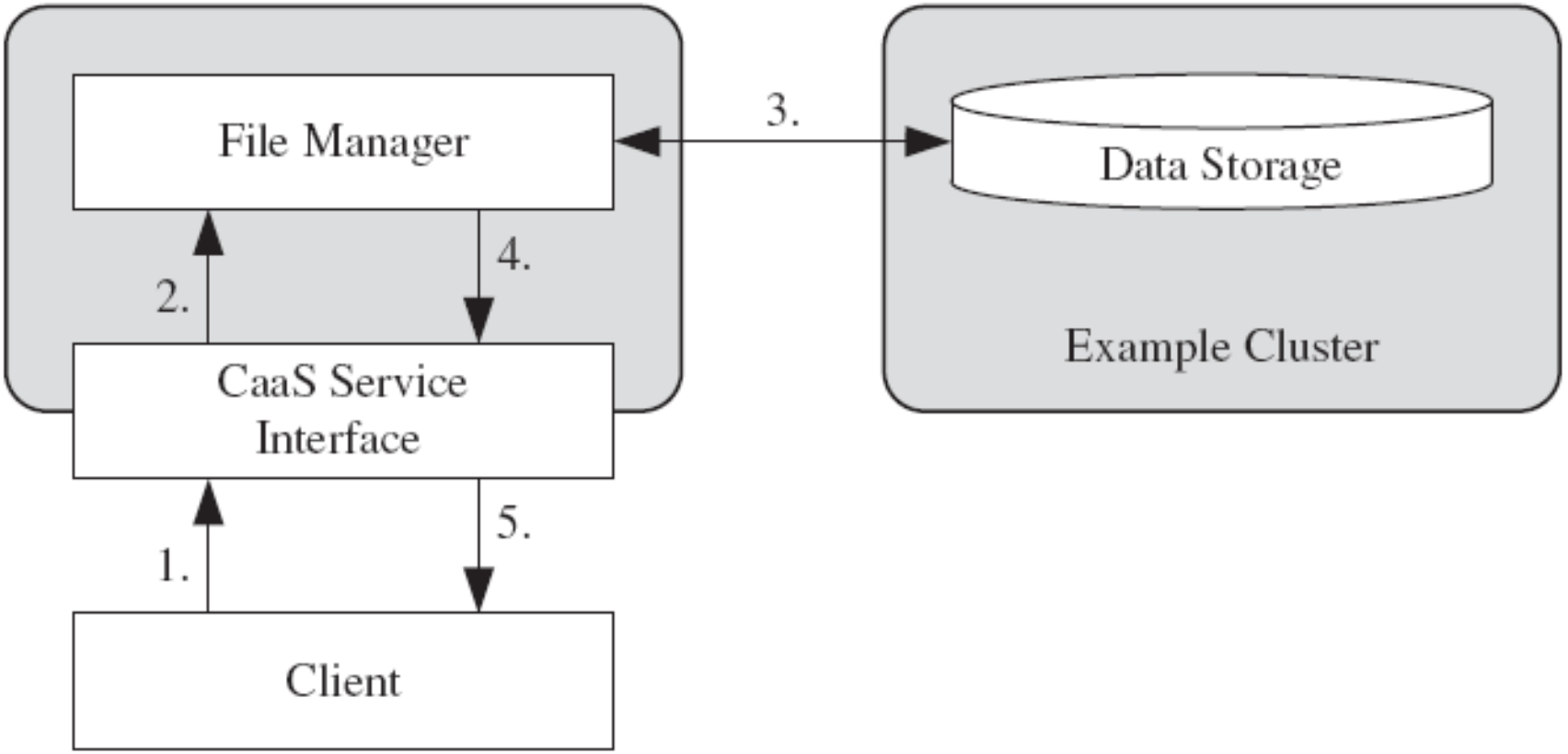
Job Monitoring



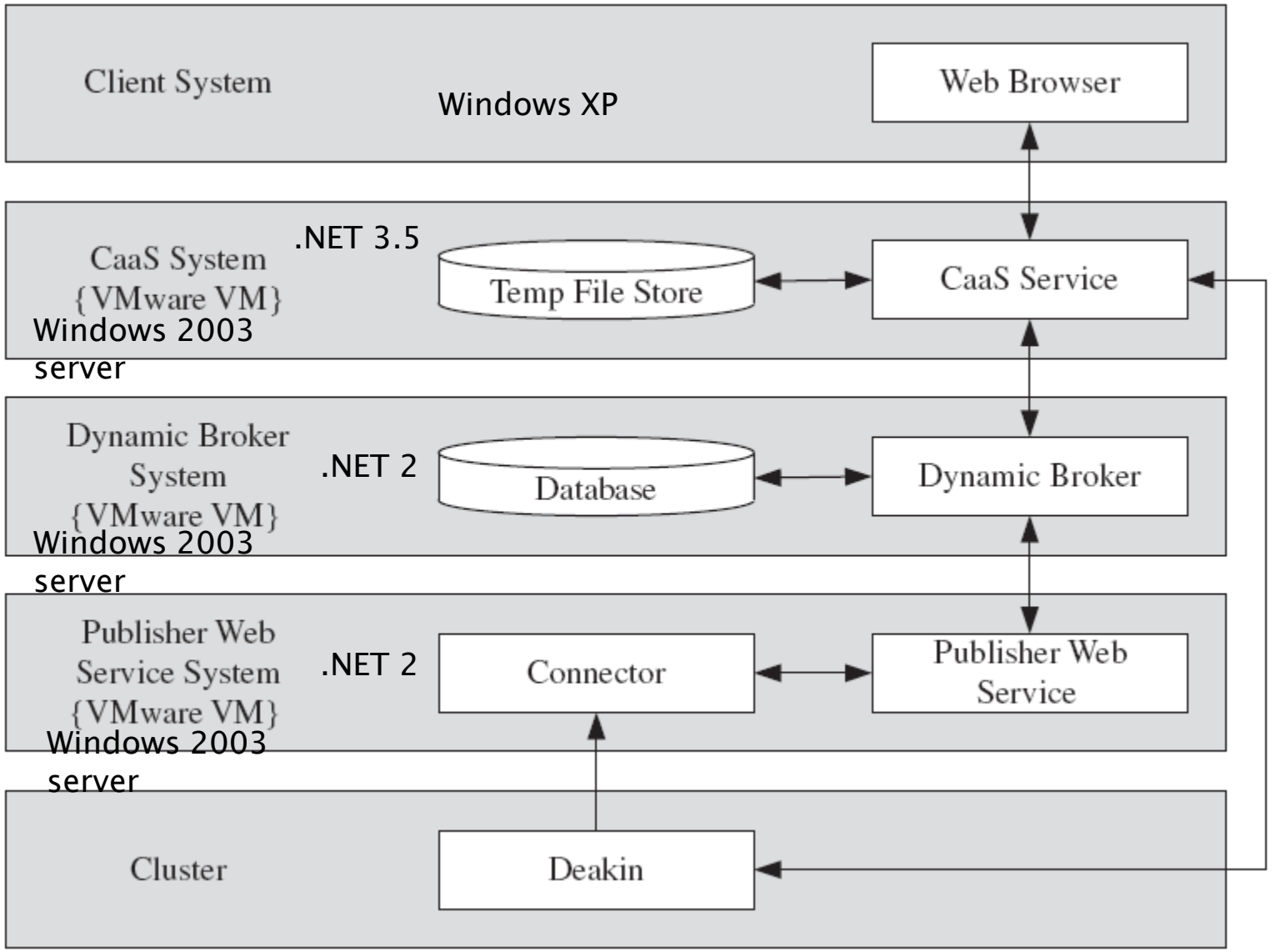
Result Collection



Result Collection



CaaS Environment



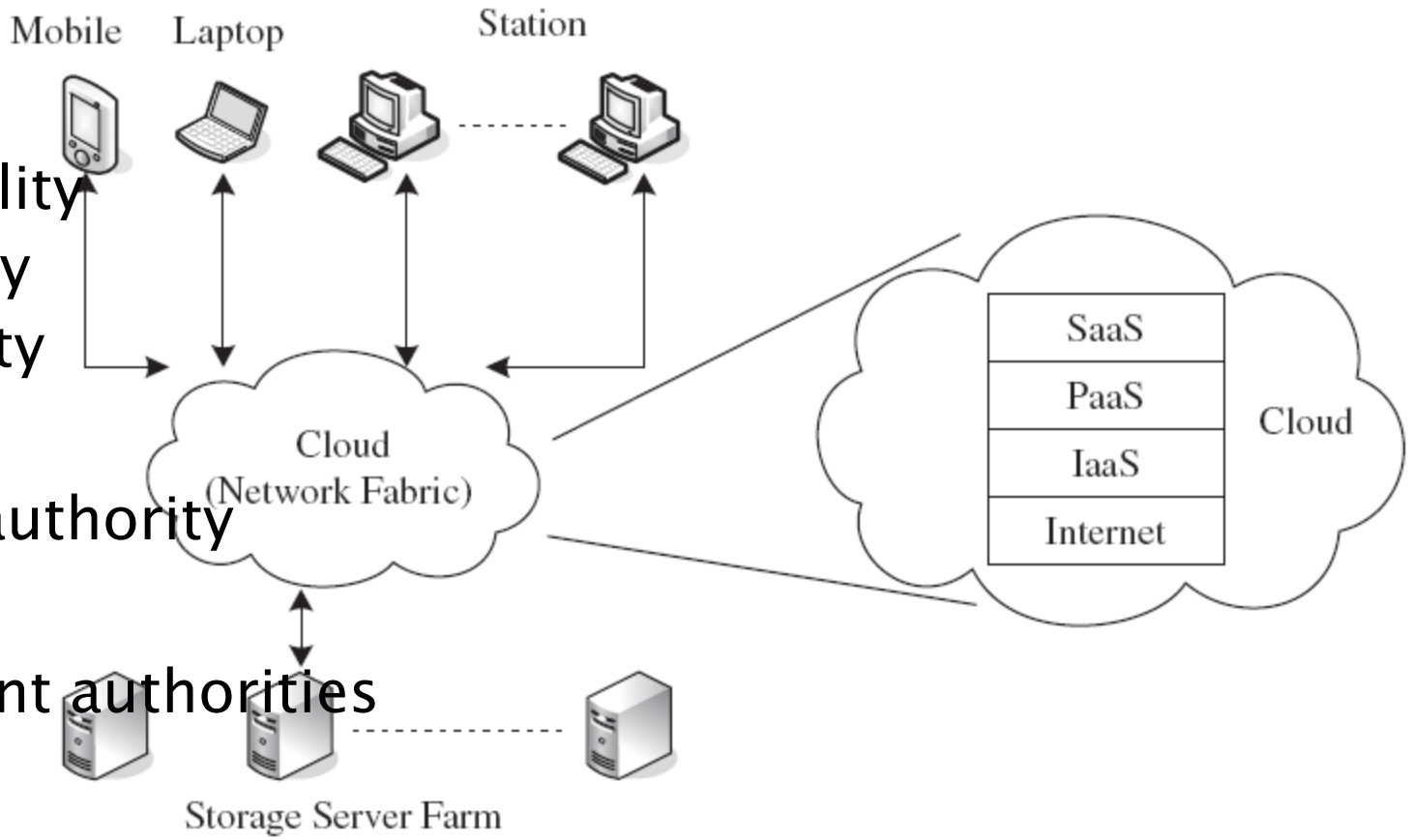
SECURE DISTRIBUTED DATA STORAGE IN CLOUD COMPUTING

Introduction

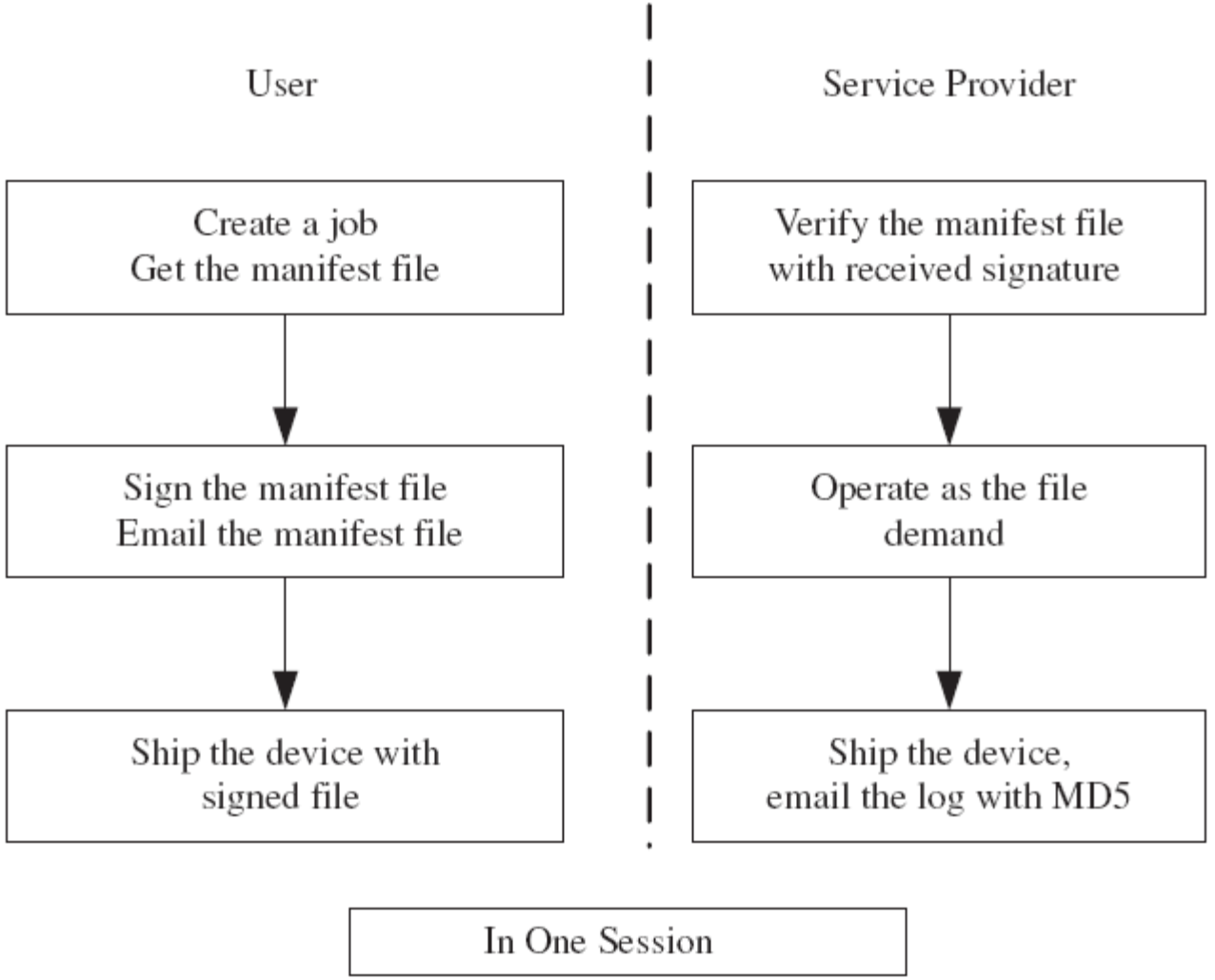
- Data Storage
- Distributed Storage
- Considerations
  - unique issues
  - specific security requirements not been well-defined
- Concerns about data in cloud
  - Privacy
  - Integrity

Cloud Storage

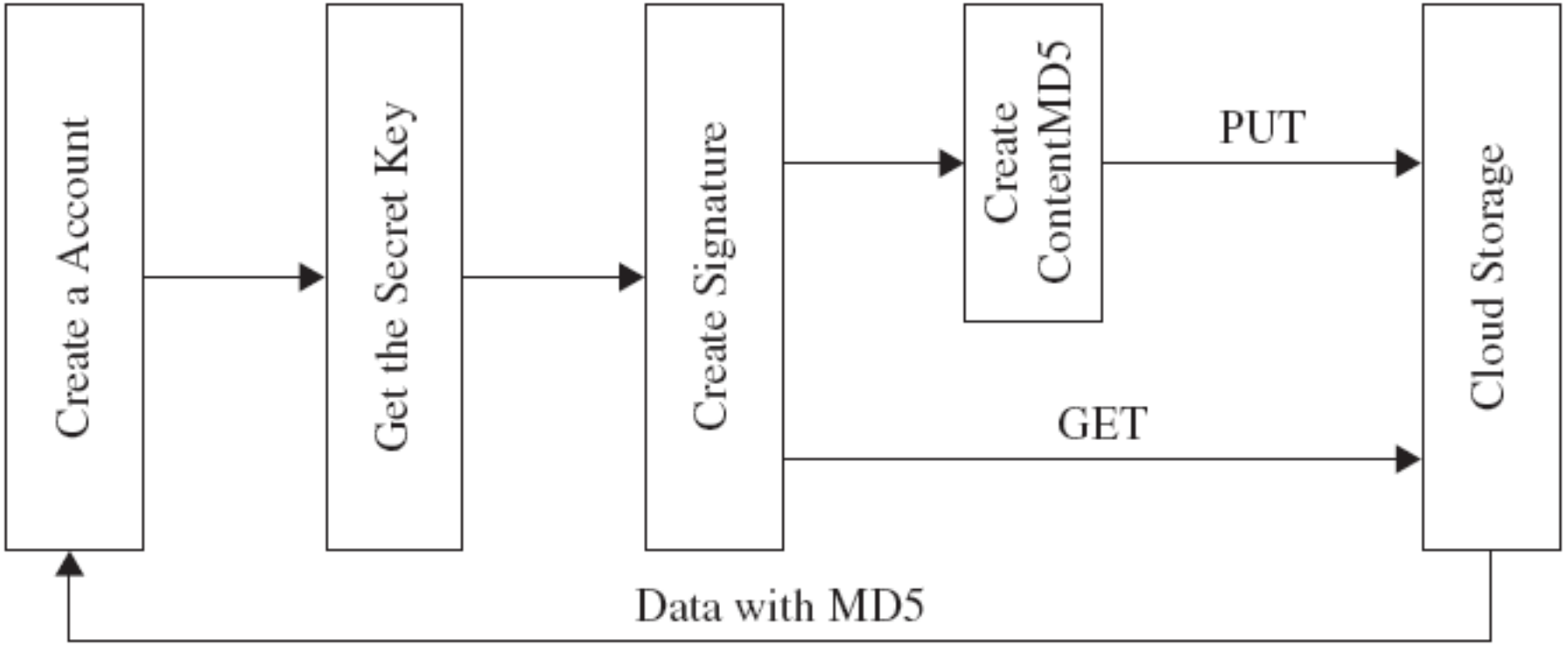
- Distributed Storage
  - Types
    - SAN
    - NAS
  - Reliability
  - Security
  - Integrity
- LAN
  - same authority
- WAN
  - different authorities



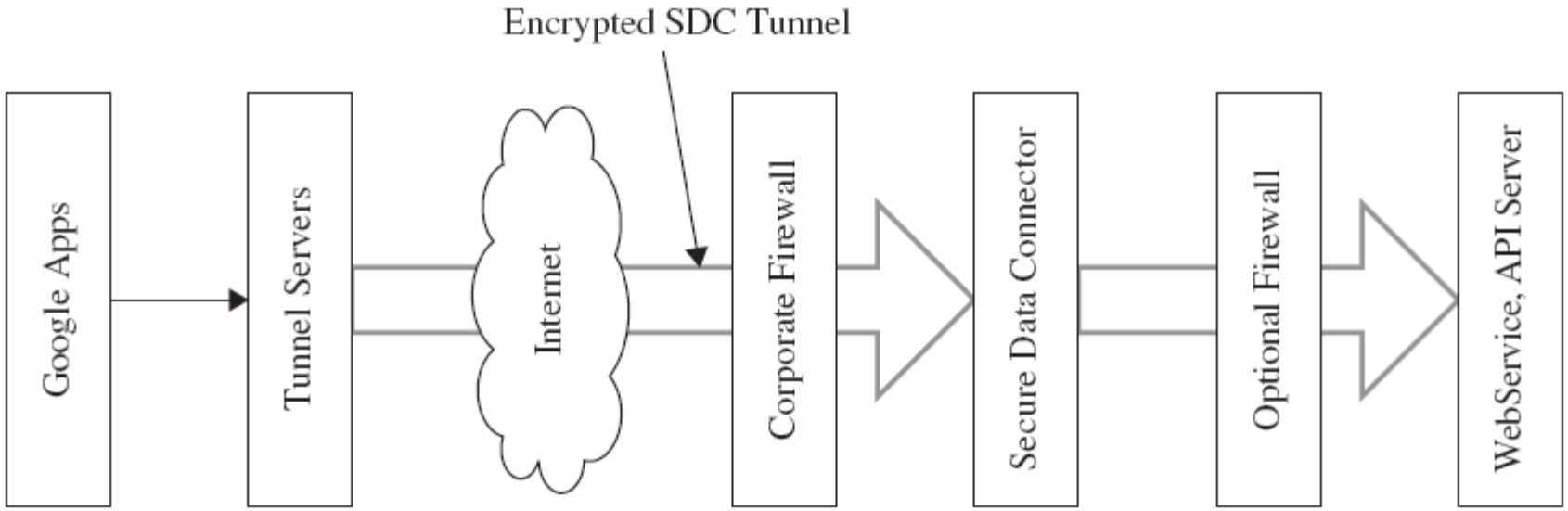
Amazon's Web Service



Microsoft Windows Azure

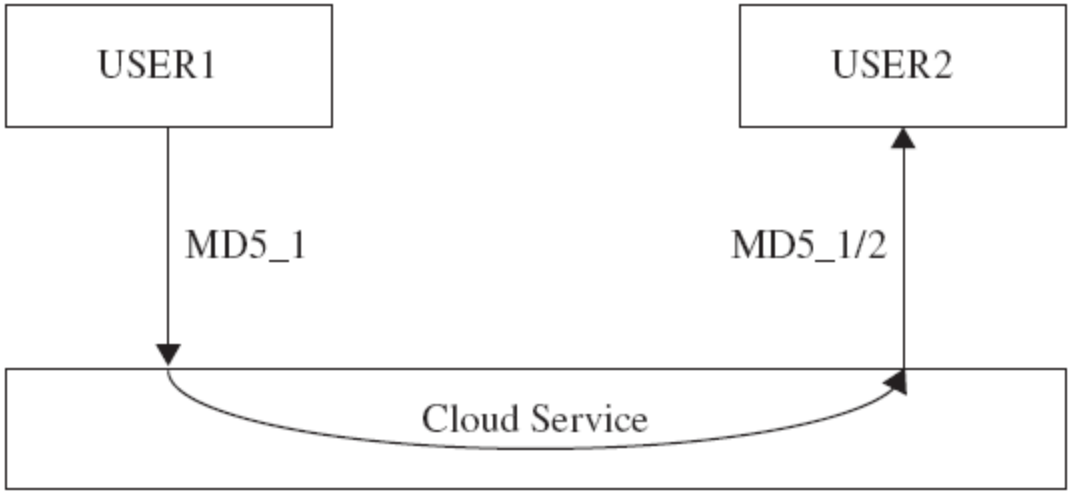


Google App Engine (GAE)



Vulnerabilities

- Confidentiality
- Integrity
- Repudiation



- Missing link between download and upload
  - Upload-to-Download Integrity
  - Repudiation Between Users and Service Providers

Solutions for missing link

- Third authority certified (TAC)
- Secret key sharing technique (SKS)
- Solutions
  - Neither TAC nor SKS
  - With SKS but without TAC
  - With TAC but without SKS
  - With Both TAC and SKS

Neither TAC nor SKS

- Uploading Session
  1. User: Sends data to service provider with MD5 checksum and MD5 Signature by User (MSU).
  2. Service Provider: Verifies the data with MD5 checksum, if it is valid, the service provider sends back the MD5 and MD5 Signature by Provider (MSP) to user.
  3. MSU is stored at the user side, and MSP is stored at the service provider side.
- Downloading Session
  1. User: Sends request to service provider with authentication code.
  2. Service Provider: Verifies the request identity, if it is valid, the service provider sends back the data with MD5 checksum and MD5 Signature by Provider (MSP) to user.
  3. User verifies the data using the MD5 checksum.

With SKS but without TAC

- Uploading Session
  1. User: Sends data to service provider with MD checksum 5.
  2. Service Provider: Verifies the data with MD5 checksum, if it is valid, the service provider sends back the MD5 checksum.
  3. The service provider and the user share the MD5 checksum with SKS.
- Downloading Session
  - User: Sends request to the service provider with authentication code.
  - Service Provider: Verifies the request identity, if it is valid, the service provider sends back the data with MD5 checksum.
  - User verifies the data through the MD5 checksum.

With TAC but without SKS

- Uploading Session
  1. User: Sends data to the service provider along with MD5 checksum and MD5 Signature by User (MSU).
  2. Service Provider: Verifies the data with MD5 checksum, if it is valid, the service provider sends back the MD5 checksum and MD5 Signature by Provider (MSP) to the user.
  3. MSU and MSP are sent to TAC.
- Downloading Session
  1. User: Sends request to the service provider with authentication code.
  2. Service Provider: Verifies the request with identity, if it is valid, the service provider sends back the data with MD5 checksum.
  3. User verifies the data through the MD5 checksum.

With Both TAC and SKS

- Uploading Session
  1. User: Sends data to the service provider with MD5 checksum.
  2. Service Provider: verifies the data with MD5 checksum.
  3. Both the user and the service provider send MD5 checksum to TAC.
  4. TAC verifies the two MD5 checksum values. If they match, the TAC distributes MD5 to the user and the service provider by SKS.
- Downloading Session
  1. User: Sends request to the service provider with authentication code.
  2. Service Provider: Verifies the request identity, if it is valid, the service provider sends back the data with MD5 checksum.
  3. User verifies the data through the MD5 checksum

TECHNOLOGIES FOR DATA SECURITY IN CLOUD COMPUTING

- Database Outsourcing and Query Integrity Assurance
- Data Integrity in Untrustworthy Storage
- Web-Application-Based Security
- Multimedia Data Security

TECHNOLOGIES FOR DATA SECURITY IN CLOUD COMPUTING

- Database Outsourcing and Query Integrity Assurance
- Data Integrity in Untrustworthy Storage
- Web-Application-Based Security
- Multimedia Data Security

Database Outsourcing

- Security Concern
  - Data privacy
    - Hacigumus et al.
    - Agrawal et al.
  - Query integrity
    - Correct and Complete
      - Merkle hash tree

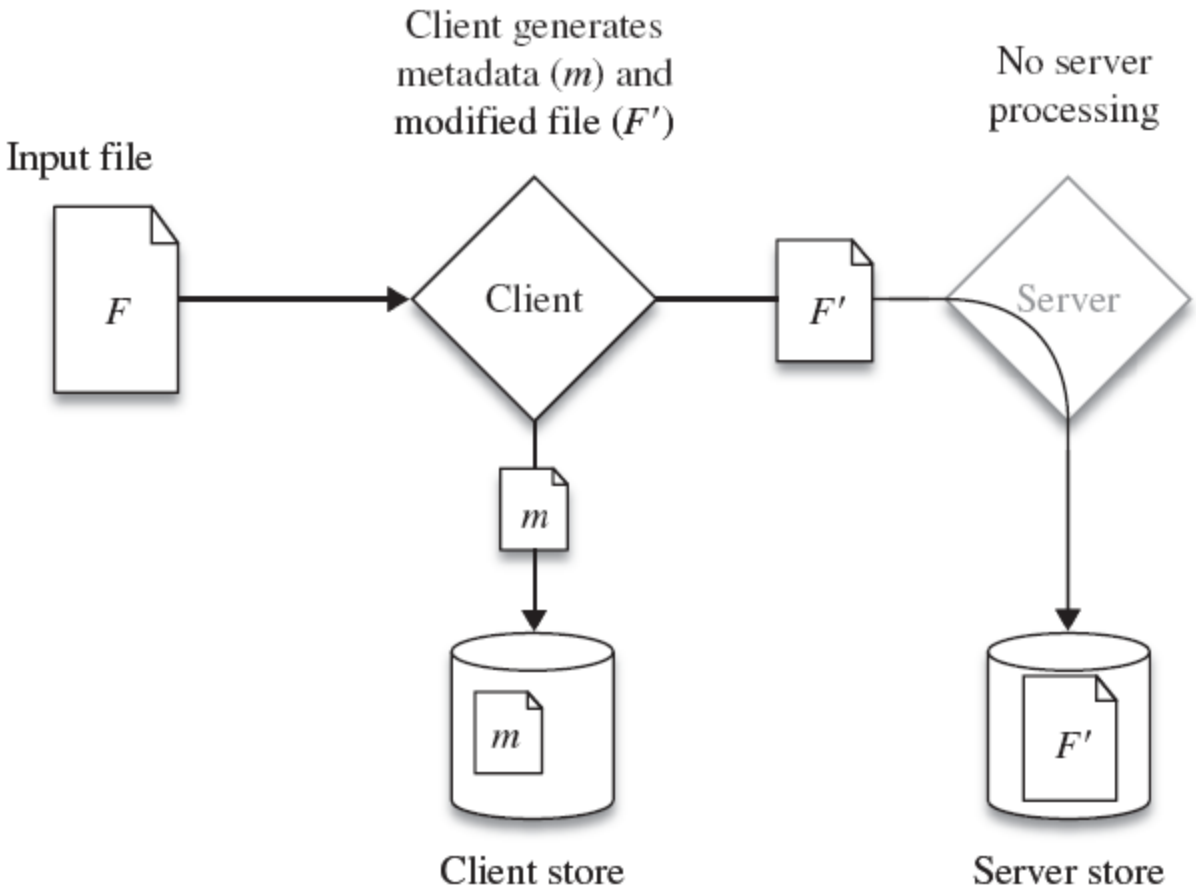
TECHNOLOGIES FOR DATA SECURITY IN CLOUD COMPUTING

- Database Outsourcing and Query Integrity Assurance
- Data Integrity in Untrustworthy Storage
- Web-Application-Based Security
- Multimedia Data Security

Data Integrity in Untrustworthy Storage

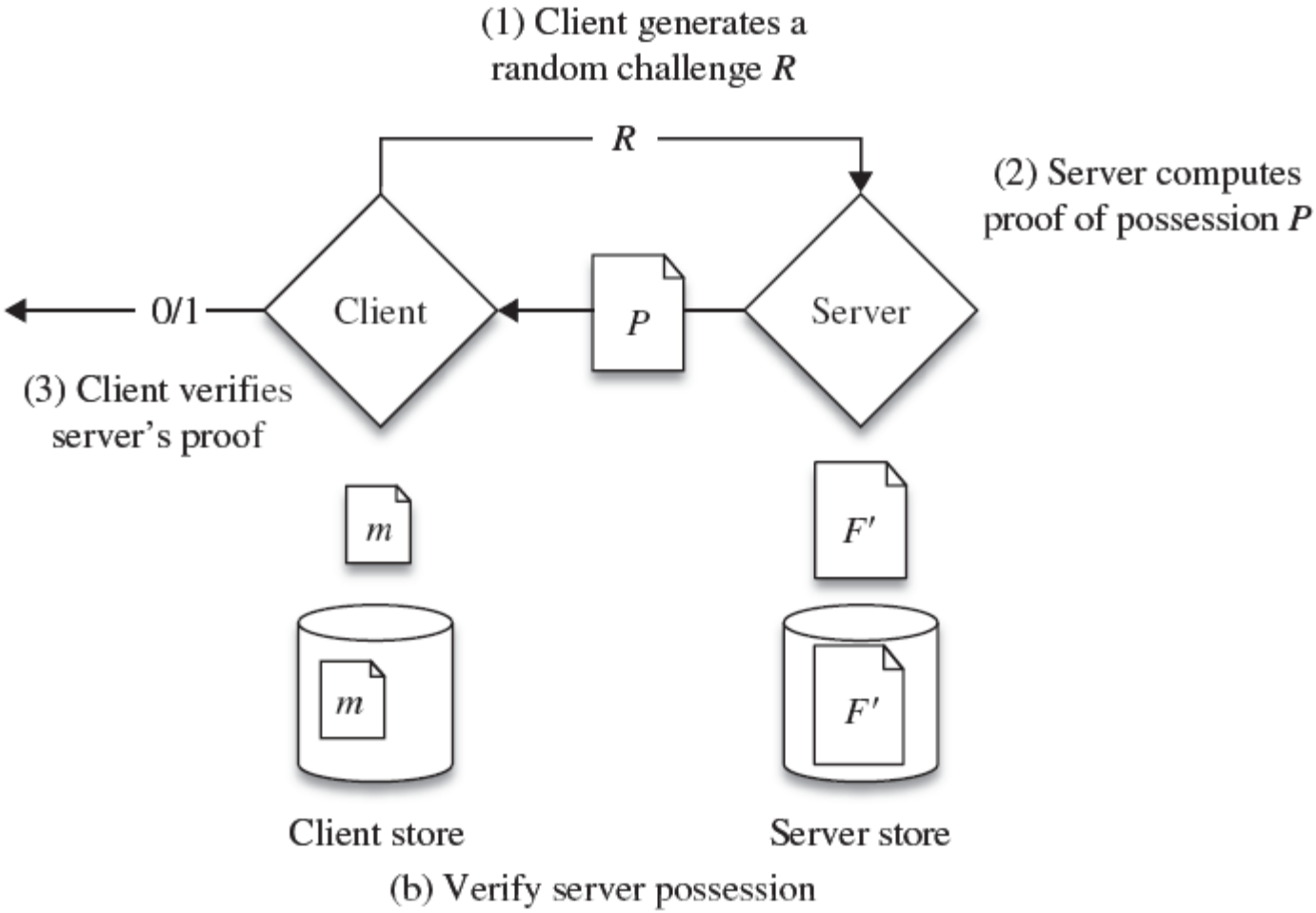
- Remote data storage possession checking protocol
  - Requirements
    1. Partial copy of the data
    2. Robust protocol
    3. High communication overhead
    4. Computationally efficient
    5. Unlimited verification
  - Technologies
    - A PDP-Based Integrity Checking Protocol
    - An Enhanced Data Possession Checking Protocol

A PDP-Based Integrity Checking Protocol



(a) Pre-process and store

A PDP-Based Integrity Checking Protocol



An Enhanced Data Possession Checking Protocol

- Enhance PDP-based protocol
  - Satisfy Requirement #2 with 100% probability
- Computationally more efficient
- Verification time has been shortened
- Trade-offs between
  - the computation times required by the prover
  - the storage required at the verifier

TECHNOLOGIES FOR DATA SECURITY IN CLOUD COMPUTING

- Database Outsourcing and Query Integrity Assurance
- Data Integrity in Untrustworthy Storage
- Web-Application-Based Security
- Multimedia Data Security

Web-Application-Based Security

- Web attack techniques
  - Authentication
    - Brute force, Insufficient Authentication, Weak password recovery
  - Authorization
    - Insufficient Authorization, Session attacks
  - Client-Side Attacks
    - Content Spoofing, XSS, CSRF
  - Command Execution
    - Like code injection or denial of service via buffer overflow
  - Information Disclosure
    - Path Traversal
  - Logical Attacks
    - DoS attack

TECHNOLOGIES FOR DATA SECURITY IN CLOUD COMPUTING

- Database Outsourcing and Query Integrity Assurance
- Data Integrity in Untrustworthy Storage
- Web-Application-Based Security
- Multimedia Data Security

Multimedia Data Security

- Protection from Unauthorized Replication
  - Advantage
    - improve system performance
  - Disadvantage
    - contents copyright
    - waste of replication cost
    - extra control overheads
- Protection from Unauthorized Replacement
  - Limited storage capacity
  - Remove stored content to make space
- Protection from Unauthorized Pre-fetching
  - Just pre-fetch necessary content

Concerns at Different Levels

- The cloud infrastructure providers (back-end)
- The cloud service providers
- The cloud consumers (front-end)
  - Application developer
  - End user

Challenges

- Technical
  - Open security profiling
  - Remote control
  - Security compliance with standards
  - Certificates
- Non-Technical
  - User's fear of losing control

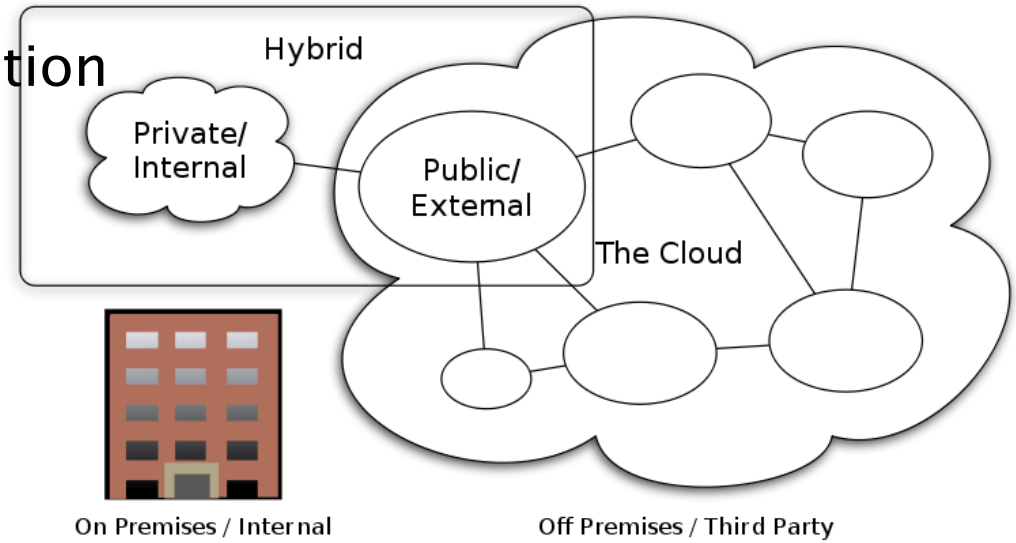
ANEKA—INTEGRATION OF PRIVATE AND PUBLIC CLOUDS

Outline

- Introduction
- Technology and Tools for Cloud Computing
- Aneka Architecture overview
- Aneka Resource Provisioning Service
- Aneka Implementation
- Future Directions and Conclusion

Introduction

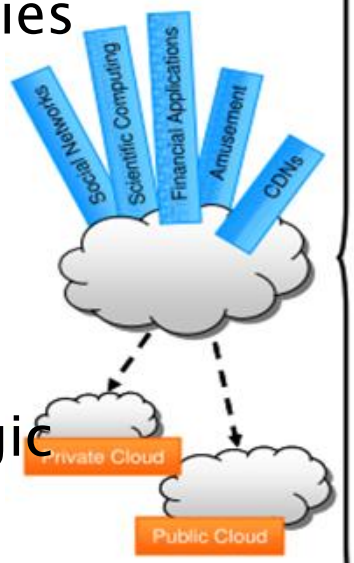
- How to form private and public cloud?
- Private cloud
  - Advantages
    - Information Protection
    - Ensuring SLA
    - Standards
  - Disadvantages
    - Scale out
    - Solution
      - Hybrid clouds
  - PaaS solutions
    - Manjrasoft Aneka

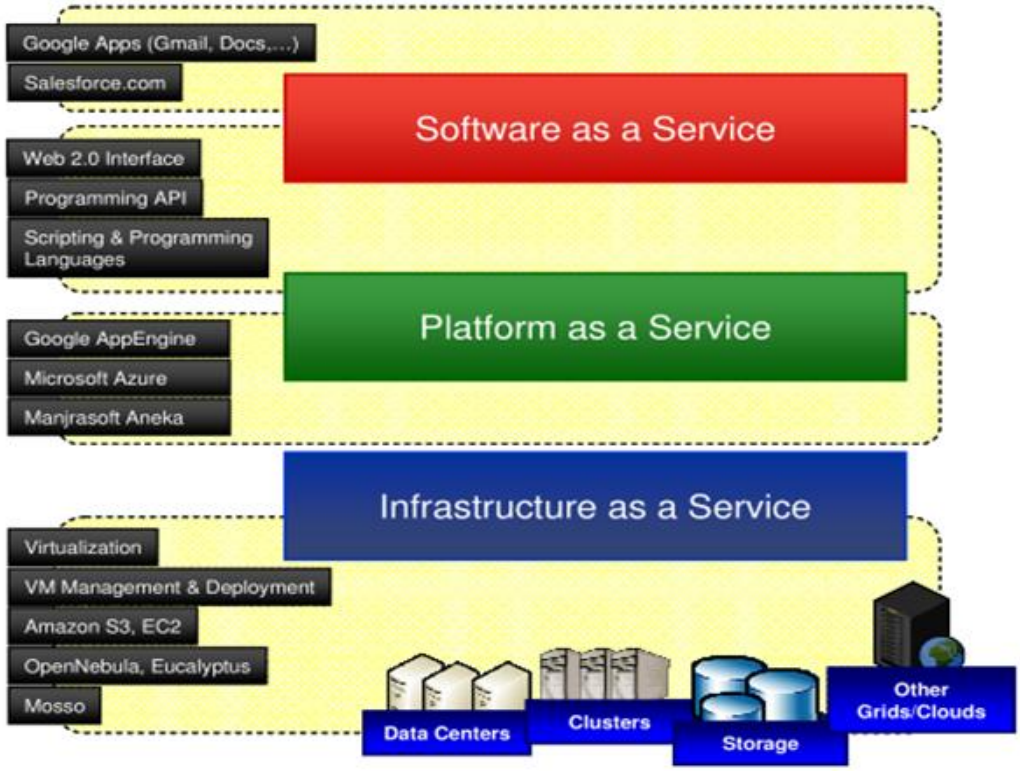


Cloud Computing Types

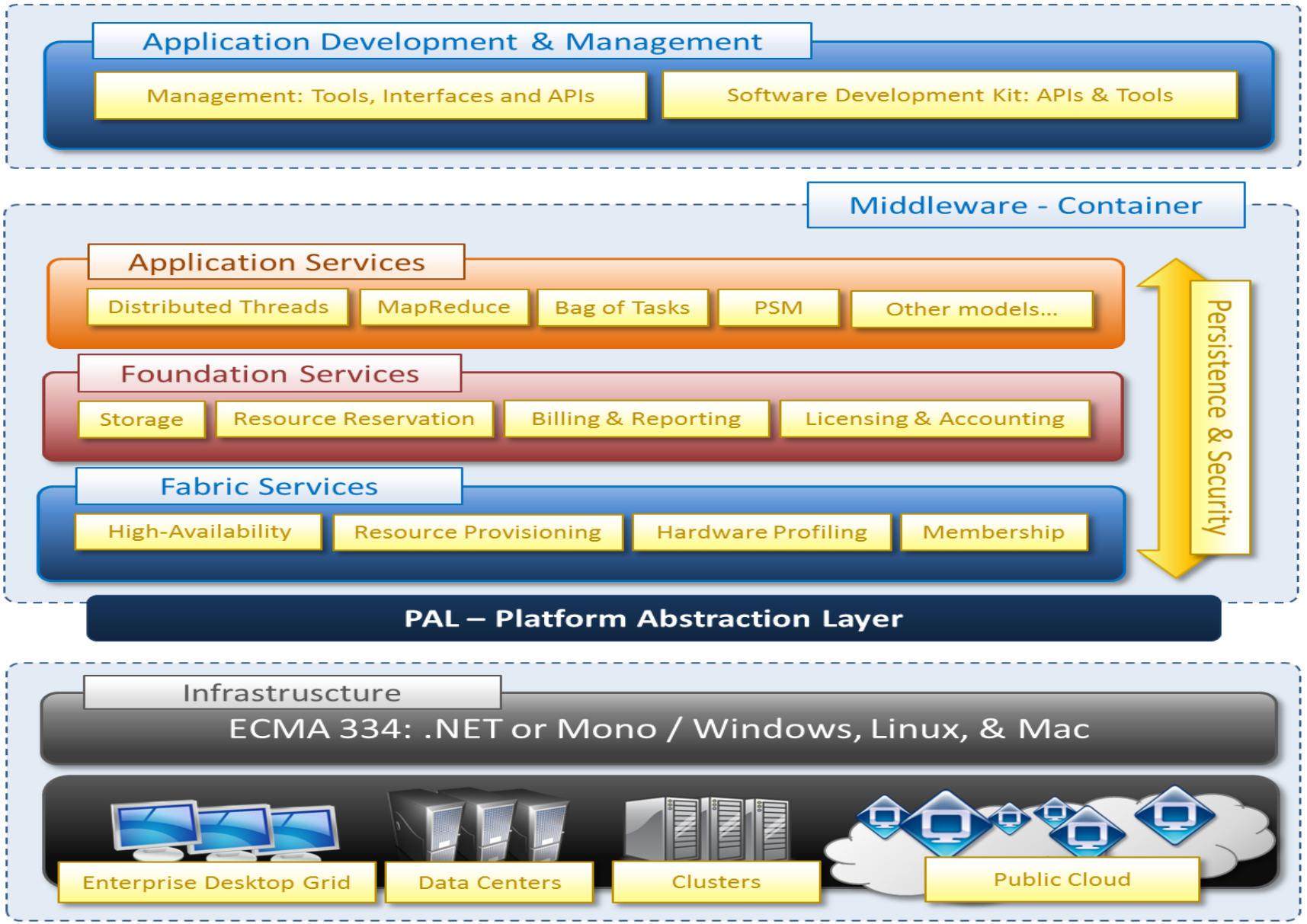
Technologies and Tools

- Deploying private cloud
  - VM technologies
  - VM managers
- IaaS
  - Amazon
    - EC2,S3
  - GoGrid
  - 3Tera AppLogic
- PaaS
  - Microsoft Azure
  - Google AppEngine
- Other Tools (commercial and research)
  - DataSynapse, Elastra, Zimory Pools, App-Logic
  - Aneka, OpenNebula, Nimbus

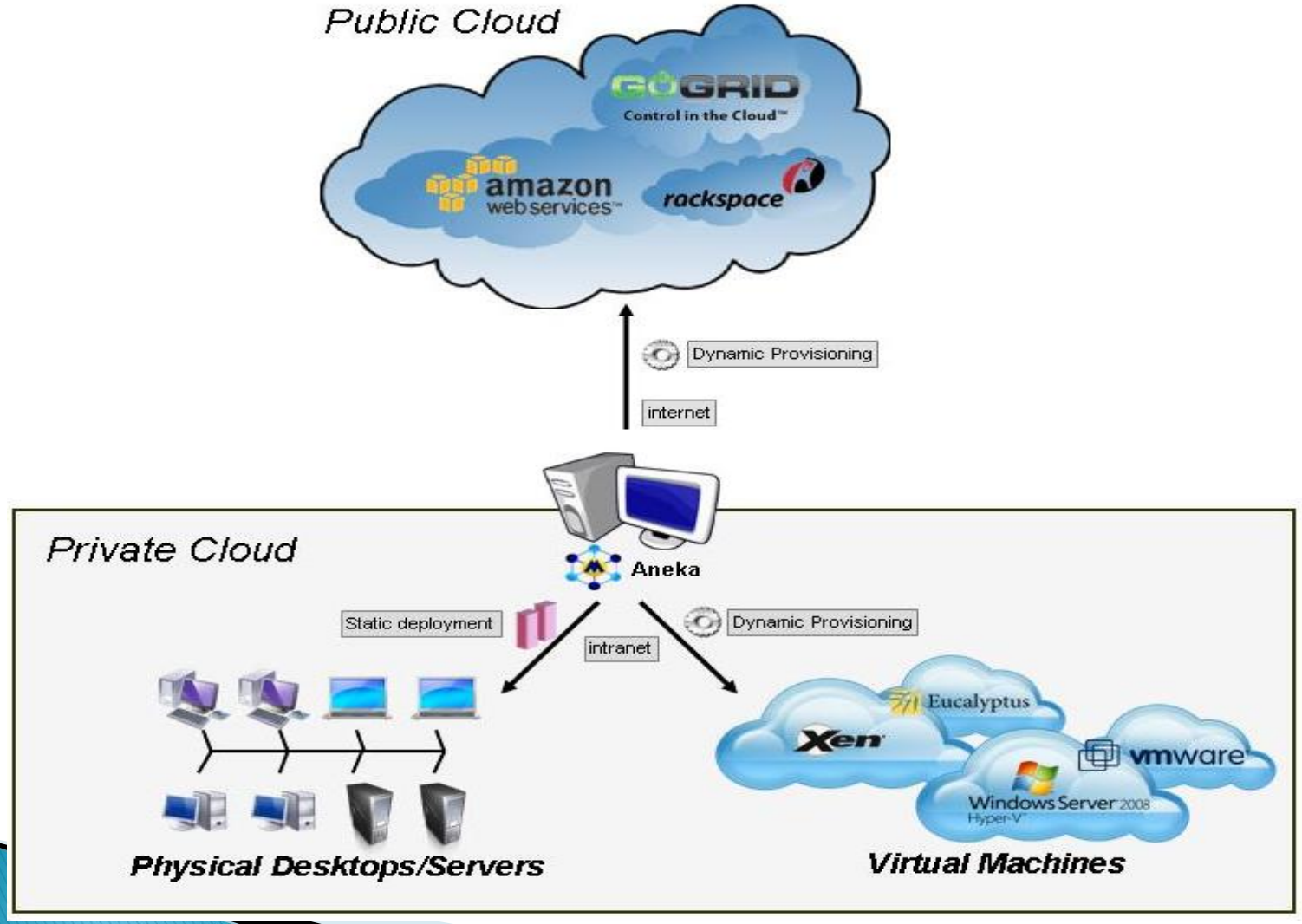




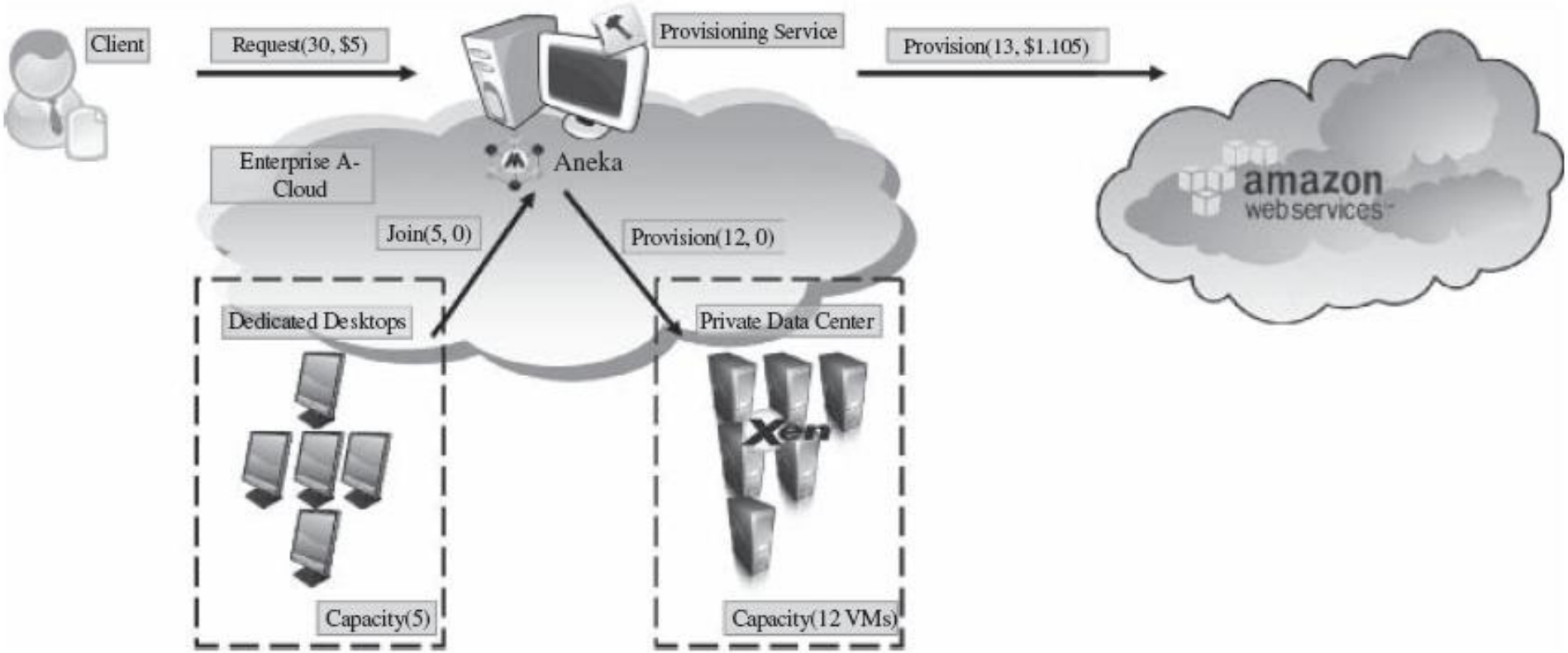
ANEKA – Architecture



Resource Provisioning



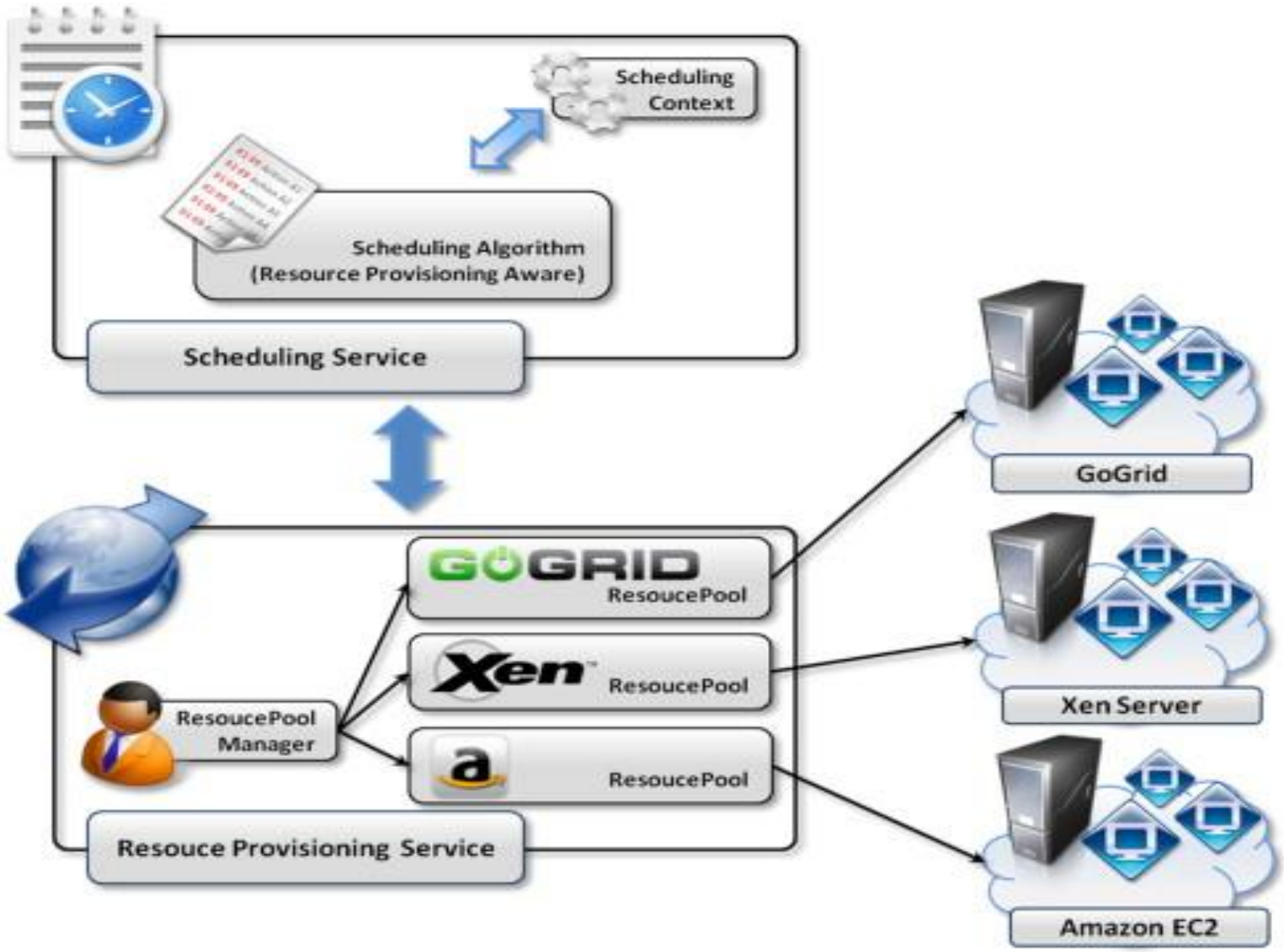
Use case of resource provisioning under Aneka



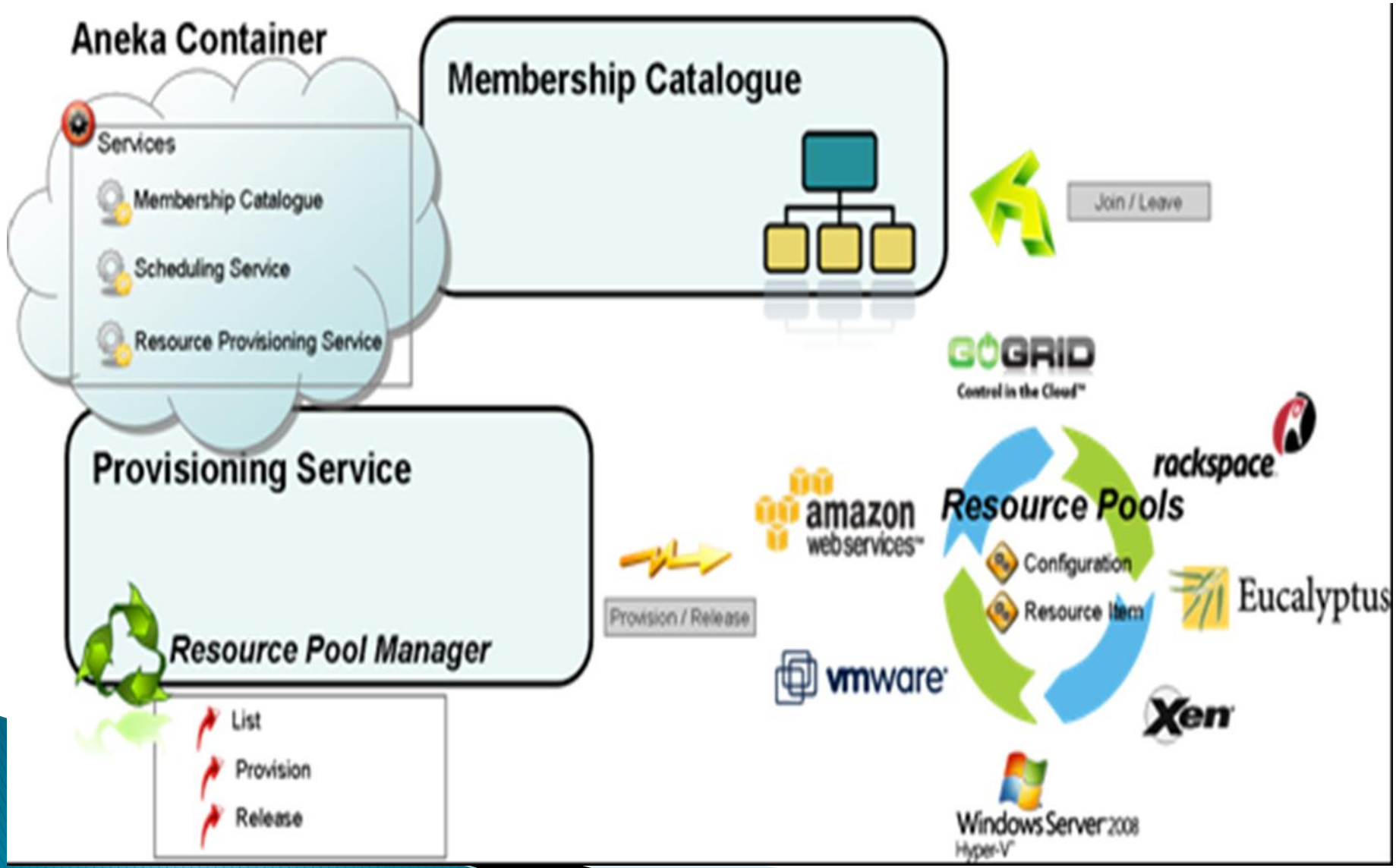
features of hybrid clouds

- Support for Heterogeneity
  - integrate additional cloud service providers (IaaS) without major changes to the entire system
- Support for Dynamic and Open Systems
  - plugging new components and rapidly integrating new features
- Support for Basic VM Operation Management
  - software frameworks that support hypervisor-based execution should implement a minimum set of operations
- Support for Flexible Scheduling Policies
  - Public and private resources can be differently utilized, and the workload should be dynamically partitioned
- Support for Workload Monitoring
  - To lease a subset of resources and dismiss resources if they are no longer necessary

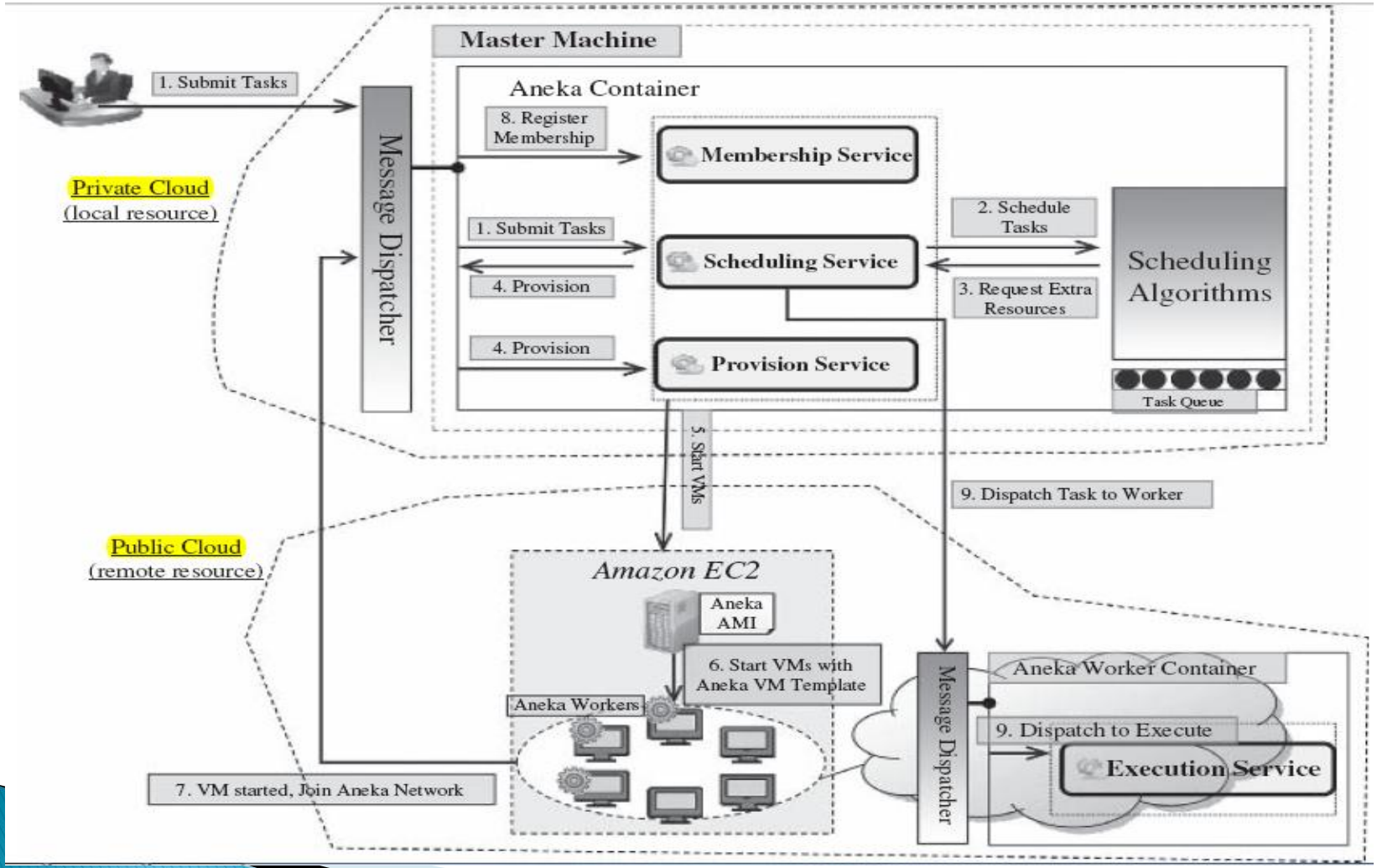
Aneka-Based Hybrid Cloud Architecture



Aneka Hybrid Cloud Architecture



Implementation for Aneka Resource Provisioning



Research in Hybrid Clouds

- Pricing Models
- Security Standardization
- Management and Scheduling Policies for heterogeneous environment
- Security in hybrid cloud
- Data retention
- Possibility of massive outage
- Provider trust
- Jurisdiction (confidentiality of data)
- Standardization

COMETCLOUD: AN AUTONOMIC CLOUD ENGINE

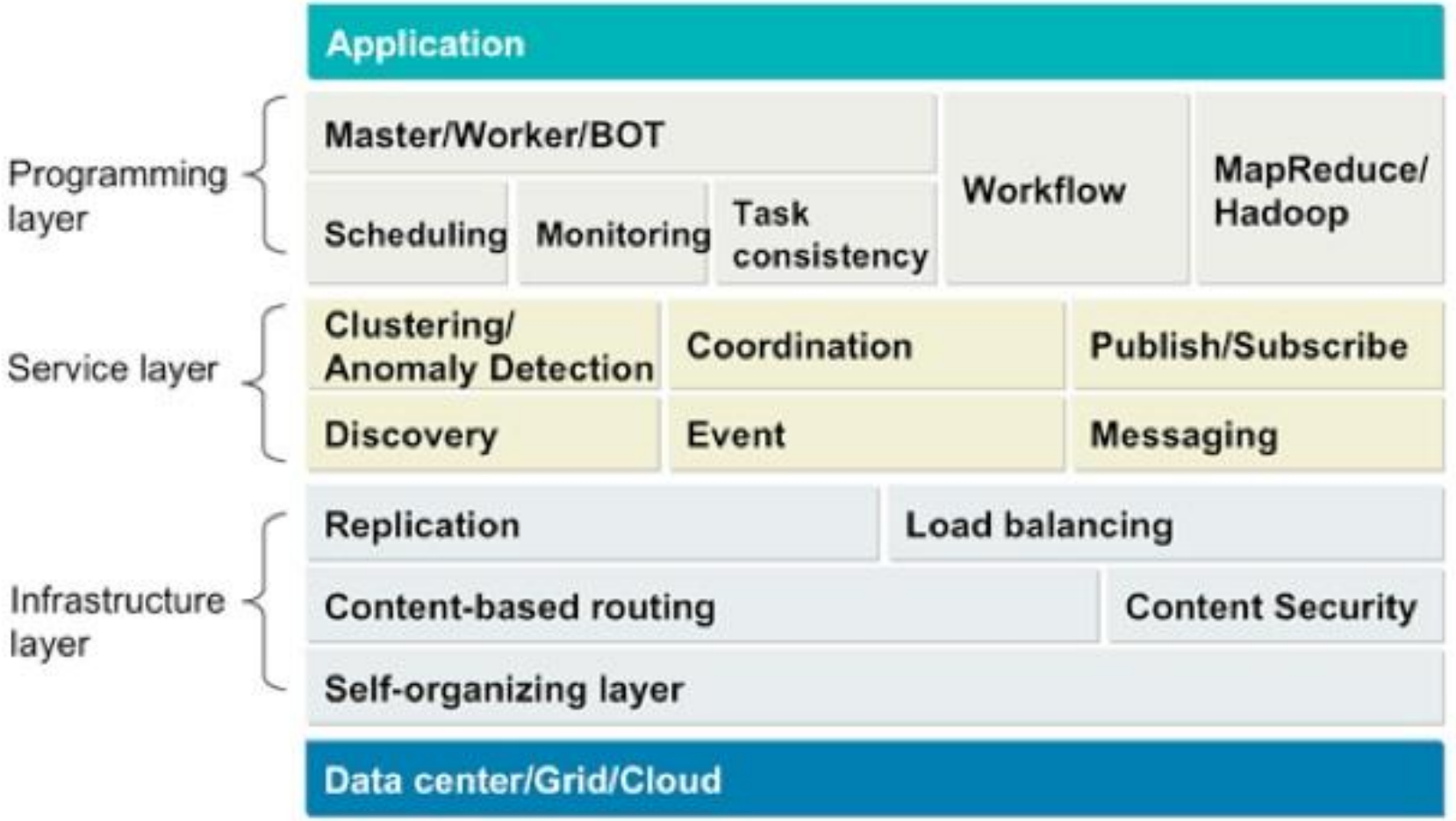
Outline

- Introduction
- Architecture overview
- Autonomic behavior of CometCloud
- Overview of CometCloud-based applications
- Implementation and Evaluation
- Future Research Directions

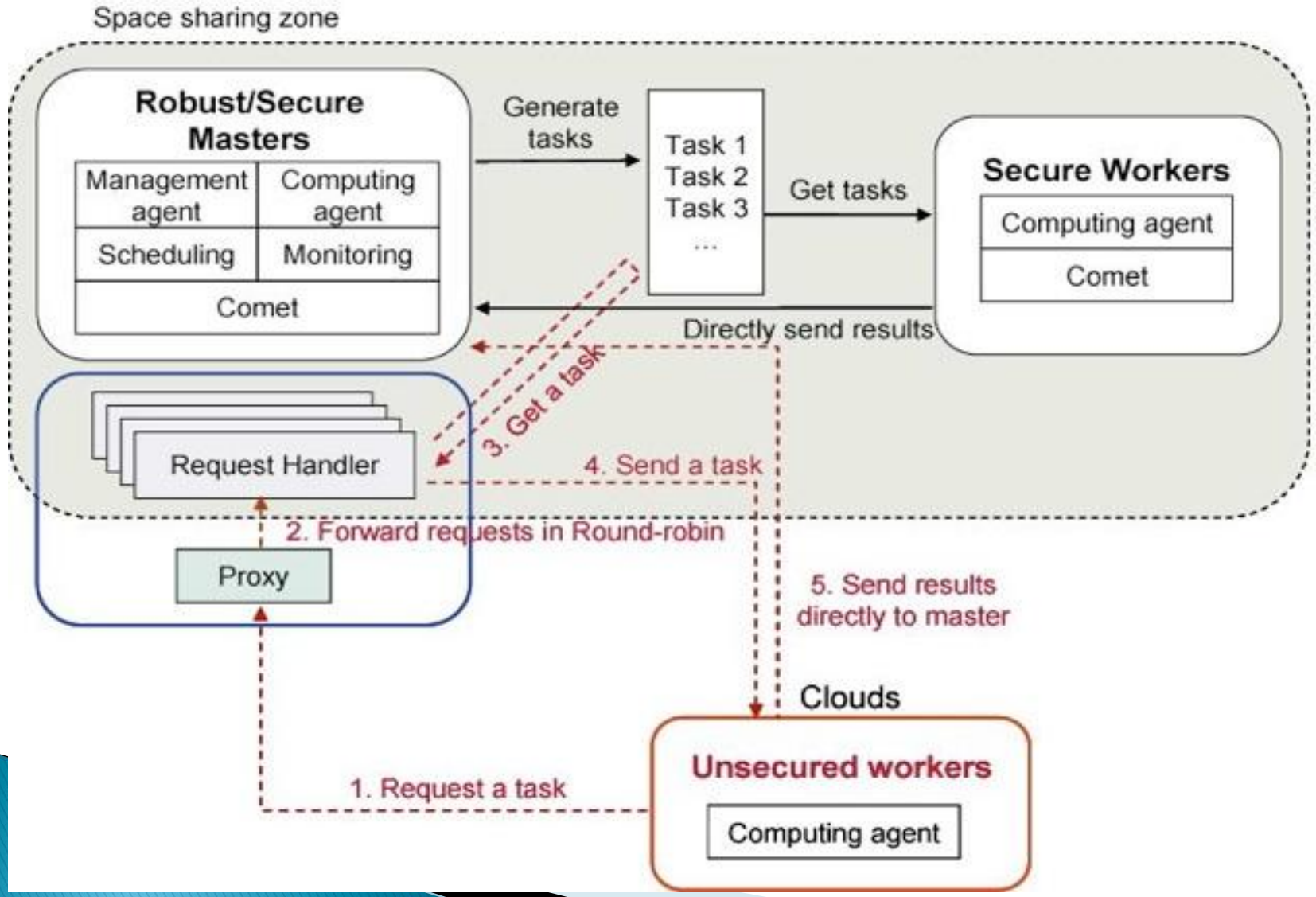
Introduction

- What
  - Integrates of public and private cloud
  - Is a PaaS
- Why
  - to enable on-demand scale-up, scale-down and scale-out
- How
  - Cloudbursting
  - Cloudbridging

Architecture



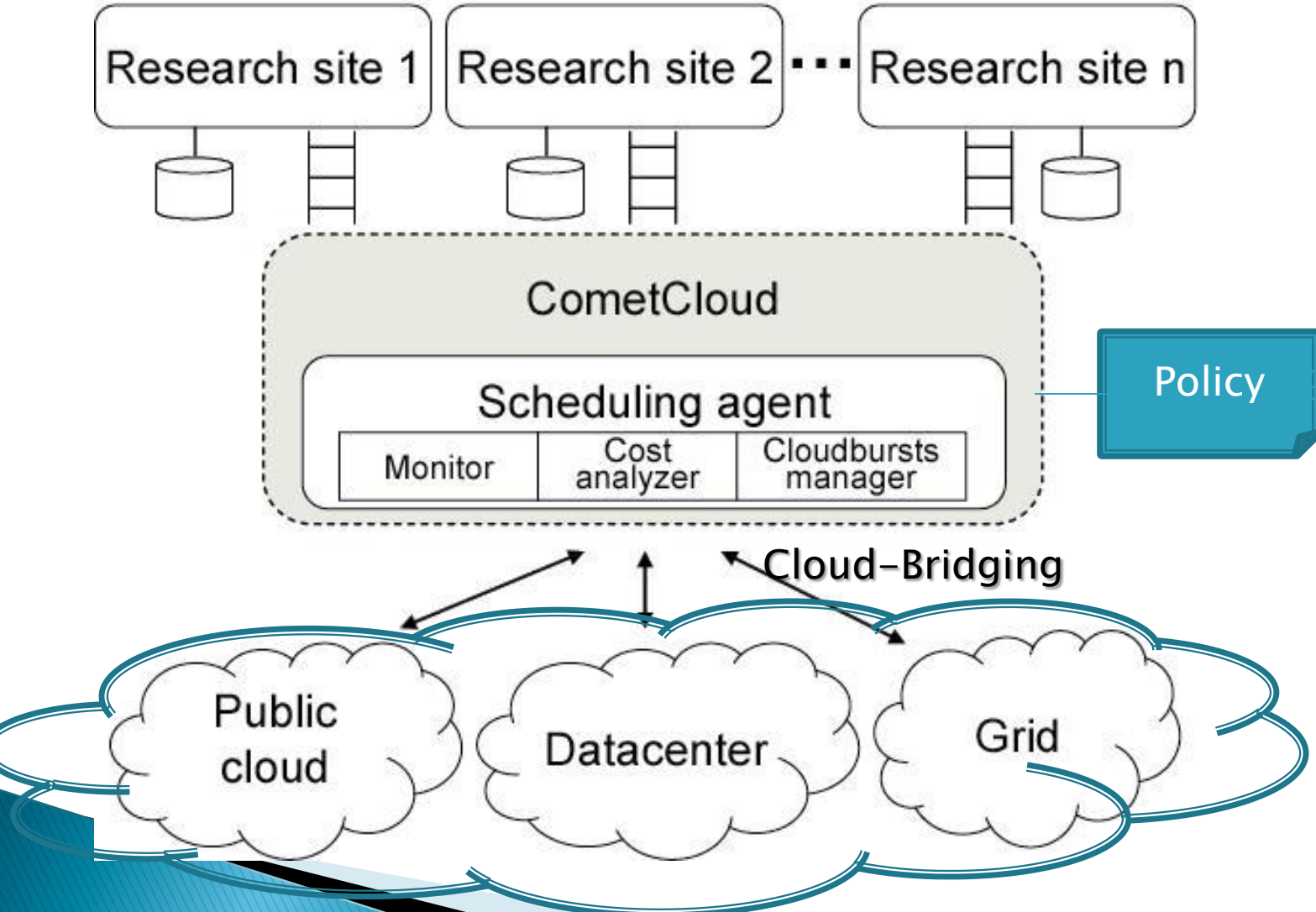
Automatic Cloudbursting



Motivations on Cloudbursting

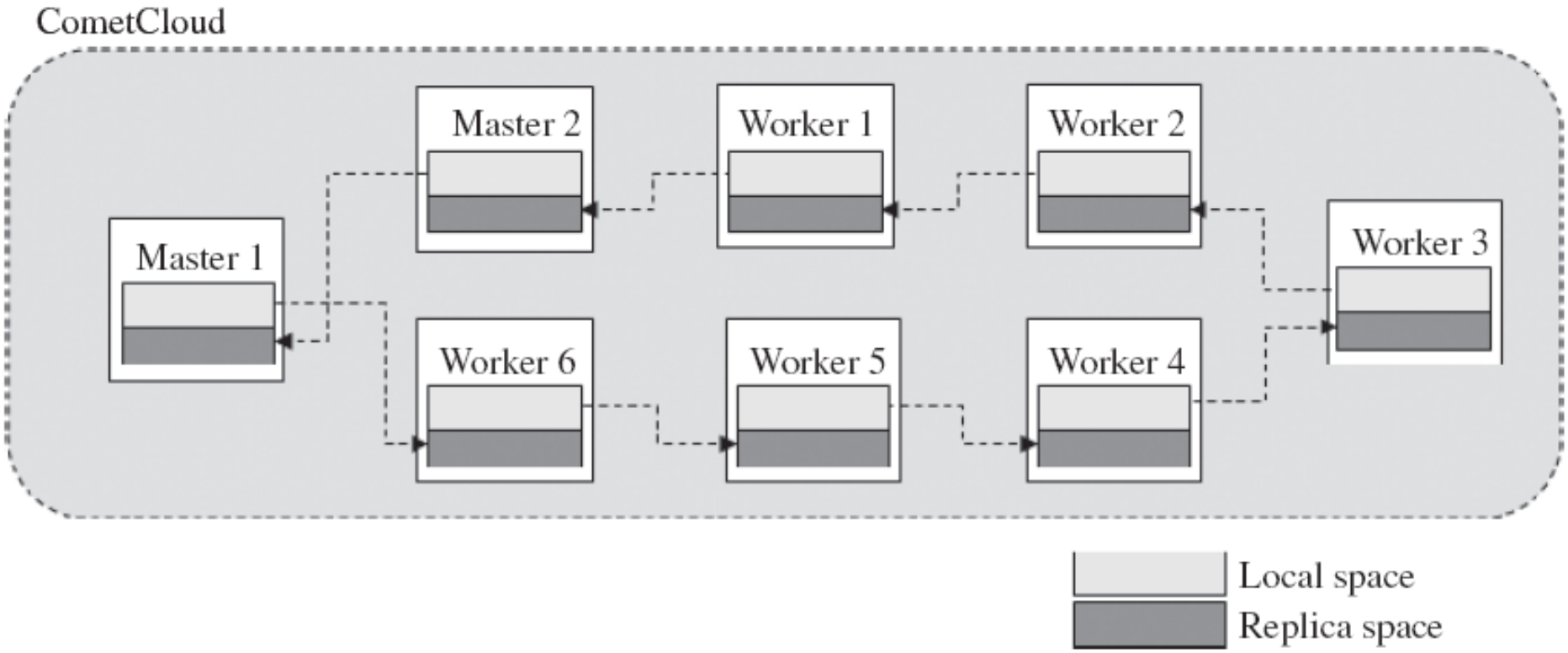
- Load Dynamics
  - The computational environment must dynamically grow (or shrink)
  - In response to dynamic loads
- Accuracy of the Analytics
  - The required accuracy of risk analytics
  - To dynamically adapt to satisfy the accuracy requirements
- Collaboration of Different Groups
  - Different groups run the same app. with different dataset policies
  - To satisfy their SLA.
- Economics
  - Application tasks can have very heterogeneous and dynamic priorities.
  - To handle heterogeneous and dynamic prov. and sched. requirements.
- Failures
  - To manage failures without impacting application QoS.

Automatic Cloudbridging



Virtually Integrated working cloud

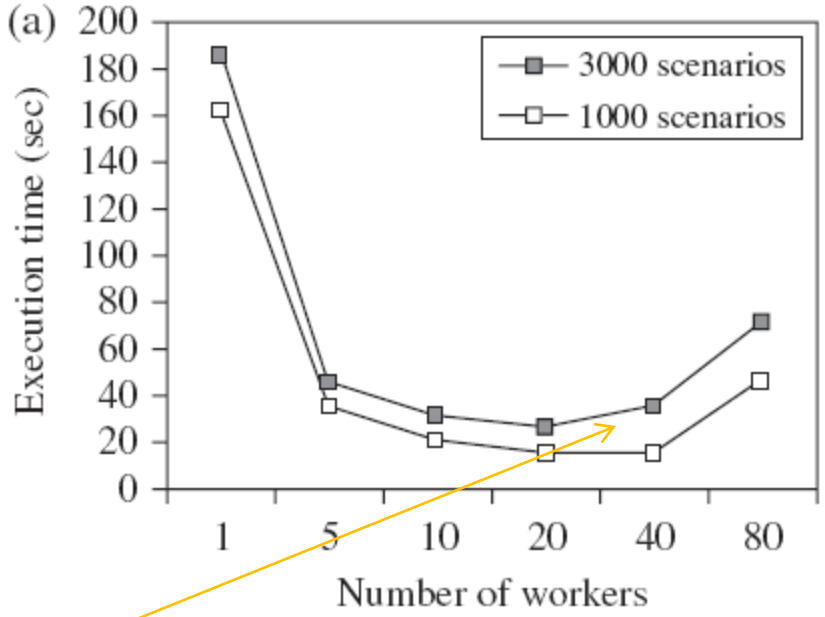
Fault Tolerance

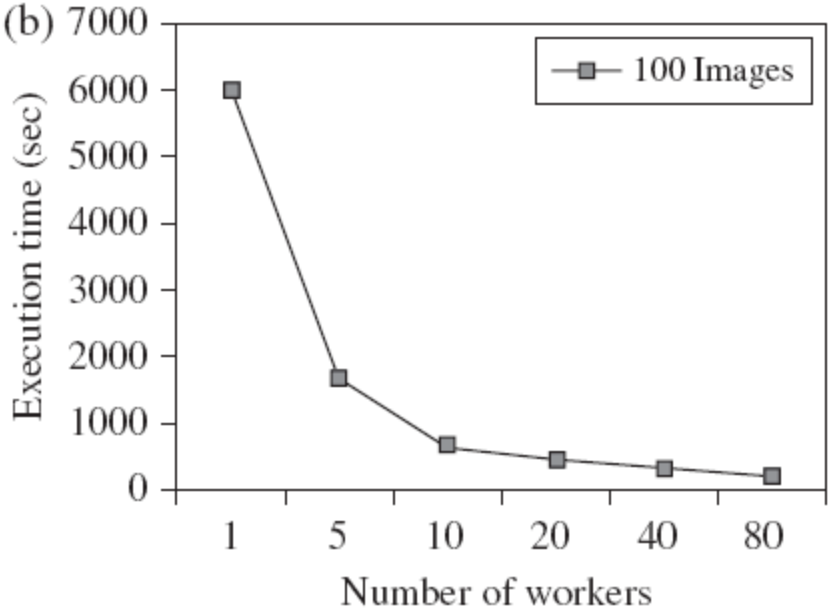


CometCloud based apps

- VaR
  - measuring the risk level of portfolios of financial instruments
  - VaR calculation should be completed within the limited time
  - computational requirements can change significantly
  - autonomic cloudbursts
  - Workload-based policy
- Image Registration
  - determine the mapping between two images
  - for medical informatics
  - budget-based policy

Application Runtime on EC2





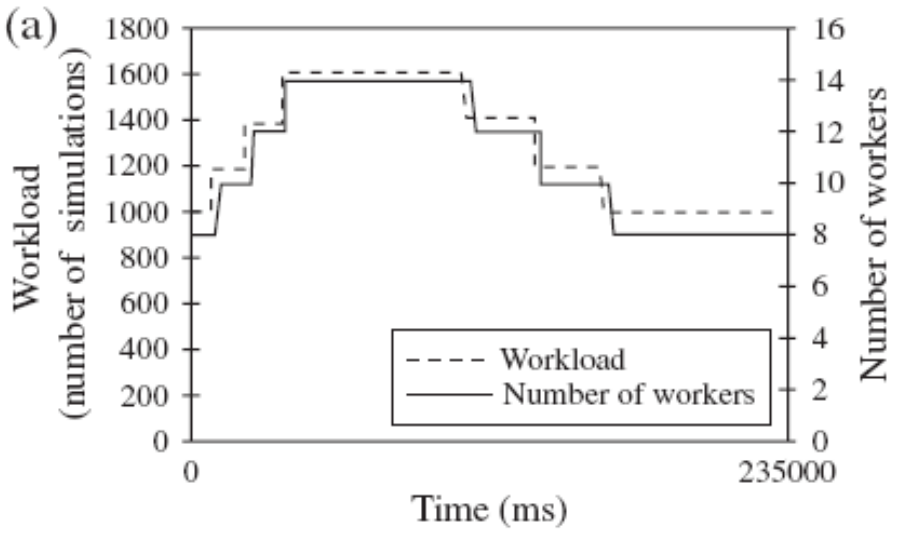
Communication Overhead

- All worker were unsecured
- Each worker ran on different instance

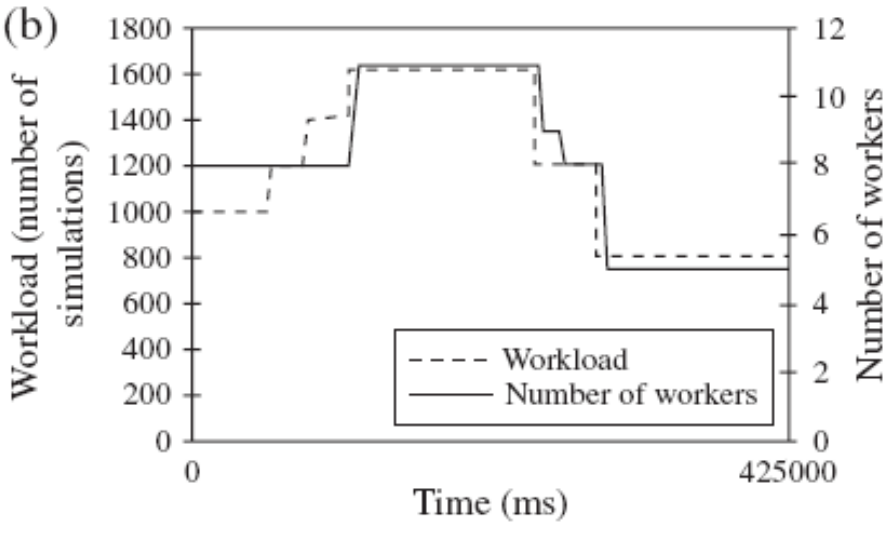
a: VaR

b: Image Registration

Automatic Cloudbursts Behaviors



a: Workload-specific policy



b: Workload-bounded policy

VaR using Workload-Based Policy

Automatic Cloudbursts Behaviors

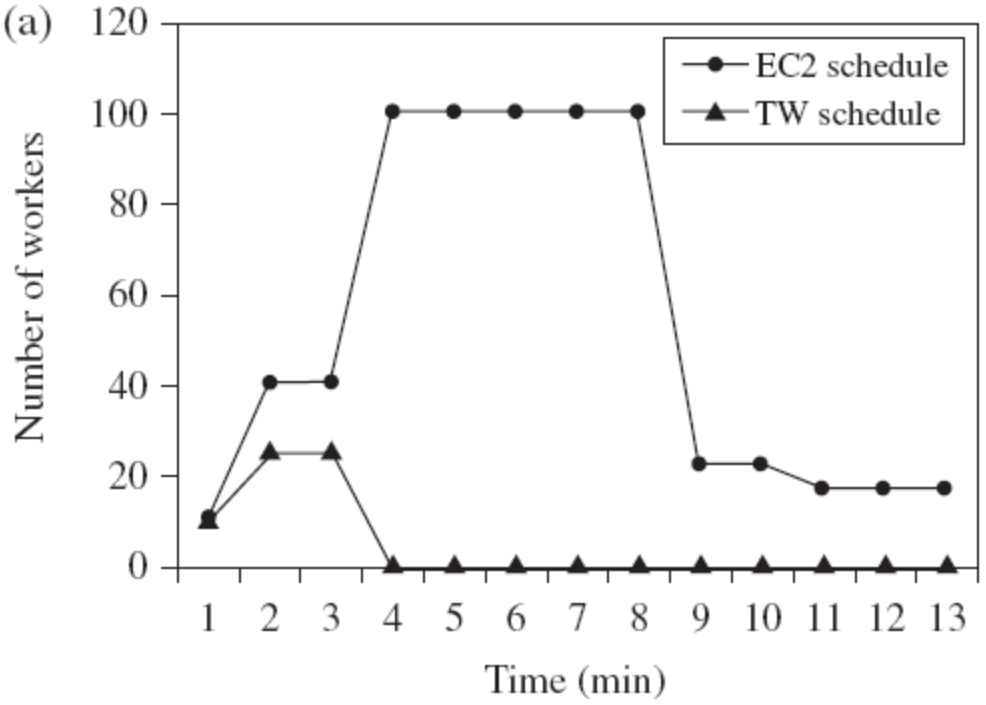
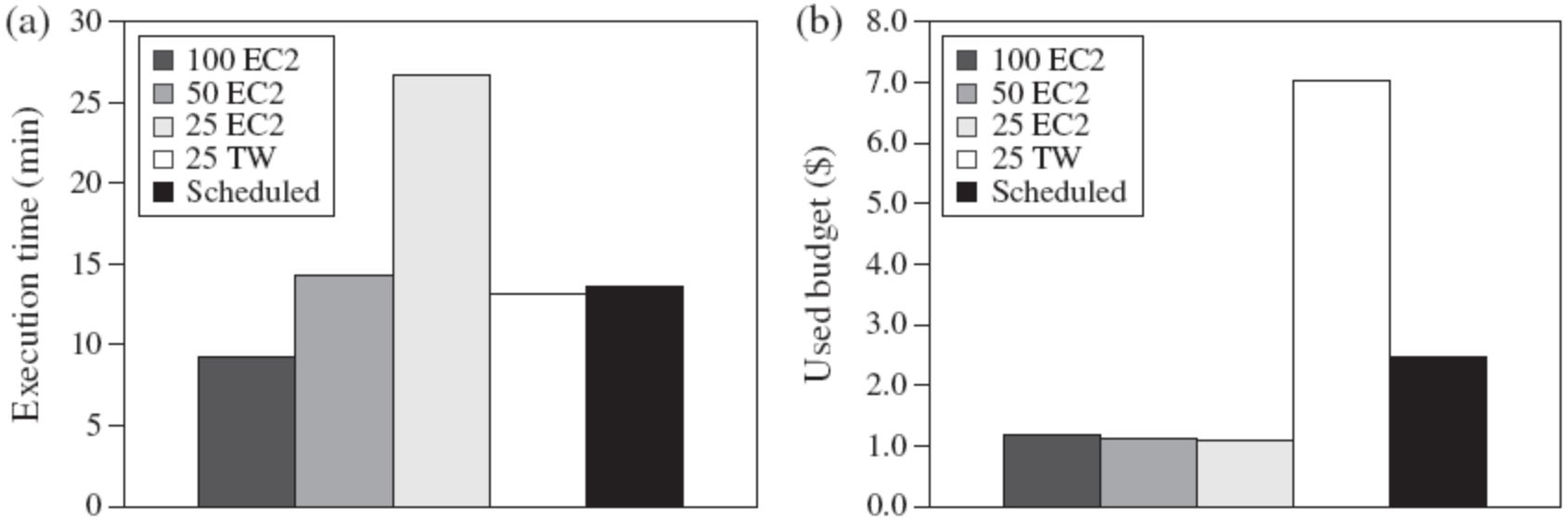


Image Registration using Budget-Based Policy

With/Without Scheduling Agent



Rationale

- **Cloud Computing for HPC**
  - Introduction
  - Cloud and GRID
  - Security Issues
- **PerfCloud**
  - Proposed approach and overall architecture
- Access control and Identity Federation in PerfCloud
- **Conclusions**

Cloud Computing and HPC

According to the definition of NIST, Cloud Computing is a model for enabling on demand network access to a shared pool of configurable resources

- **Cloud Computing delivery models**
  - IaaS (Infrastructure as a Service)
  - SaaS (Service as a Service)
  - AaaS (Application as a Service)
- Clouds to provide ”servers”, to provide ”application environments”, to provide ”datacenters”,……
- **Cloud for HPC => IaaS**
  - Performance
  - Interconnections
  - Security (give administration rights to consumers)

Clouds, GRID and Performance

- The use of clouds for HPC makes sense only if performance is satisfactory
- The availability of an existing GRID infrastructure is a great opportunity to be exploited
- Resources provided by clouds can be used with grid (standard?) access mechanisms
- Comparison of Cloud and GRID is an open discussion (management of great number of distributed/computational resources, huge datacenters, different approach towards the applications)

PerfCloud: Cloud Computing and GRID Integration

Cloud on GRID:

- The complex and stable GRID infrastructure is exploited to build up a cloud environment.
- A set of GRID services is offered in order to manage (create, migrate, ...) virtual machines, usually organized in (Virtual) Clusters.
- A standard way to access the Cloud (via GRID interfaces – read Web Services interfaces)

PerfCloud: The Approach

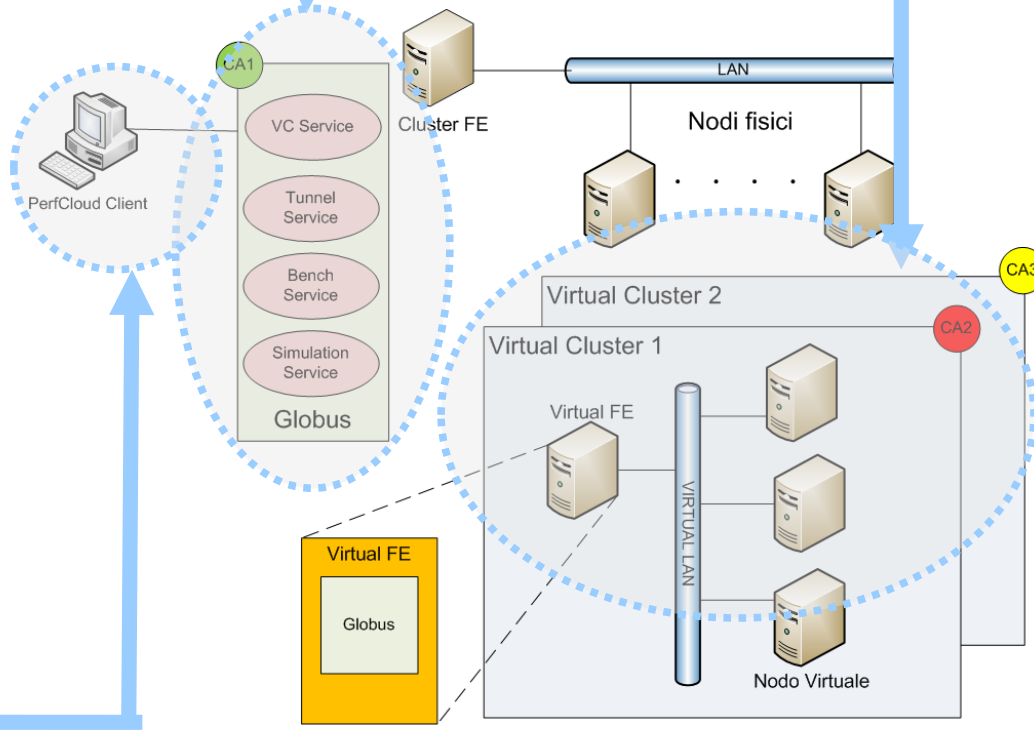
PerfCloud is a complete framework that provides (virtual) cluster-on-demand functionalities integrated with performance prediction services and a Gui client:

- To provide a virtual cluster (with a set of pre-installed applications) with its security domain, giving full management to users
- To evaluate on-the-fly the performance of an application on the VC created,

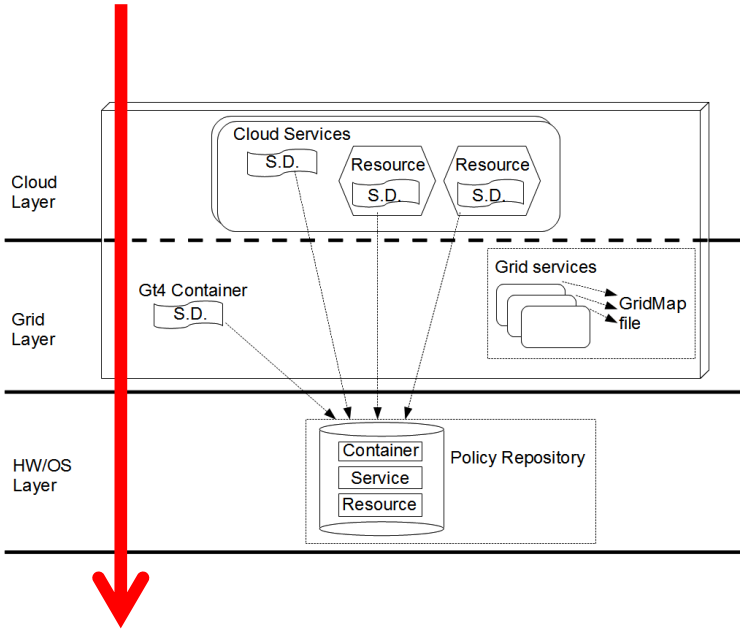
PerfCloud: Overall Architecture

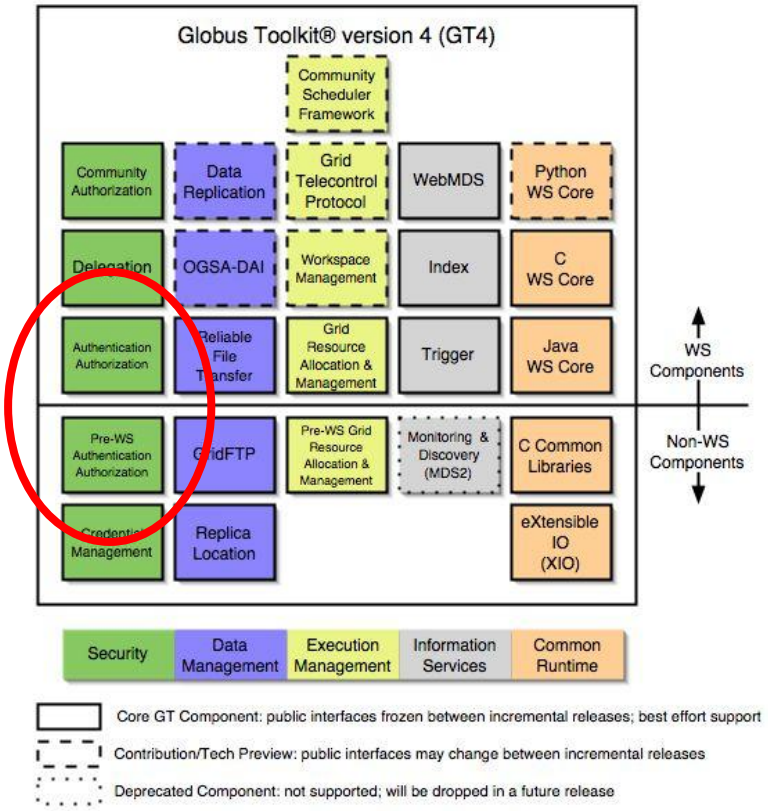
It is composed of:

- **GRID Services** able to manage, evaluate and predict performances of Virtual Clusters
- **Virtual Clusters** Machine **Images** preconfigured for HPC
- **Clients** for easy access to the environment



PerfCloud from a security point of view (1/2): Access Control to virtual and physical resources

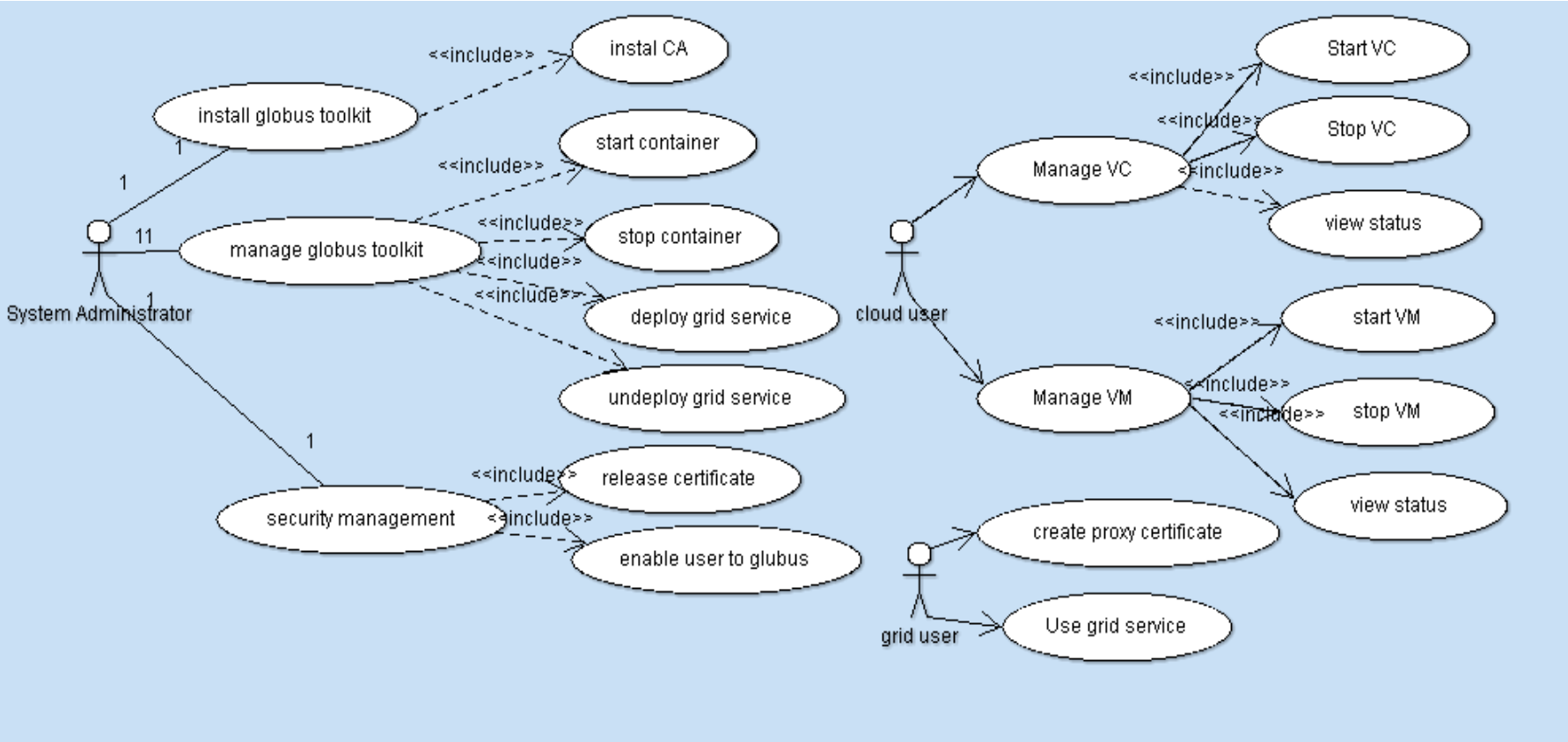




Resources to protect

Available GT4 components

Analysis of access control profiles



System/GRID Administrator
GRID User

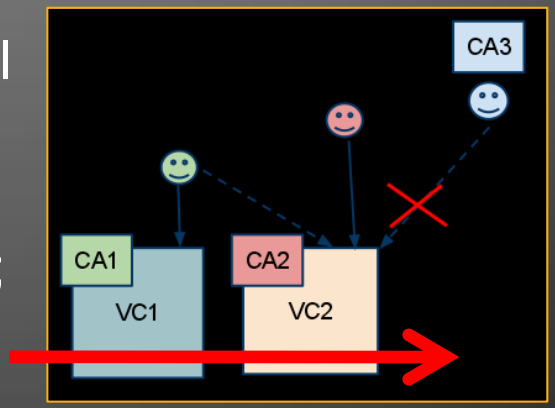
} Manage and access physical resources

Cloud Administrator
Cloud User

} Manage and access virtual resources

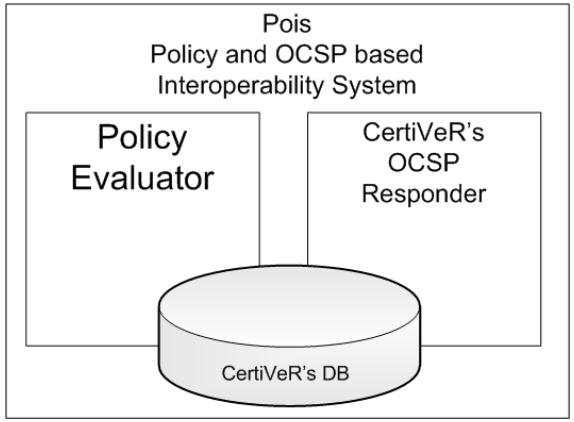
PerfCloud from a security point of view (2/2): Authentication and Identity Federation

- GRID user authentication is based on digital certificates (x.509 and proxy certificates).
- Digital Certificates are accepted if the basic path validation process is successful; it implies that all CA in the certification path are trusted and all certificates are valid.
- To validate certificates from external untrusted domains an extended path validation is required; it implies that there is a cross certification among different CAs that can form or not an explicit federation (hierarchical or peer-2-peer), this operation is manually performed.
- To fully authomatize the process of extending trust to other CA and so enable the idenity federation, we propose a system to evaluate on-line the CRL and evaluate the security level associated to a CA.



POIS: Policy and OCSP based Interoperability System

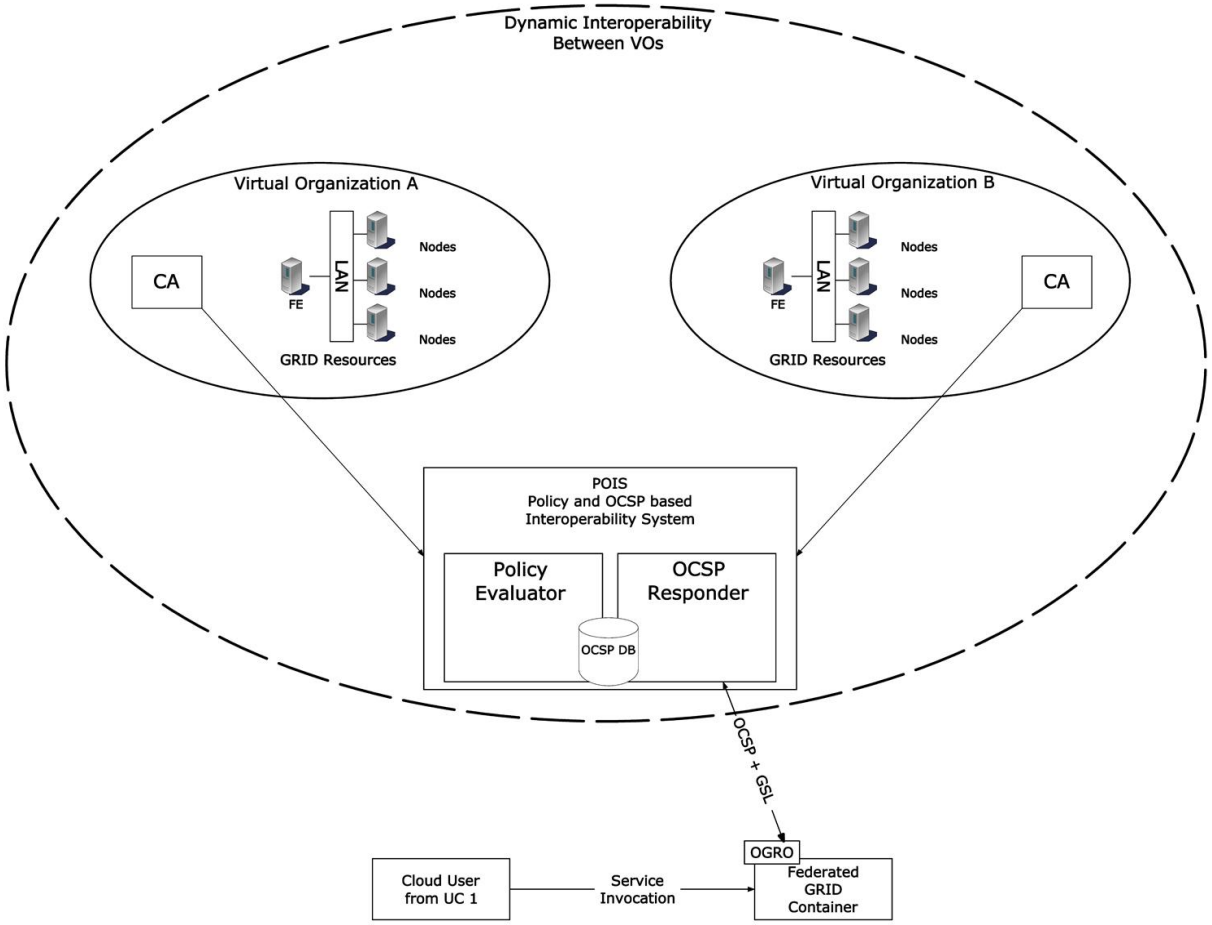
- Enable Extended Path Validation in untrusted Grid domains.
- Our approach is to build a dynamic cross certification (federation) of CAs by evaluating their Certificate Policies, on the basis of 3 components:
  - An automatic policy evaluation methodology (REM),
  - An OCSP Client (OGRO),
  - An OCSP Responder (as CertiVer)
- In order to define the Certificate Policy and further audit the CA, we refer to a Trusted Third Party .



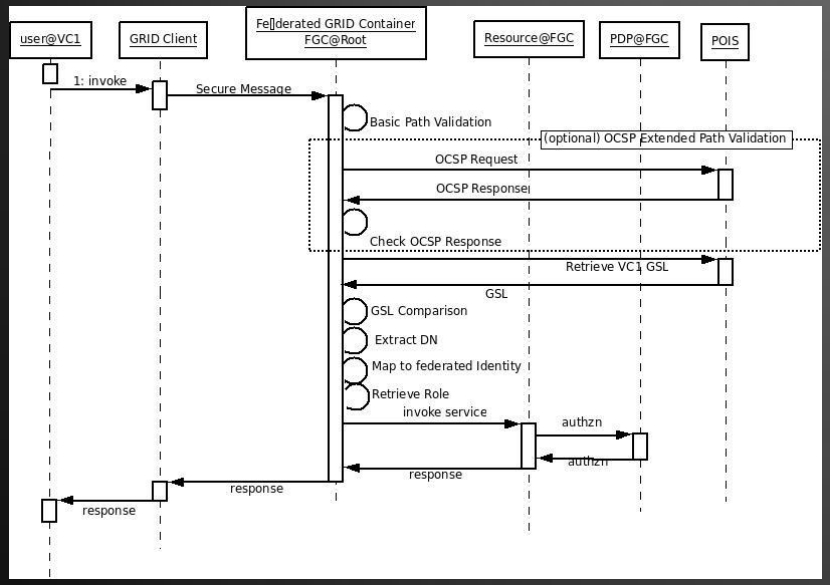
The REM methodology to evaluate a Certificate Policy and extend trust to other CAs

1) formalize a policy according to a common template;
2) each provision is structured and normalized according to a Local Security Level
3) an aggregation function which is based on an Euclidean distance gives the Global Security Level associated to the policy

POIS: Policy and OCSP based Interoperability System



An example scenario: access to federated resources



- basic path validation on the proxy certificate is performed;
- the digital certificate status is evaluated on-line through the OCSP Responder;
- the GSL value is directly retrieved from the POIS (that holds a database with all pre-evaluated Certification Authorities).
- the GSL of the Cloud user's CA is compared against the minimum required-GSL defined by the Federated Grid Container to extend trust, and if GSLV C1 > GSLGC, the validation is successful.
- If the extended path validation is successful, the cloud user is mapped to a "federated user".

PerfCloud Authentication and Authorization mechanisms

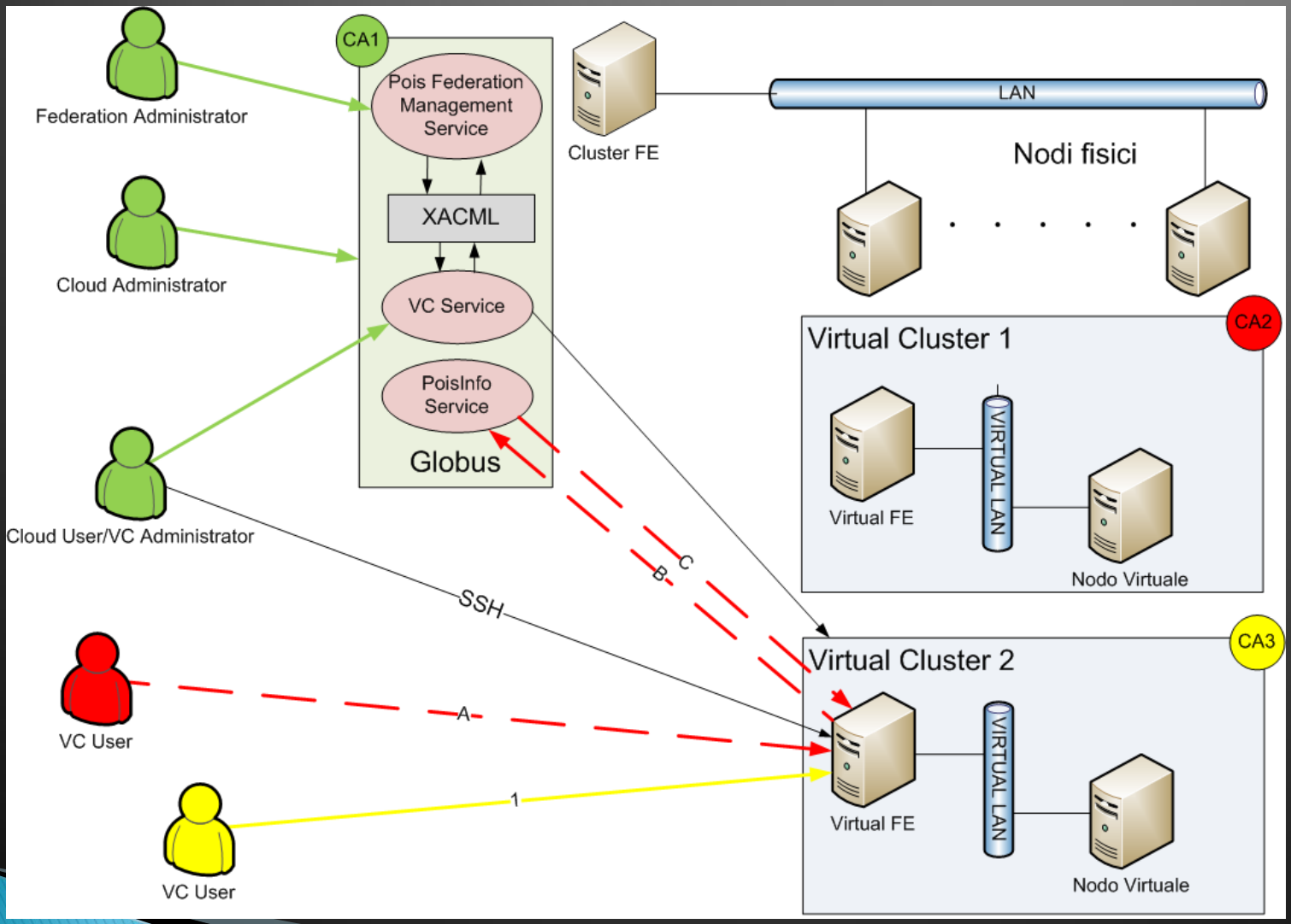
- Authentication mechanisms:
  - **None.**
  - **GSISecureMessage**: each individual message is encrypted.
  - **GSISecureConversation** : a secure context is established.
  - **GSITransport**: transport-level security is provided by using TLS.

WS-Security specification

- Authorization mechanisms:
  - **Container security level** (Authzn to access a container);
  - **Service security level** (Authzn to access a service);
  - **Resource security level** (Authzn to access a resource);

Policy-based Authzn services

POIS services in PerfCloud



Conclusions

- PerfCloud offers cluster-on-demand functionalities integrated with a simulation environment able to predict user application performance on the newly instantiated Virtual Clusters
- We have analyzed cloud-on-grid security issues and in particular, the access control problem and the identity federation among untrusted virtual clusters.
- As for access control, we identified the main roles within the PerfCloud and we are able to enforce different security policies to separate the access to physical and virtual resources.
- As for identity federation: an innovative interoperability system has been proposed to perform the extended path validation of digital certificates in an automatic way.
- Future works:
  - Performance/security tradeoff (SLA)

Next Generation Web Apps

Towards Transformative UX

Modern Web applications

Web 2.0

Social

user involvement in the creation of contents

culture of participation [Fischer 2009]

Active co-creation of knowledge and new ideas

Development of modern Web applications

public APIs

Application Integration

data sources

Data Integration

Integration

widgets

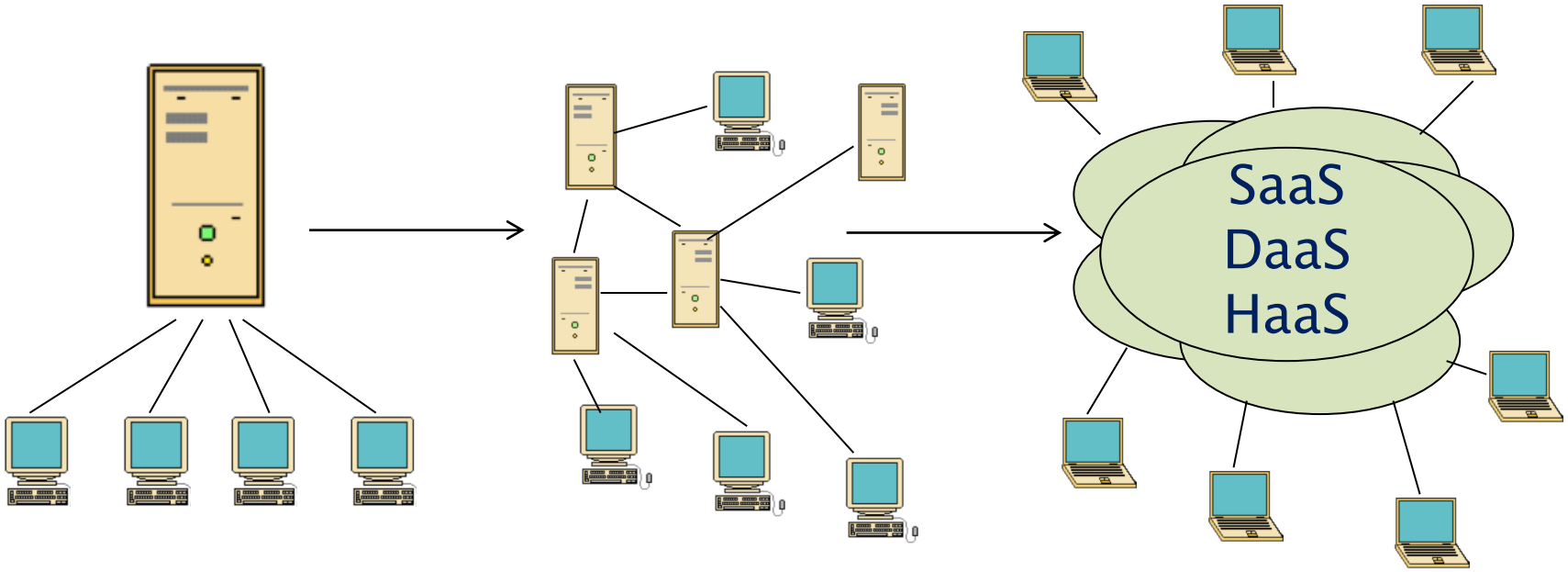
User Interface Integration

Web Mashups

Computing paradigms

"The network is the computer"
(John Gage, Sun Microsystems, 1984)

"The cloud is the computer"



Mainframe computing
1 computer / multiple users

Client-server computing
Computer networks / multiple users

Cloud computing
The Cloud / ALL the users

1965

1985

2005+

The developers' point of view…

Availability on the Web of ready-to-use "building blocks":

- **Software services** (content, functionality) accessible throuhg public Web APIs to build composite applications

- **API: Application Programming Interface**
  a defined set of HTTP request messages, along with a definition of the structure of response messages, which is usually in XML or JSON format [**wikipedia.com**]

Mashups

Mashup: young **integration** practice using the Web as platform.

Some definitions:

"...a mashup is a web application that combines **data** from more than one **source** into a single integrated tool..." *[wikipedia.com]*

"...you can integrate two or more [...] **Web APIs** to create something new and unique, known as a mashup..." *[IBM web site]*

Similar terms: service mashups, data mashups

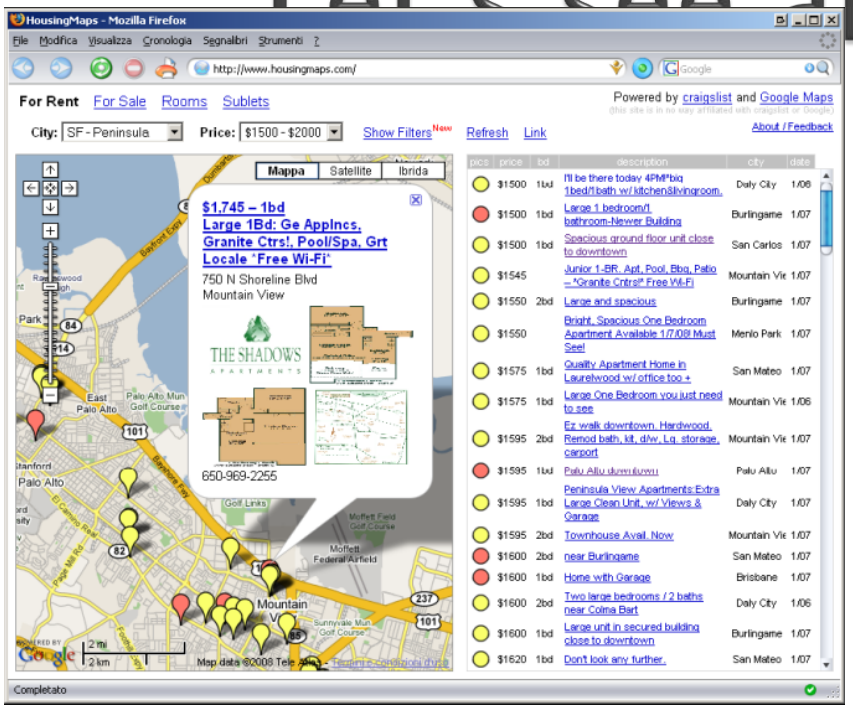
Mashup = Integration in the Web 2.0 way

Highly **user-driven**:

- Oftentimes the actual providers of content/functionality are not even aware of being "wrapped"
- Google Maps example: initially skilled users «hacked» the code of the application

Strong evolution: from hacking to first systematic development approaches in a few years

Let's see an example

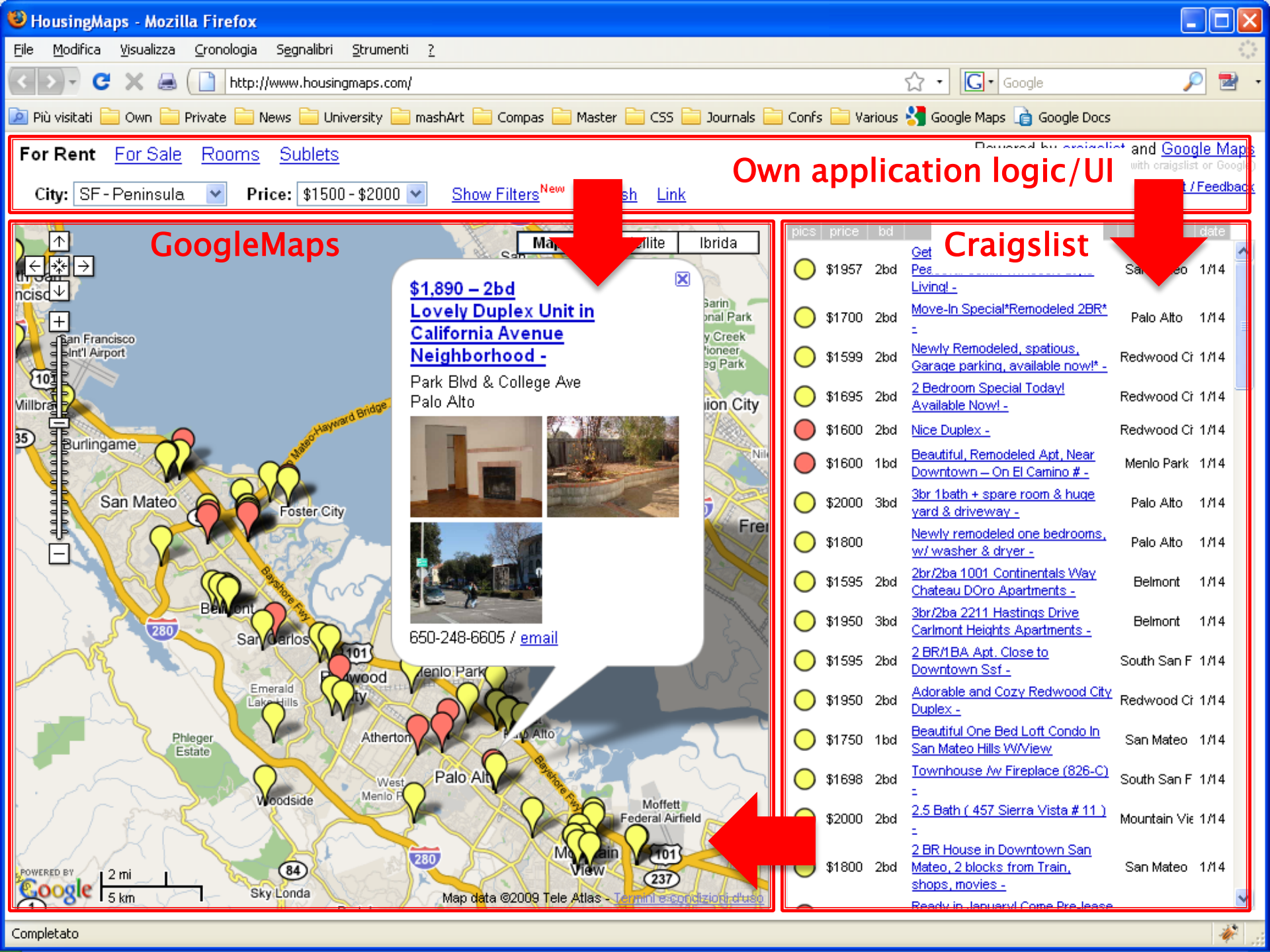


The HousingMaps application (http://www.housingmaps.com)

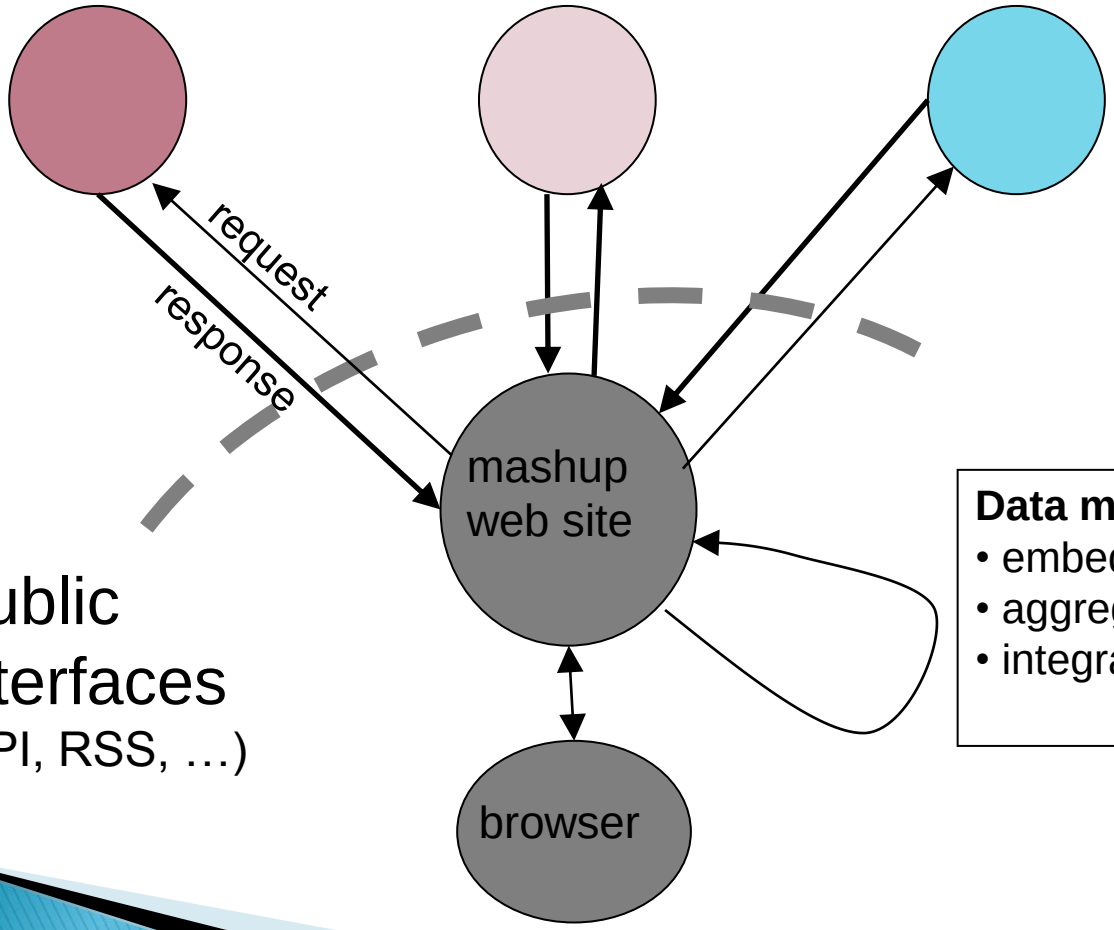
A utility for finding a house for sale or for rent

Composed of:

- Google Maps (http://maps.google.com)
- Craigslist (http://www.craigslist.com)


HousingMaps - Mozilla Firefox
File Modifica Visualizza Cronologia Segnalibri Strumenti ?
http://www.housingmaps.com/
Più visitati Own Private News University mashArt Compas Master CSS Journals Confs Various Google Maps Google Docs
For Rent For Sale Rooms Sublets
City: SF - Peninsula
Price: $1500 - $2000
Show Filters New
Link
and Google Maps
Own application logic/UI
GoogleMaps
Craigslist
pics price bd
$1957 2bd Get Living! -
$1700 2bd Move-In Special*Remodeled 2BR* - Palo Alto 1/14
$1599 2bd Newly Remodeled, spatious, Garage parking, available now!* - Redwood Ci 1/14
$1695 2bd 2 Bedroom Special Today! Available Now! - Redwood Ci 1/14
$1600 2bd Nice Duplex - Redwood Ci 1/14
$1600 1bd Beautiful, Remodeled Apt, Near Downtown -- On El Camino # - Menlo Park 1/14
$2000 3bd 3br 1bath + spare room & huge yard & driveway - Palo Alto 1/14
$1800 Newly remodeled one bedrooms, w/ washer & dryer - Palo Alto 1/14
$1595 2bd 2br/2ba 1001 Continentals Way Chateau DOro Apartments - Belmont 1/14
$1950 3bd 3br/2ba 2211 Hastings Drive Carlmont Heights Apartments - Belmont 1/14
$1595 2bd 2 BR/1BA Apt. Close to Downtown Ssf - South San F 1/14
$1950 2bd Adorable and Cozy Redwood City Duplex - Redwood Ci 1/14
$1750 1bd Beautiful One Bed Loft Condo In San Mateo Hills W/View San Mateo 1/14
$1698 2bd Townhouse /w Fireplace (826-C) - South San F 1/14
$2000 2bd 2.5 Bath (457 Sierra Vista # 11) - Mountain Vie 1/14
$1800 2bd 2 BR House in Downtown San Mateo, 2 blocks from Train, shops, movies - San Mateo 1/14
$1,890 – 2bd Lovely Duplex Unit in California Avenue Neighborhood -
Park Blvd & College Ave
Palo Alto
650-248-6605 / email
Mappa Satellite Ibrida
San Francisco Int'l Airport
Millbrae
Burlingame
San Mateo
Foster City
Belmont
San Carlos
Redwood City
Menlo Park
Atherton
Palo Alto
Woodside
Moffett Federal Airfield
Mountain View
Sky Londa
Emerald Lake Hills
Phleger Estate
West Menlo Park
Bayshore Fwy
San Mateo-Hayward Bridge
101
280
84
237
2 mi
5 km
Map data ©2009 Tele Atlas - Termini e condizioni d'uso
Completato

Mashup architecture



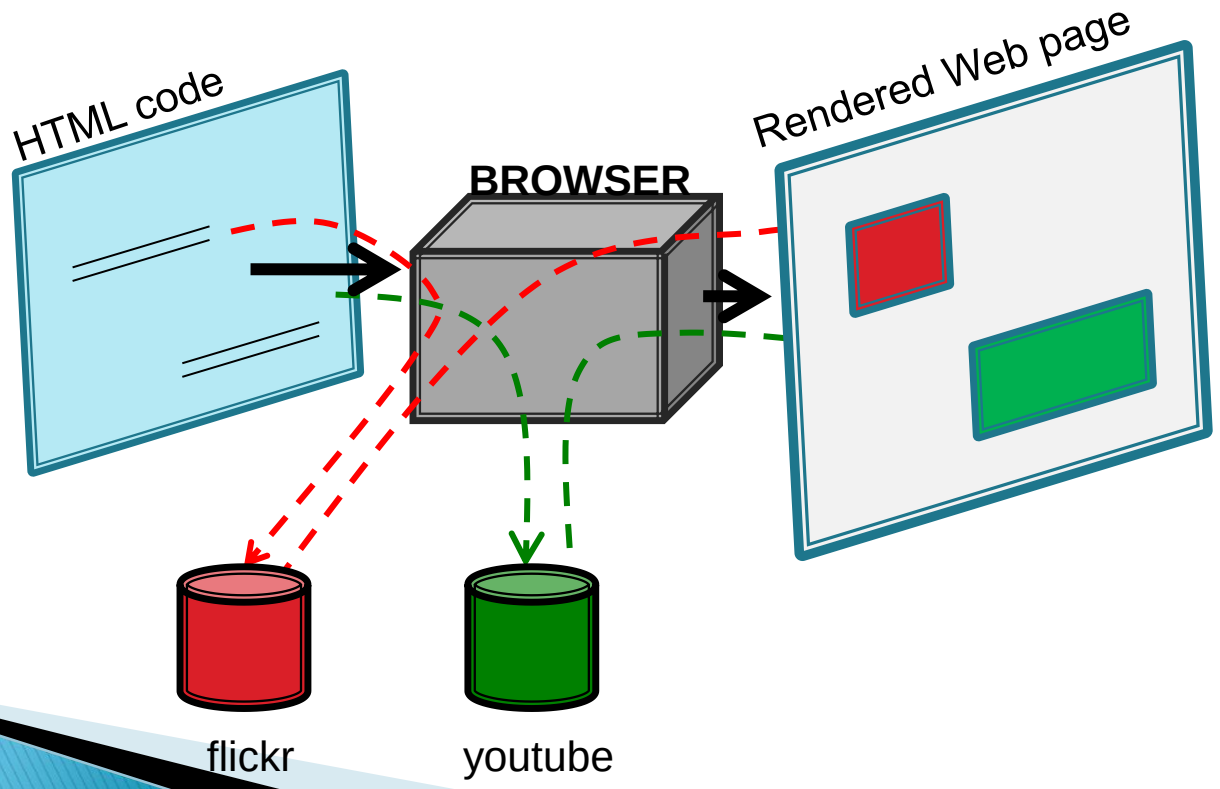
Public interfaces
(API, RSS, …)

sources:
- Videos
- Images
- Maps
- News
- RSS feeds
- Social contents
- …

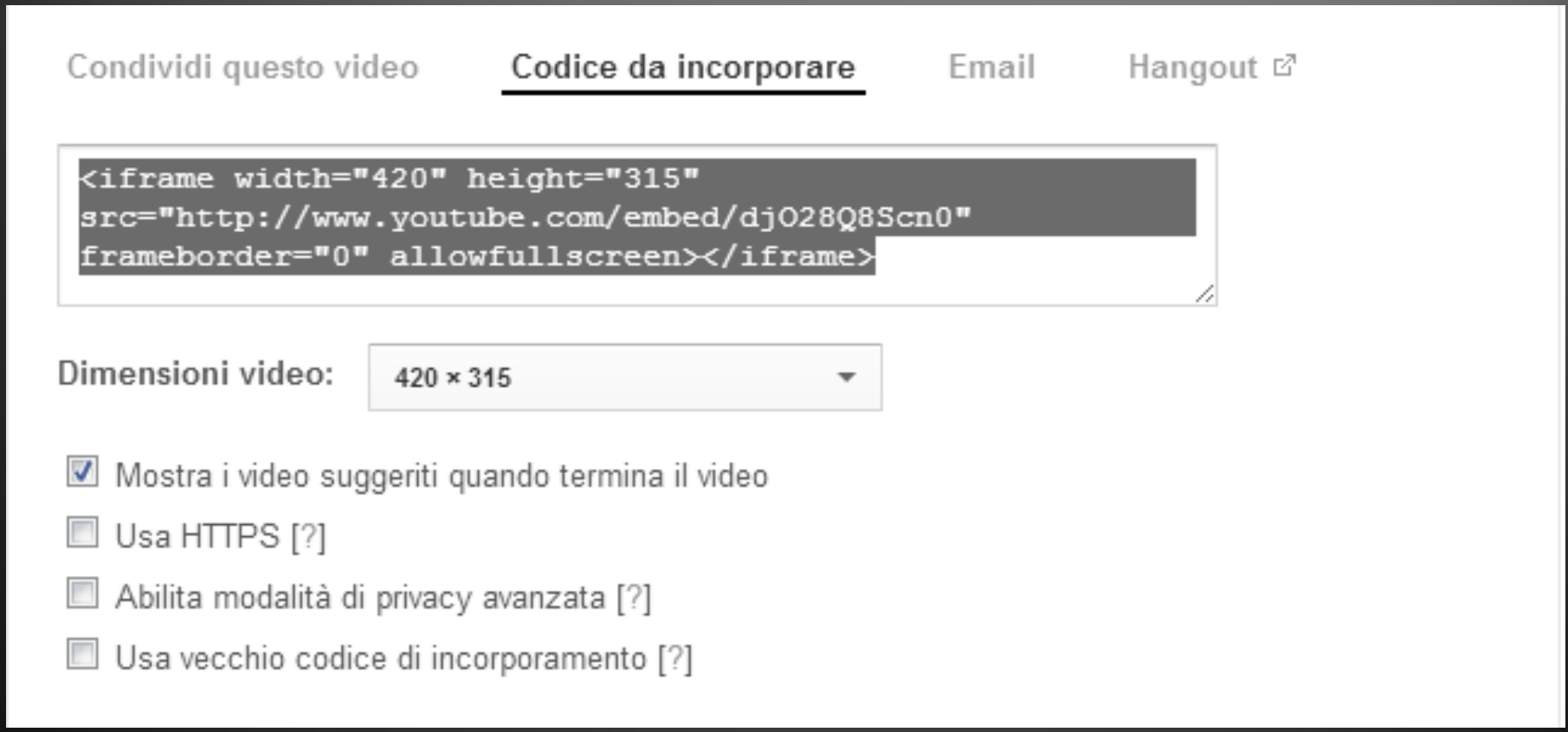
Data manipulation:
- embedding
- aggregation
- integration

The simplest case: Embedding

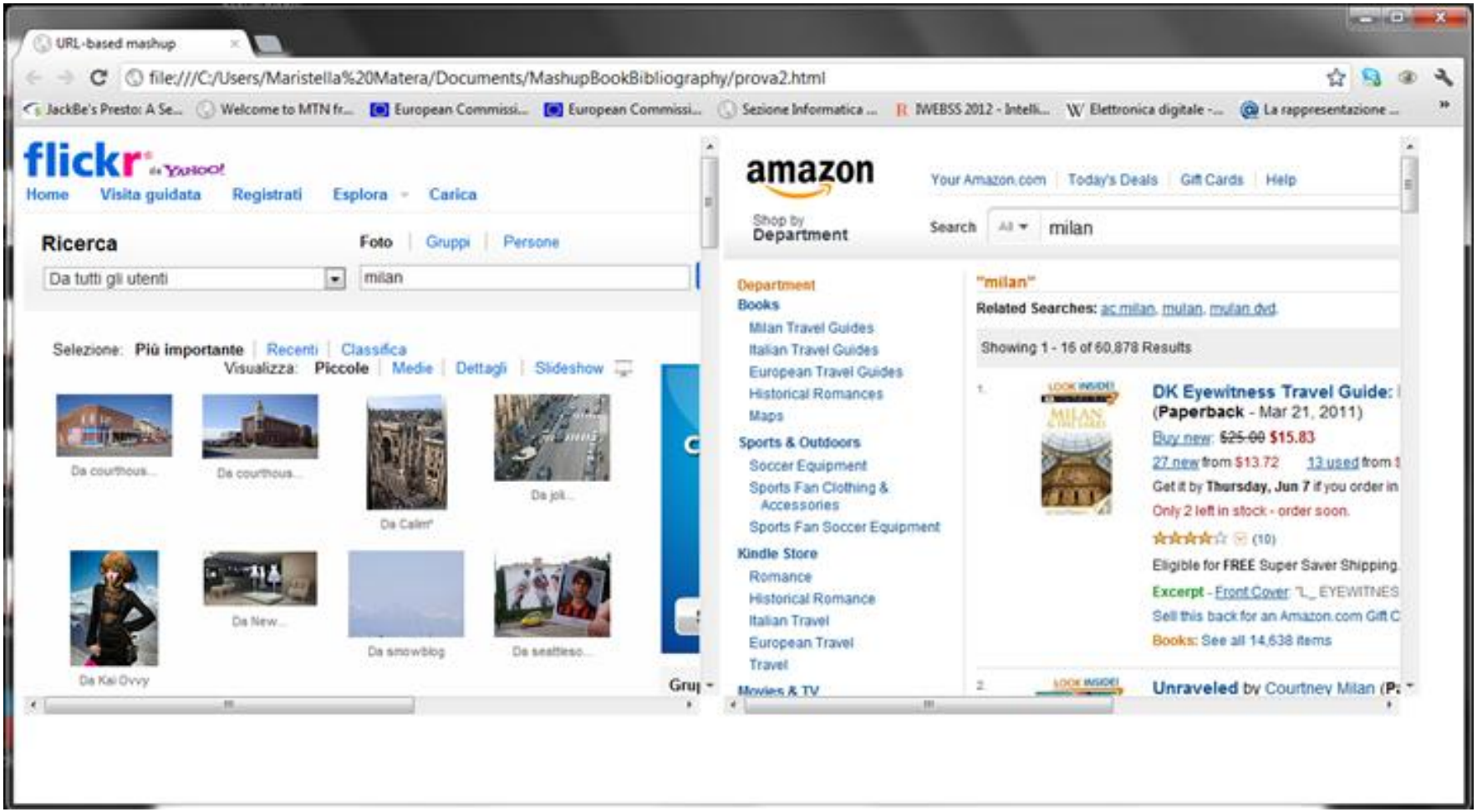
To add a multimedia object in a Web page, it is sufficient to copy an HTML “snippet” into the HTML code of my Web page



Youtube videos



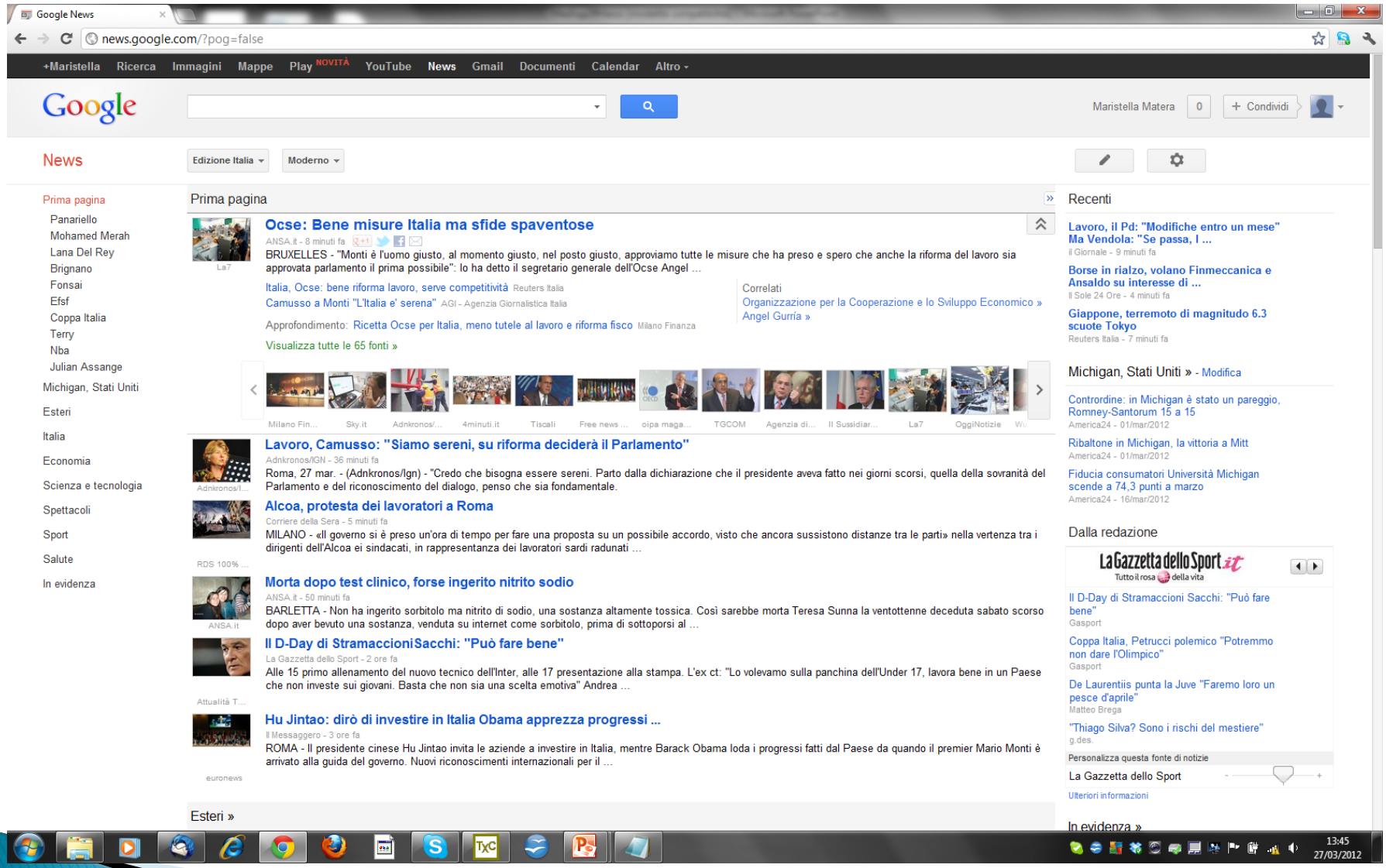
HTML page embedding



HTML embedding

```
<body>

 <iframe id="FlickrFrame"
   src="http://www.flickr.com/search/?q=milan"
   name="Flickr" style="width:600px; height:500px; border:
0px"></iframe>
 <iframe id="AmazonFrame"
   src="http://amazon.com/s/?url=search-alias%3Daps&field-
keywords=milan"
   name="Amazon" style="width:600px; height:500px; border:
0px"></iframe>
 </body>

</html>
```

Content Aggregation

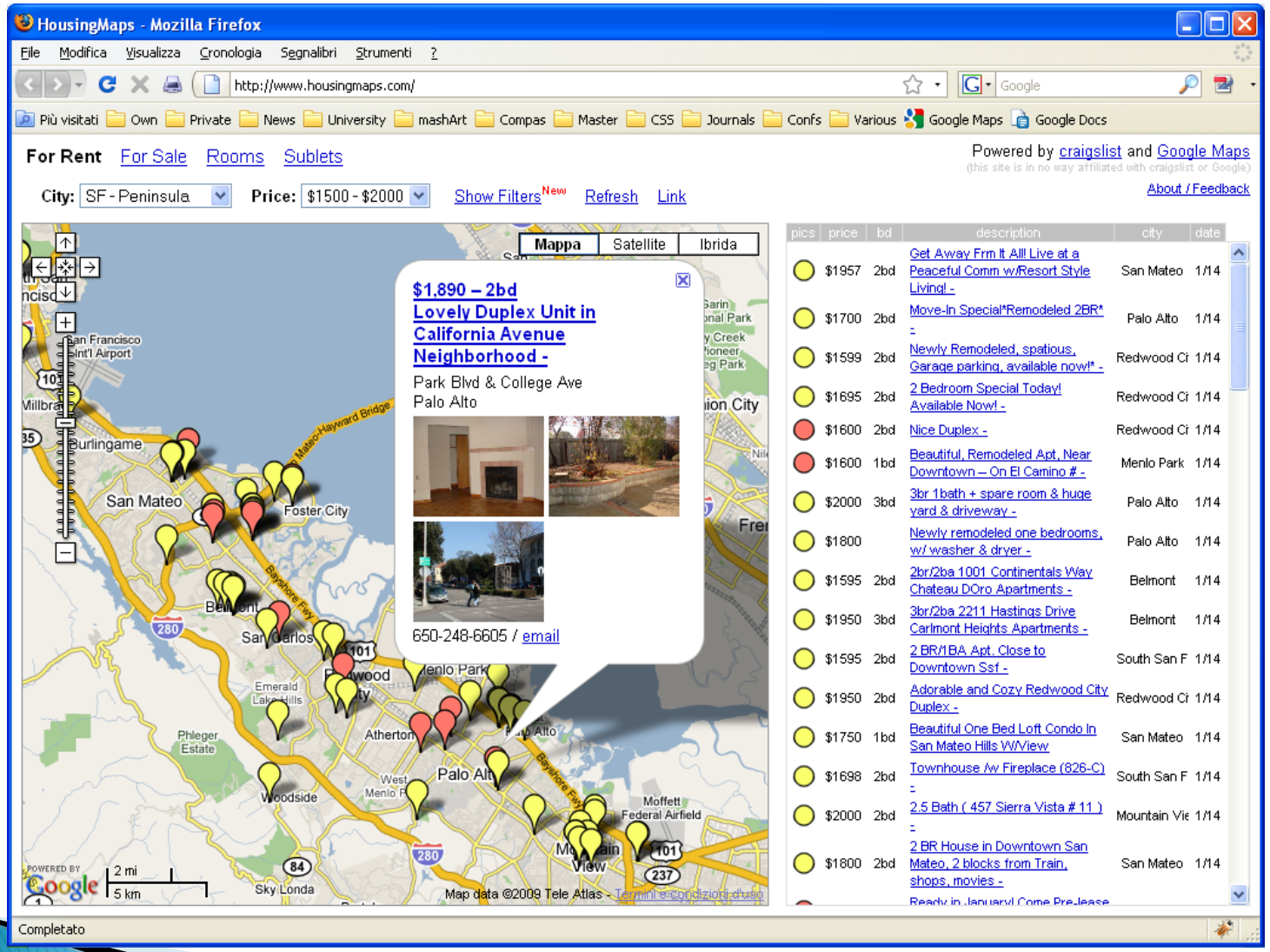


Google News

Visual Aggregation

http://newsmap.jp (aggregates from Google News)

Integration



www.housingmaps.com

Components

- **Collections**:
  - ProgrammableWeb (www.programmabelWeb.com)
  - Mashery (developer.mashery.com/apis)
  - …
- **Ecosystems**: offer software components that are «compatible» and «integrable» to build composite applications
  - WordPress (www.wordpress.org)
    offer a large set of widgets and the possibility to include corresponding plugins into the development workspace
  - Netvibes.com: a portal with a huge number of widgets

Let's build some examples

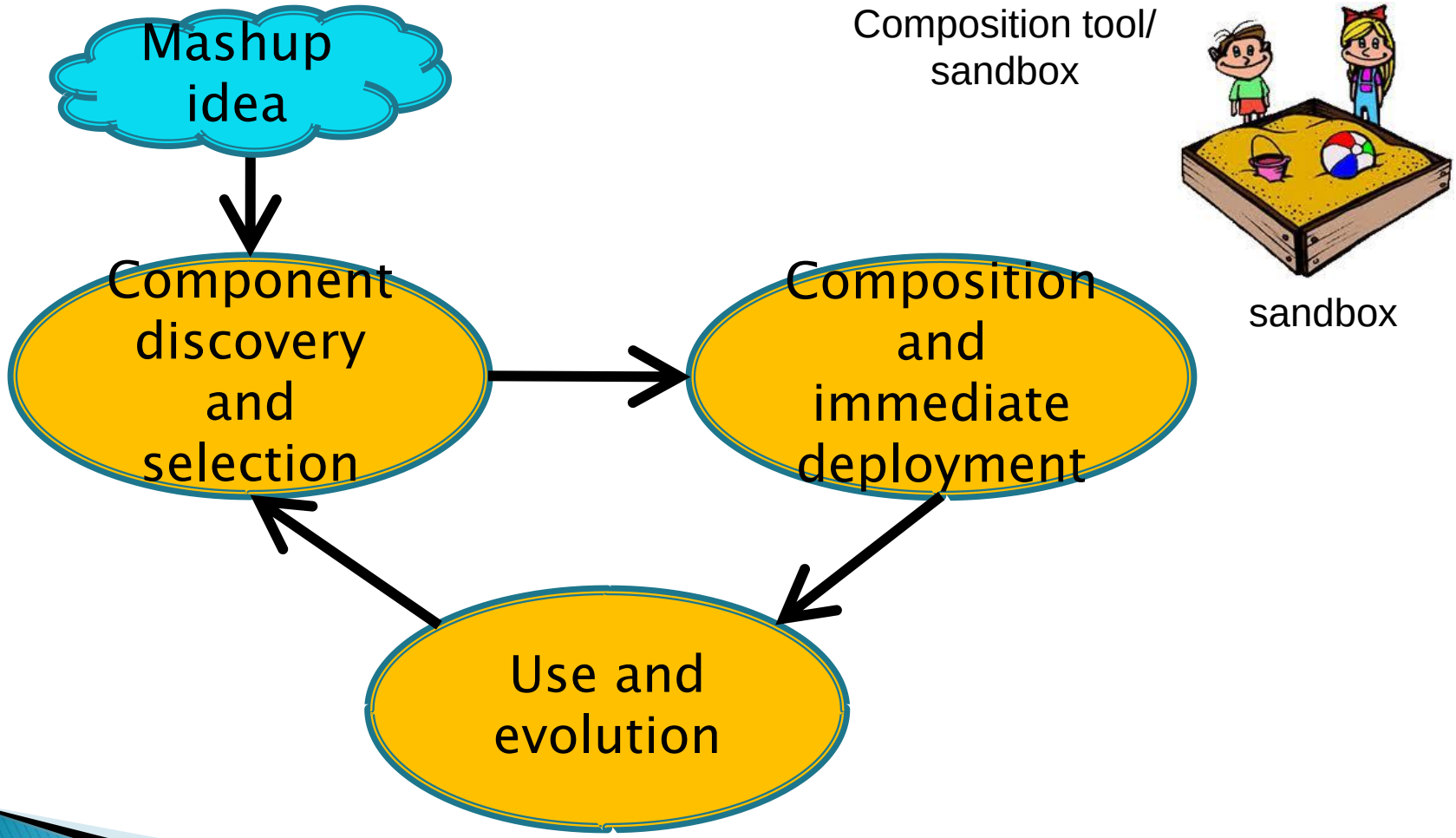
1. Wordpress plugins
2. HTML embedding: **The Expo Mashup**

User-driven innovation

"There are creative people all around the world, hundreds of millions of them, and they are going to think of things to do with our basic platform that we didn't think of"

Vinton Cerf – Google

Lightweight development process



Mashup development manually

Select the components (e.g., GMaps API and the Craigslist RSS):

- **Include** GMaps component
- **Define a layout** for the RSS feed
- Set **markers** through GMaps API

Problems

- Manual development for skilled programmers
- Manual parsing of RSS feed
- No common Web API format

Assisted development

Mashup tools/platforms

- **Simplify** the overall development process, enabling even the **less experienced** user to mash up own applications

Aggregation: widget portals

netvibes.com

Content extraction

dapper from YAHOO!

Like Send 140 people like this. Be the first of your friends.

Get any content from the Web

Search for existing content feeds and web services

e.g. youtube.com, photos, translation

Choose a format (optional)

Search ›

or create a new Dapp

Get more traffic to your site

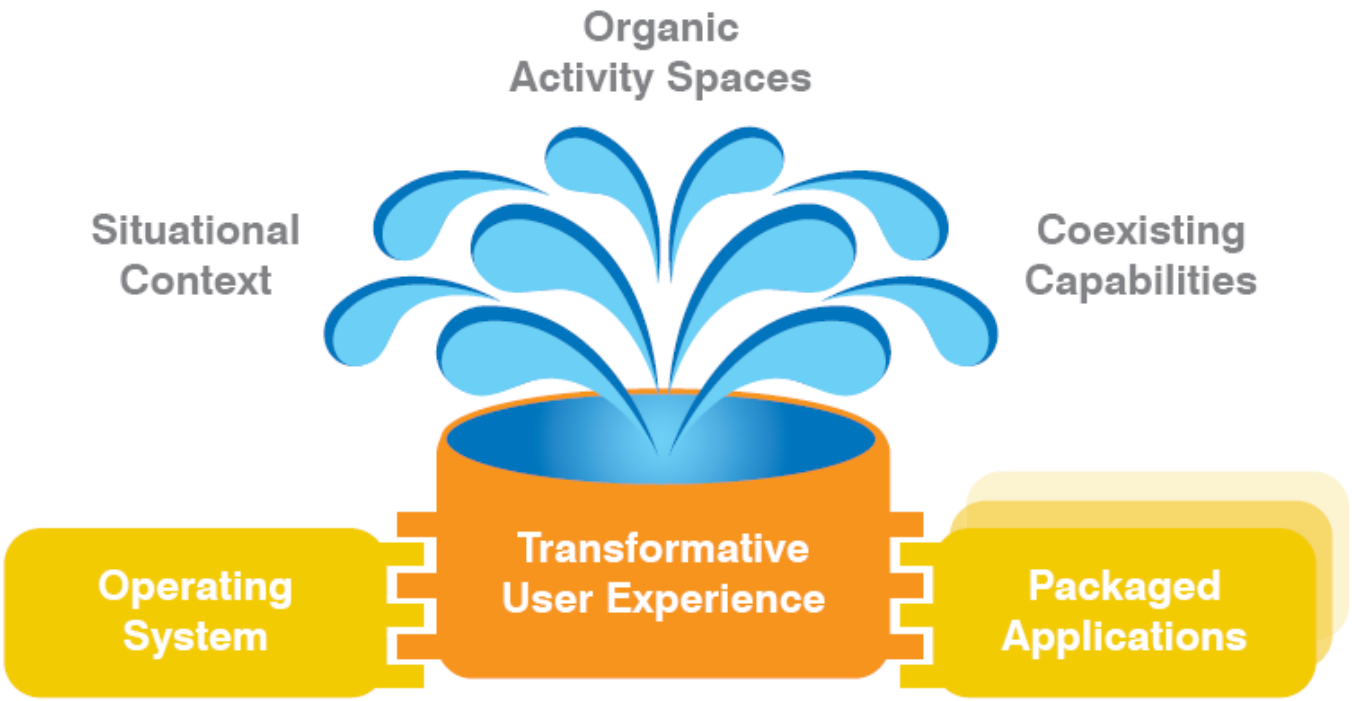
Dapper is a tool that enables users to create update feeds for their favorite sites and website owners to optimize and distribute their content in new ways.

Examples

1. Dapper for content extraction + Netvibes
2. Our tools

How the mashup paradigm can transform UX?

Transformative UX



Chasm between operating systems and packaged applications

Latzina, Beringer. ACM Interaction, March 2012

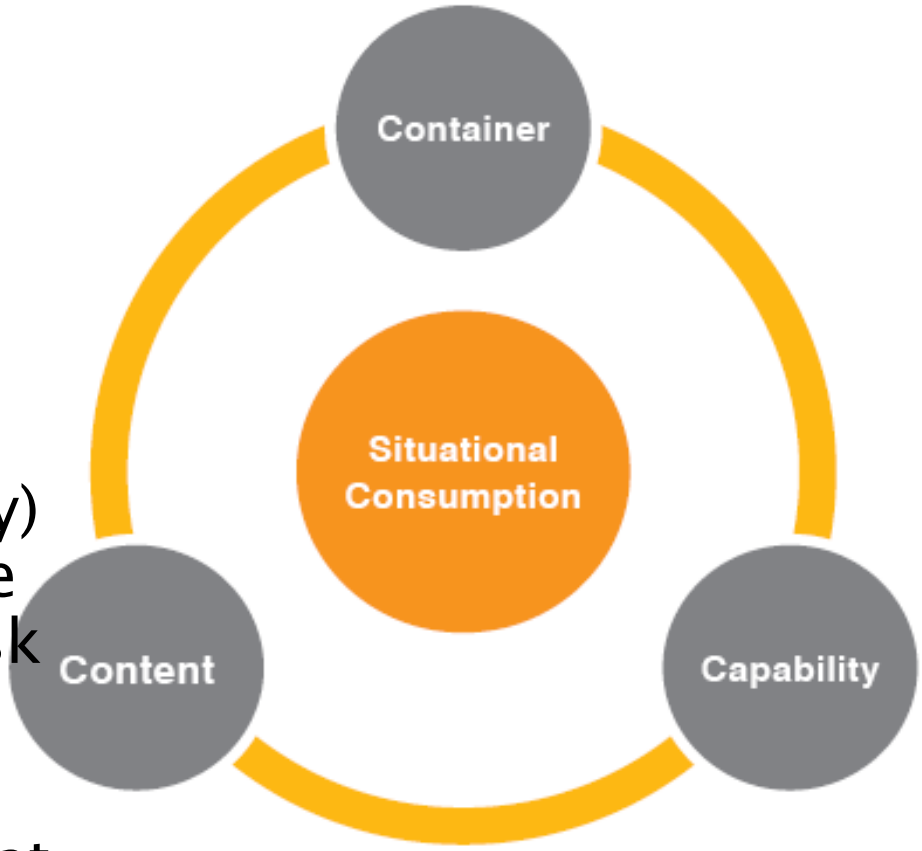
(http://www.sapdesignguild.org/community/readers/reader_latzina_beringer.asp)

Transformative UX

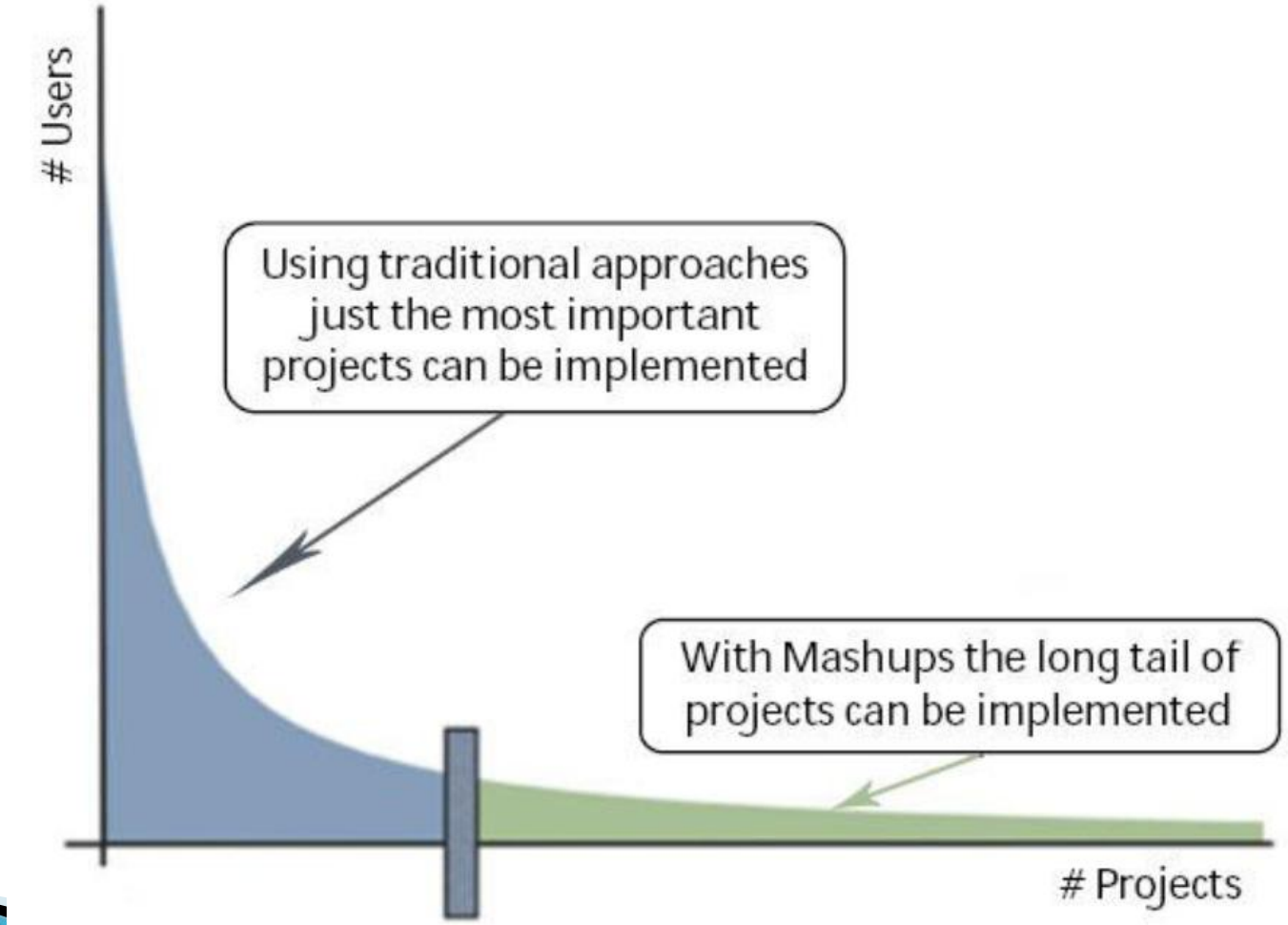
Spaces of interaction where users can realize their **current goals** by moving across **various task contexts**

Elastic systems where data objects (content) and functions (capability) are detached and can be moved into different task contexts (container)

User interfaces **emerge** at runtime



The long tail of the SW market



Web engineering: evolution

Evolution of web application development:

- **Manual** development: static (plain HTML) and dynamic (PHP, JSP,...) pages coded via simple text editors or dedicated tools
- **Tool-aided development**: authoring tools enable developers to reason on the application content and «structure»; the tools automatically generate the code
- **Mashup development**: we are going toward the user-driven development of web applications

Mashup: pros

"Lightweight" applications

reduced amount of code to be written; just the code for integrating APIs

«Lightweight» development

availability of tools who do not require many technical skills – e.g., pipes

Low (o zero) costs for gathering data

Rapid development

Reduced time-to-market, quick prototyping

Mashup: cons

Dependency from the online data sources

data quality, performances, service availbility and reliability, change in the service policy (licensing, acess restrictions, etc.)

APIs: standards e versioning

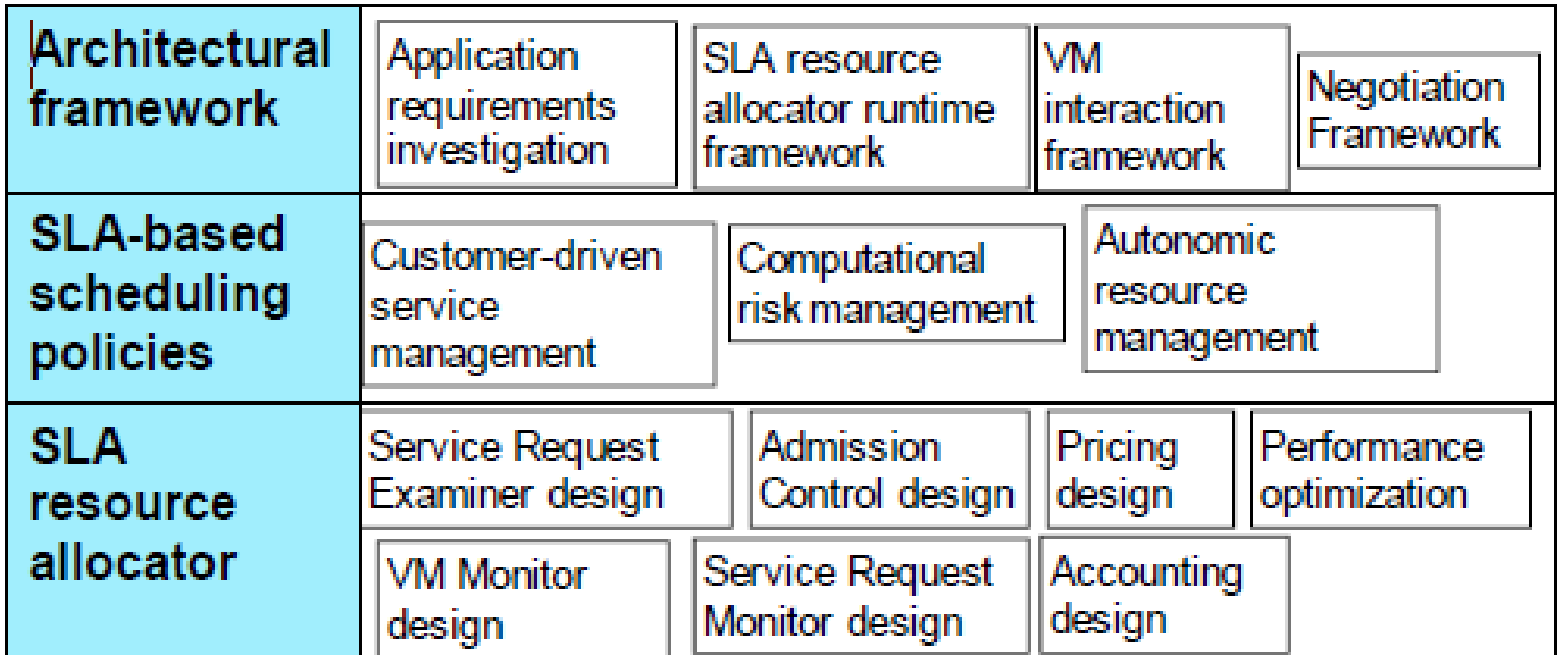
Intellectual property and copyright

“right to remix”: in which measure?

Challenges, Architecture, and Solutions

SLA-Oriented Resource Provisioning for Cloud Computing

Challenges and Requirements – 1



Challenges and Requirements – 2

- Customer-driven Service Management
- Computational Risk Management
- Autonomic Resource Management
- SLA-oriented Resource Allocation Through Virtualization
- Service Benchmarking and Measurement
- System Modeling and Repeatable Evaluation

SLA-Oriented Cloud Computing Vision

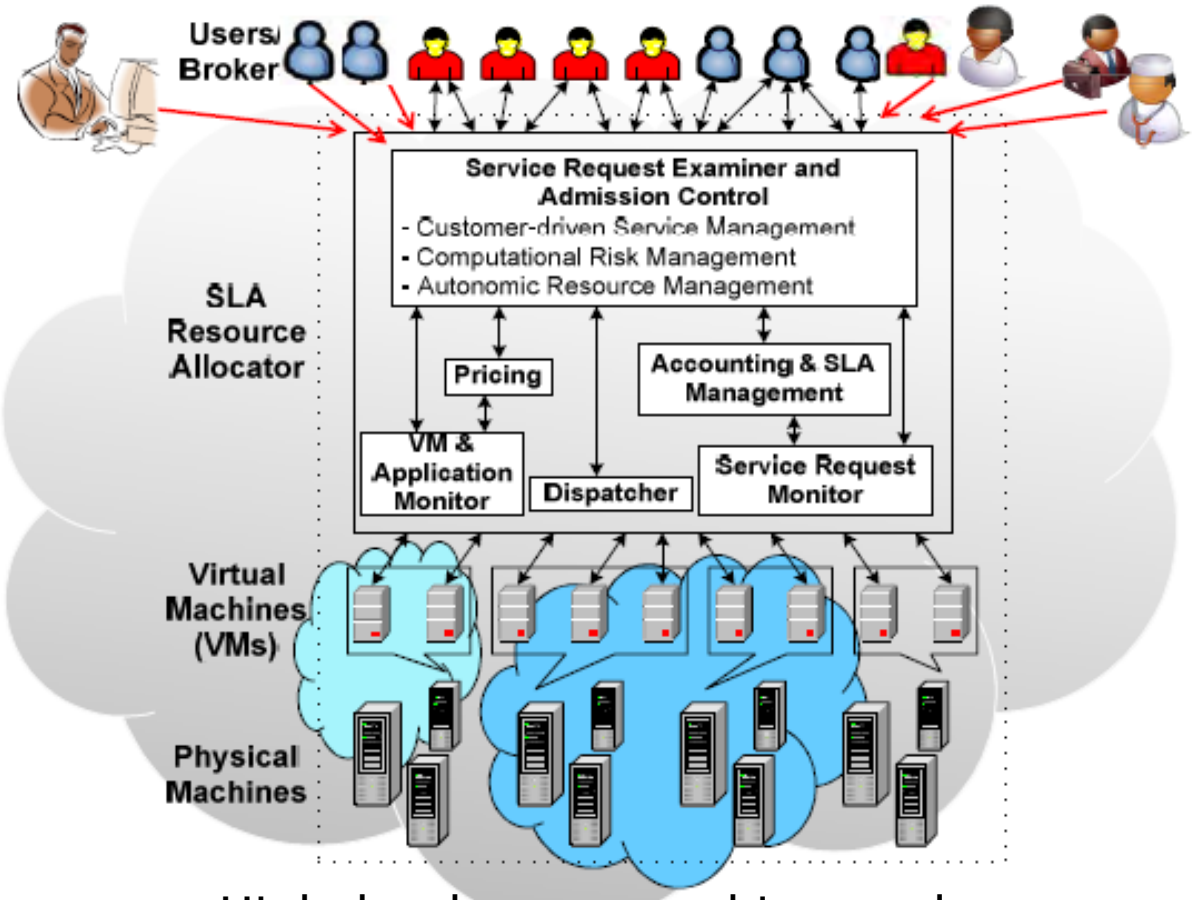
The resource provisioning will be driven by market-oriented principles for efficient resource allocation depending on user QoS targets and workload demand patterns.

- Support for customer-driven service management based on customer profiles and QoS requirements;
- Definition of computational risk management tactics to identify, assess, and manage risks involved in the execution of applications;
- Derivation of appropriate market-based resource management strategies that encompass both customer-driven service management and computational risk management to sustain SLA-oriented resource allocation;
- Incorporation of autonomic resource management models;
- Leverage of Virtual Machine technology to dynamically assign resource shares;
- Implementation of the developed resource management strategies and models into a real computing server;

State-of-the-art

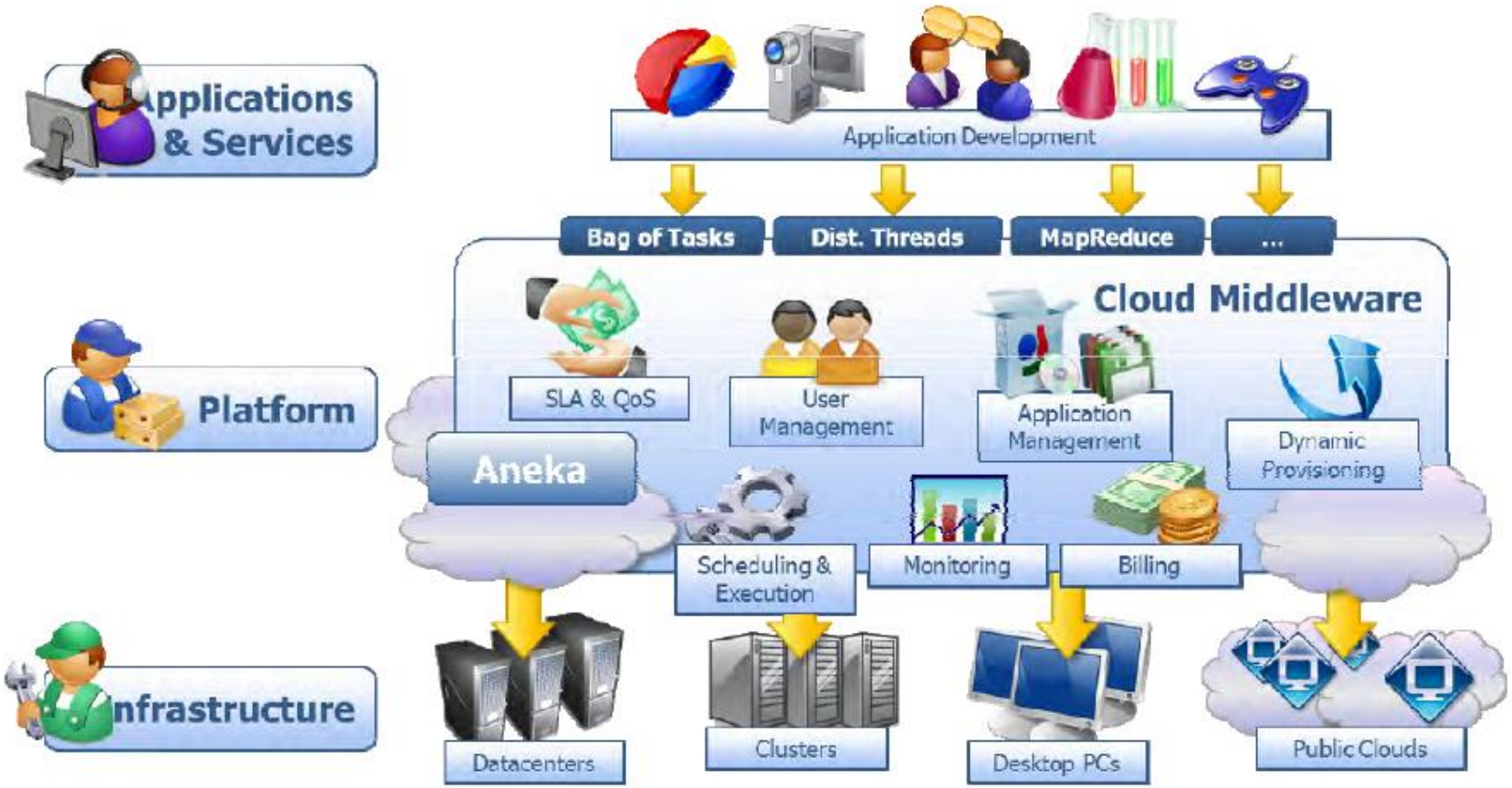
- Traditional Resource Management Systems(Condor, LoadLeveler, Load Sharing Facility, Portable Batch System)
  - adopt system-centric resource allocation approaches that focus on optimizing overall cluster performance
    - Increase processor throughput and utilization for the cluster
    - Reduce the average waiting time and response time for jobs
  - Assume that all job requests are of equal user importance and neglect actual levels of service required by different users.
- Virtual Machine management platform solutions(Eucalyptus, OpenStack, Apache VCL, Citrix Essentials)
  - Main goal is to provide automatic configuration and maintenance of the centers
- Market-based resource management
  - Not considered and incorporated customer-driven service management, computational risk management, and autonomic resource management into market-driven resource management

System Architecture



High-level system architectural framework

SLA Provisioning in Aneka -1



Aneka architecture

SLA Provisioning in Aneka – 2

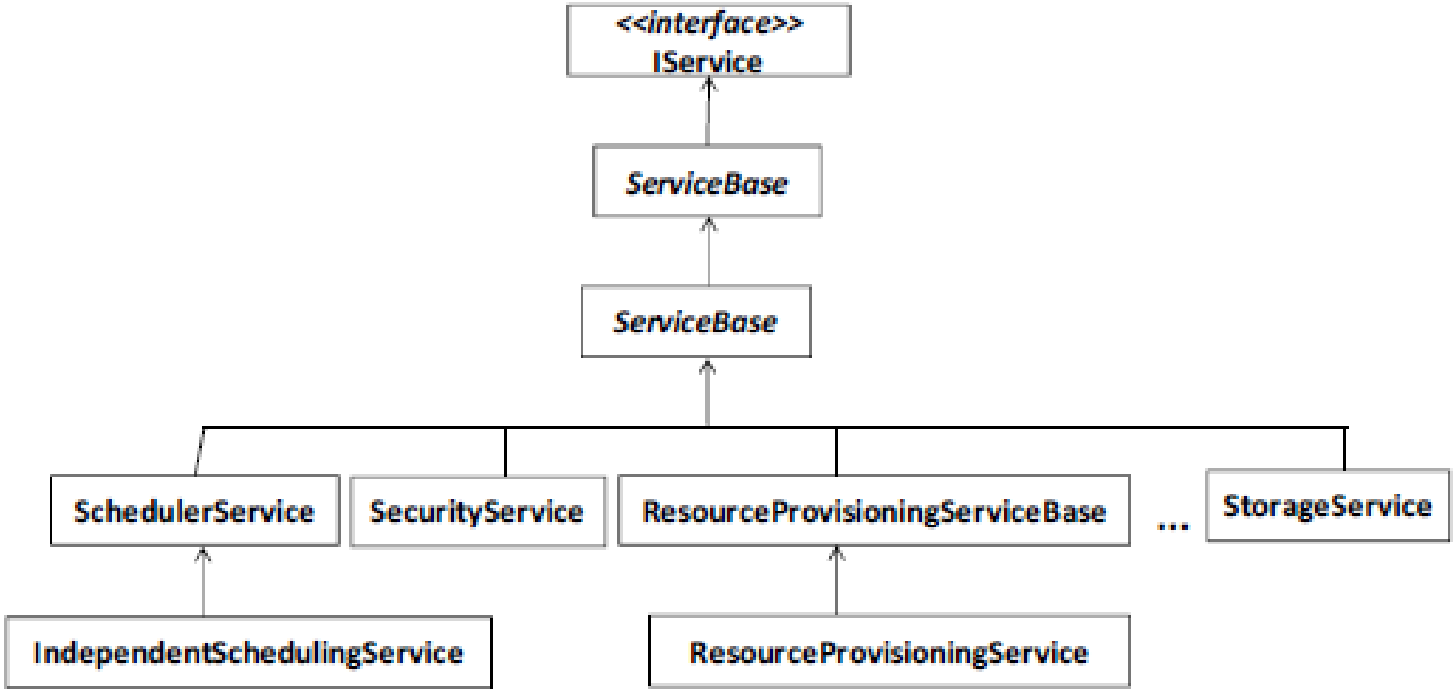


Figure 5. Class diagram of Aneka services.

SLA Provisioning in Aneka – 3

Algortihm 1. SLA-oriented Dynamic Provisioning Algortihm in Aneka.

1. When a task finishes or a new job is received:
   1.1. Updates estimation of task runtime;
   1.2. Defines estimated job completion time with current amount of resources;
   1.3. If completion time > deadline
      1.3.1. Determines number of extra resources required
      1.3.2. Submits a request for resources to the Provisioner.
   Else
      1.3.2. If resources can be released
         1.3.2.1. Submits request for release of resources to the Provisioner

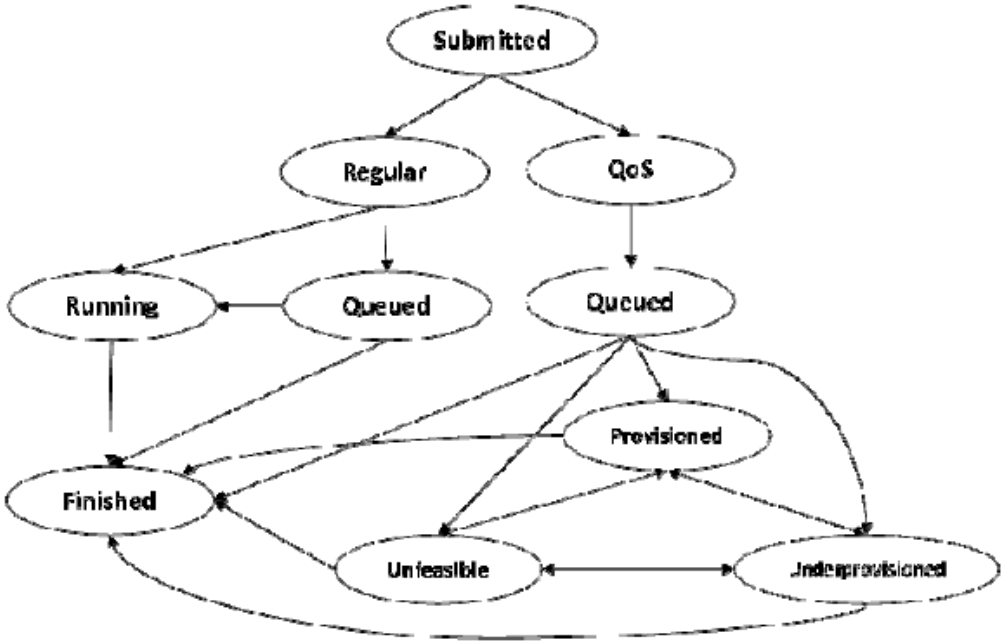


Figure 6. State diagram of jobs in Aneka.

SLA Provisioning in Aneka – 4

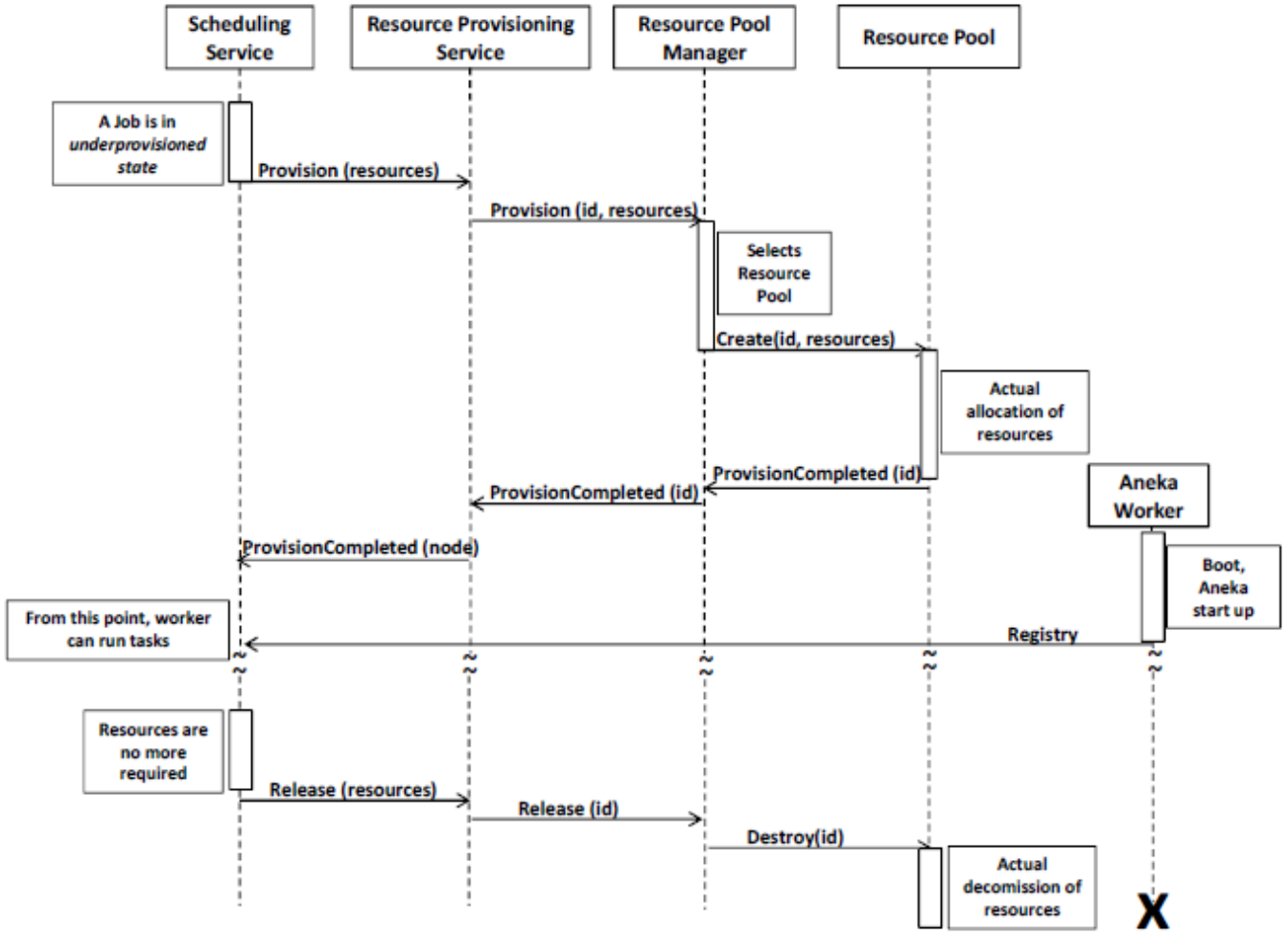


Figure 7. Interaction between Aneka services during dynamic provisioning process.

Performance Evaluation – 1

- Static resource
  - 1 Aneka master – m1.large(7.5GB memory, 4 EC2 compute units, 850GB instance storage, 64bit platform, US0.48 per instance per hour) Windows-based VM
  - 4 Aneka workers – m1.small(1.7GB memory, 1 EC2 compute unit, 160GB instance storage, 32bit platform, US0.085 per instance per hour) Linux-based VM
- Dynamic resources
  - m1.small Linux-based instances

Performance Evaluation – 2

- CPU-intensive application
- SLA is defined in terms of user-defined deadline
- execution time of each task was set to 2 minutes
- Each job consists of 120 tasks

	Static machines	Dynamic machines	Execution Time	Extra cost
No QoS	4	0	1:00:58	0
45min	4	2	0:41:06	U$ 0.17
30 min	4	6	0:28:24	U$ 0.51
15 min	4	20	0:14:18	U$ 1.70