

ANNAI WOMEN'S COLLEGE

PUNNAMCHATHRAM, KARUR - 639 136.

(Affiliated to Bharathidasan University)

DEPARTMENT OF COMPUTER SCIENCE

Class : II M.Sc Computer Science – IV Semester

Subject Title : CLOUD COMPUTING

Subject Code : P16CS41

Staff Name : P.Suseela M.Sc.,,M.Phil.,

Designation : Asst. Professor

UNIT I

INTRODUCTION TO CLOUD COMPUTING

Technologies such as cluster, grid, and now, cloud computing, have all aimed at allowing access to large amounts of computing power in a fully visualized manner, by aggregating resources and offering a single system view. In addition, an important aim of these technologies has been delivering computing as a utility.

Utility computing describes a business model for on-demand delivery of computing power; consumers pay providers based on usage (“pay-as-you-go”), similar to the way in which we currently obtain services from traditional public utility services such as water, electricity, gas, and telephony.

Cloud computing has been coined as an umbrella term to describe

a category of sophisticated on-demand computing services initially offered by commercial providers, such as Amazon, Google, and Microsoft. It denotes a model on which a computing infrastructure is viewed as a “cloud,” from which businesses and individuals access applications from anywhere in the world on demand . The main principle behind this model is offering computing, storage, and software “as a service.”

Buyya have defined it as follows: “Cloud is a parallel and distributed computing system consisting of a collection of interconnected and virtualised computers that are dynamically provisioned and presented as one or more unified computing

resources based on service-level agreements (SLA) established through negotiation between the service provider and consumers.”

While there are countless other definitions, there seems to be common characteristics between the most notable ones listed above, which a cloud should have: (i) pay-per-use (no ongoing commitment, utility prices); (ii) elastic capacity and the illusion of infinite resources; (iii) self-service interface; and (iv) resources that are abstracted or virtualised.

In addition to raw computing and storage, cloud computing providers usually offer a broad range of software services. They also include APIs and development tools that allow developers to build seamlessly scalable applications upon their services. The ultimate goal is allowing customers to run their everyday IT infrastructure “in the cloud.”

ROOTS OF CLOUD COMPUTING

We can track the roots of clouds computing by observing the advancement of several technologies, especially in hardware (virtualization, multi-core chips), Internet technologies (Web services, service-oriented architectures, Web 2.0), distributed computing (clusters, grids), and systems management (autonomic computing, data center automation).

1.From Mainframes to Clouds

We are currently experiencing a switch in the IT world, from in-house generated computing power into utility-supplied computing resources delivered over the Internet as Web services. This trend is similar to what occurred about a century ago when factories, which used to generate their own electric power, realized that it is was cheaper just plugging their machines into the newly formed electric power grid .

Computing delivered as a utility can be defined as “on demand delivery of infrastructure, applications, and business processes in a security-rich, shared, scalable, and based computer environment over the Internet for a fee” .

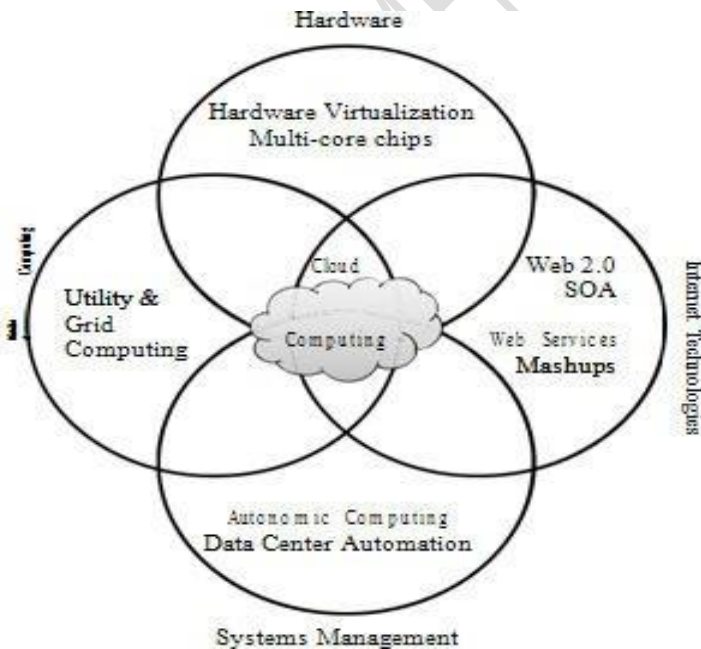


FIGURE : Convergence of various advances leading to the advent of cloud computing.

This model brings benefits to both consumers and providers of IT services. Consumers can attain reduction on IT-related costs by choosing to obtain cheaper services from external providers as opposed to heavily investing on IT infrastructure and personnel hiring.

The “on-demand” component of this model allows consumers to adapt their IT usage to rapidly increasing or unpredictable computing needs.

Providers of IT services achieve better operational costs; hardware and software infrastructures are built to provide multiple solutions and serve many users, thus increasing efficiency and ultimately leading to faster return on investment (ROI) as well as lower total cost of ownership (TCO) .

2.SOA, Web Services, Web 2.0, and Mashups

Web services can glue together applications running on different messaging product plat-forms, enabling information from one application to be made available to others, and enabling internal applications to be made available over the Internet.

WS standards have been created on top of existing ubiquitous technologies such as HTTP and XML, thus providing a common mechanism for delivering services, making them ideal for implementing a service-oriented architecture (SOA).

The purpose of a SOA is to address requirements of loosely coupled, standards-based, and protocol-independent distributed computing. In a SOA, software resources are packaged as “services,” which are well-defined, self-contained modules that provide standard business functionality and are independent of the state or context of other services.

This concept of gluing services initially focused on the enterprise Web, but gained space in the consumer realm as well, especially with the advent of Web 2.0. In the consumer Web, information and services may be programmatically aggregated, acting as building blocks of complex compositions, called service mashups.

Many service providers, such as Amazon, del.icio.us, Facebook, and Google, make their service APIs publicly accessible using standard protocols such as SOAP.

3. Grid Computing

Grid computing enables aggregation of distributed resources and transparently access to them. It seek to share compute and storage resources distributed across different administrative domains, with their main focus being speeding up a broad range of scientific applications, such as climate modeling, drug design, and protein analysis.

A key aspect of the grid vision realization has been building standard Web services-based protocols that allow distributed resources to be “discovered, accessed, allocated, monitored, accounted for, and billed for, etc., and in general managed as a single virtual system.”

The Open Grid Services Archi-tecture (OGSA) addresses this need for standardization by defining a set of core capabilities and behaviors that address key concerns in grid systems.

Virtualization technology has been identified as the perfect fit to issues that have caused frustration when using grids, such as hosting many dissimilar software applications on a single physical platform. In this direction, some research projects (e.g., Globus Virtual Workspaces aimed at evolving grids to support an additional layer to virtualize computation, storage, and network resources.

4. Utility Computing

In utility computing environments, users assign a “utility” value to their jobs, where utility is a fixed or time-varying valuation that captures various QoS constraints (deadline, importance, satisfaction).

The valuation is the amount they are willing to pay a service provider to satisfy their demands. The service providers then attempt to maximize their own utility, where said utility may directly correlate with their profit. Providers can choose to prioritize high yield (i.e., profit per unit of resource) user jobs, leading to a scenario where shared systems are viewed as a marketplace, where users compete for resources based on the perceived utility or value of their jobs.

5. Hardware Virtualization

Cloud computing services are usually backed by large-scale data centers composed of thousands of computers. Such data centers are built to serve many users and host many disparate applications.

For this purpose, hardware virtualization can be considered as a perfect fit to overcome most operational issues of data center building and maintenance.

As depicted in Figure a software layer, the virtual machine monitor (VMM), also called a hypervisor, mediates access to the physical hardware presenting to each guest operating system a virtual machine (VM), which is a set of virtual platform interfaces .

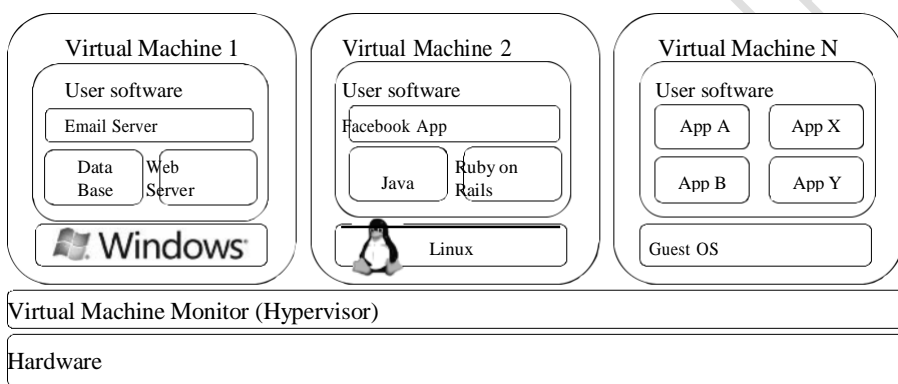


FIGURE: A hardware virtualized server hosting three virtual machines, each one running distinct operating system and user level software stack.

A VM’s state includes a full disk or partition image, configuration files, and an image of its RAM .

6. Virtual Appliances and the Open Virtualization Format

An application combined with the environment needed to run it (operating system, libraries, compilers, databases, application containers, and so forth) is referred to as a “virtual appliance.”

Packaging application environments in the shape of virtual appliances eases software customization, configuration, and patching and improves portability. Most commonly, an appliance is shaped as a VM disk image associated with hardware requirements, and it can be readily deployed in a hypervisor.

On-line marketplaces have been set up to allow the exchange of ready-made appliances containing popular operating systems and useful software combinations, both commercial and open-source.

Most notably, the VMWare virtual appliance marketplace allows users to deploy appliances on VMWare hypervisors or on partners public clouds and Amazon allows developers to share

specialized Amazon Machine Images (AMI) and monetize their usage on Amazon EC2 .

7. Autonomic Computing

The increasing complexity of computing systems has motivated research on autonomic computing, which seeks to improve systems by decreasing human involvement in their operation.

Autonomic, or self-managing, systems rely on monitoring probes and gauges (sensors), on an adaptation engine (autonomic manager) for computing optimizations based on monitoring data, and on effectors to carry out changes on the system.

LAYERS AND TYPES OF CLOUDS

Cloud computing services are divided into three classes, according to the abstraction level of the capability provided and the service model of providers, namely: (1) Infrastructure as a Service, (2) Platform as a Service, and (3) Software as a Service . Figur depicts the layered organization of the cloud stack from physical infrastructure to applications.

A core middleware manages physical resources and the VMs deployed on top of them; in addition, it provides the required features (e.g., accounting and billing) to offer multi-tenant pay-as-you-go services.

Cloud development environments are built on top of infrastructure services to offer application development and deployment capabilities; in this level, various programming models, libraries, APIs, and mashup editors enable the creation of a range of business, Web, and scientific applications. Once deployed in the cloud, these applications can be consumed by end users.

1. Infrastructure as a Service

Offering virtualized resources (computation, storage, and communication) on demand is known as Infrastructure as a Service (IaaS) .

A cloud infrastructure




Service Class	Main Access & Management Tool	Service content
 SaaS	Web Browser	Cloud Applications Social networks, Office suites, CRM, Video processing
 PaaS	Cloud Development Environment	Cloud Platform Programming languages, Frameworks, Mashups editors, Structured data
 IaaS	Virtual Infrastructure Manager	Cloud Infrastructure Compute Servers, Data Storage, Firewall, Load Balancer

FIGURE: The cloud computing stack.

enables on-demand provisioning of servers running several choices of operating systems and a customized software stack. Infrastructure services are considered to be the bottom layer of cloud computing systems [39].

Amazon Web Services mainly offers IaaS, which in the case of its EC2 service means offering VMs with a software stack that can be customized similar to how an ordinary physical server would be customized.

Users are given privileges to perform numerous activities to the server, such as: starting and stopping it, customizing it by installing software packages, attaching virtual disks to it, and configuring access permissions and firewalls rules.

2. Platform as a Service

In addition to infrastructure-oriented clouds that provide raw computing and storage services, another approach is to offer a higher level of abstraction to make a cloud easily programmable, known as Platform as a Service (PaaS). A cloud platform offers an environment on which developers create and deploy applications and do not necessarily need to know how many processors or how much

memory that applications will be using. In addition, multiple programming models and specialized services (e.g., data access, authentication, and payments) are offered as building blocks to new applications .

Google AppEngine, an example of Platform as a Service, offers a scalable environment for developing and hosting Web applications, which should be written in specific programming languages such as Python or Java, and use the services' own proprietary structured object data store.

3. Software as a Service

Applications reside on the top of the cloud stack. Services provided by this layer can be accessed by end users through Web portals. Therefore, consumers are increasingly shifting from locally installed computer programs to on-line software services that offer the same functionality.

Traditional desktop applications such as word processing and spreadsheet can now be accessed as a service in the Web. This model of delivering applications, known as Software as a Service (SaaS)

Salesforce.com, which relies on the SaaS model, offers business productivity applications (CRM) that reside completely on their servers, allowing customers to customize and access applications on demand.

4. Deployment Models

Although cloud computing has emerged mainly from the appearance of public computing utilities, other deployment models, with variations in physical location and distribution, have been adopted. In this sense, regardless of its service class, a cloud can be classified as public, private, community, or hybrid based on model of deployment as shown in Figure.

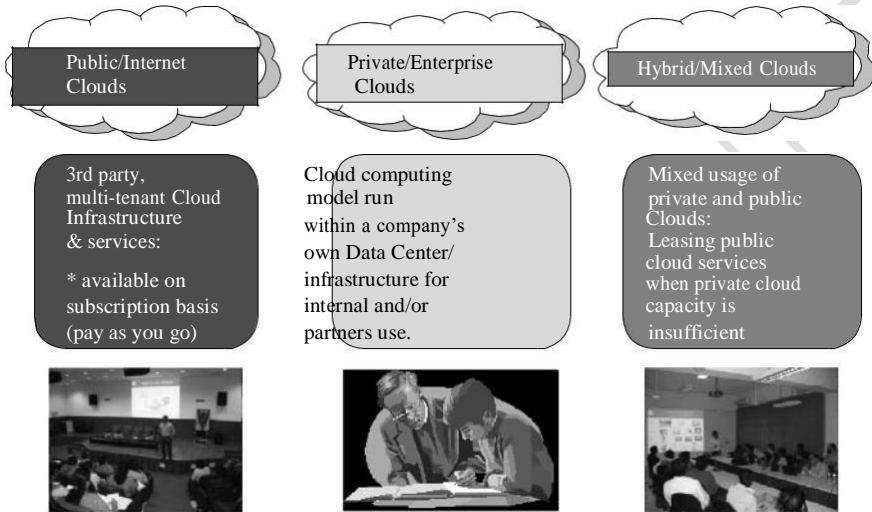


FIGURE . Types of clouds based on deployment models.

community cloud is “shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations) .”

A hybrid cloud takes shape when a private cloud is supplemented with computing capacity from public clouds . The approach of temporarily renting capacity to handle spikes in load is known as “cloud-bursting” .

DESIRED FEATURES OF A CLOUD

Certain features of a cloud are essential to enable services that truly represent the cloud computing model and satisfy expectations of consumers, and cloud offerings must be (i) self-service, (ii) per-usage metered and billed, (iii) elastic, and (iv) customizable.

1. Self-Service

Consumers of cloud computing services expect on-demand, nearly instant access to resources. To support this expectation, clouds must allow self-service access so that customers can request, customize, pay, and use services without intervention of human operators .

2. Per-Usage Metering and Billing

Cloud computing eliminates up-front commitment by users, allowing them to request and use only the necessary amount. Services must be priced on a short-term basis (e.g., by the hour), allowing users to release (and not pay for) resources as soon as they are not needed .

3. Elasticity

Cloud computing gives the illusion of infinite computing resources available on demand. Therefore users expect clouds to rapidly provide resources in any quantity at any time. In particular, it is expected that the additional resources can be (a) provisioned, possibly automatically, when an application load increases and (b) released when load decreases (scale up and down) .

4. Customization

In a multi-tenant cloud a great disparity between user needs is often the case. Thus, resources rented from the cloud must be highly customizable.

CLOUD INFRASTRUCTURE MANAGEMENT

A key challenge IaaS providers face when building a cloud infrastructure is managing physical and virtual resources, namely servers, storage, and net-works, in a holistic fashion .

The software toolkit responsible for this orchestration is called a virtual infrastructure manager (VIM) . This type of software resembles a traditional operating system—but instead of dealing with a single computer, it aggregates resources from multiple computers, presenting a uniform view to user and applications.

The first category—cloud toolkits—includes those that “expose a remote and secure interface for creating, controlling and monitoring virtualize resources,” but do not specialize in VI management.

Tools in the second category—the virtual infrastructure managers—provide advanced features such as automatic load balancing and server consolidation, but do not expose remote cloud-like interfaces.

The availability of a remote cloud-like interface and the ability of managing many users and their permissions are the primary features that would distinguish “cloud toolkits” from “VIMs.”

Virtually all VIMs we investigated present a set of basic features related to managing the life cycle of VMs, including networking groups of VMs together and setting up virtual disks for VMs.

1.Features

We now present a list of both basic and advanced features that are usually available in VIMs.

Virtualization Support. The multi-tenancy aspect of clouds requires multiple customers with disparate requirements to be served by a single hardware infrastructure. Virtualized resources (CPUs, memory, etc.) can be sized and resized with certain flexibility.

Self-Service, On-Demand Resource Provisioning. Self-service access to resources has been perceived as one the most attractive features of clouds.

This feature enables users to directly obtain services from clouds, such as spawning the creation of a server and tailoring its software, configurations, and security policies, without interacting with a human system administrator. This cap-ability “eliminates the need for more time-consuming, labor-intensive, human-driven procurement processes familiar to many in IT” .

Multiple Backend Hypervisors. Different virtualization models and tools offer different benefits, drawbacks, and limitations. Thus, some VI managers provide a uniform management layer regardless of the virtualization technology used. This characteristic is more visible in open-source VI managers, which usually provide pluggable drivers to interact with multiple hypervisors .

Storage Virtualization. Virtualizing storage means abstracting logical storage from physical storage. By consolidating all available storage devices in a data center, it allows creating virtual disks independent from device and location. Storage devices are commonly organized in a storage area network (SAN) and attached to servers via protocols such as Fibre Channel.

Interface to Public Clouds. Researchers have perceived that extending the capacity of a local in-house computing infrastructure by borrowing resources from public clouds is advantageous.

Virtual Networking. Virtual networks allow creating an isolated network on top of a physical infrastructure independently from physical topology and locations .

A virtual LAN (VLAN) allows isolating traffic that shares a switched network, allowing VMs to be grouped into the same broadcast domain.

Support for creating and configuring virtual networks to group VMs placed throughout a data center is provided by most VI managers. Additionally, VI managers that interface with public clouds often support secure VPNs connecting local and remote VMs.

Dynamic Resource Allocation. Increased awareness of energy consumption in data centers has encouraged the practice of dynamic consolidating VMs in a fewer number of servers. In cloud infrastructures, where applications have variable and dynamic needs, capacity management and demand prediction are especially complicated. This fact triggers the need for dynamic resource allocation aiming at obtaining a timely match of supply and demand .

Energy consumption reduction and better management of SLAs can be achieved by dynamically remapping VMs to physical machines at regular intervals. Machines that are not assigned any VM can be turned off or put on a low power state. In the same fashion, overheating can be avoided by moving load away from hotspots .

Virtual Clusters. Several VI managers can holistically manage groups of VMs. This feature is useful for provisioning computing virtual clusters on demand, and interconnected VMs for multi-tier Internet applications [53].

Reservation and Negotiation Mechanism. When users request computational resources to available at a specific time, requests are

termed advance reservations (AR), in contrast to best-effort requests, when users request resources whenever available .

Additionally, leases may be negotiated and renegotiated, allowing provider and consumer to modify a lease or present counter proposals until an agreement is reached.

High Availability and Data Recovery. The high availability (HA) feature of VI managers aims at minimizing application downtime and preventing business disruption.

The HA solution monitors failures of system components such as servers, VMs, disks, and network and ensures that a duplicate VM serves the application in case of failures .

Data backup in clouds should take into account the high data volume involved in VM management. workload is often assigned to proxies, thus offloading production server and reducing network overhead .

INFRASTRUCTURE AS A SERVICE PROVIDERS

Public Infrastructure as a Service providers commonly offer virtual servers containing one or more CPUs, running several choices of operating systems and a customized software stack. In addition, storage space and communication facilities are often provided.

1.Features

IaaS offerings can be distinguished by the availability of specialized features that influence the cost benefit ratio to be experienced by user applications when moved to the cloud. The most relevant features are: (i) geographic distribution of data centers; (ii) variety of user interfaces and APIs to access the system; (iii) specialized components and services that aid particular applications (e.g., load-balancers, firewalls); (iv) choice of virtualization platform and operating systems; and (v) different billing methods and period (e.g., prepaid vs. post-paid, hourly vs. monthly).

Geographic Presence. To improve availability and responsiveness, a provider of worldwide services would typically build several data centers distributed around the world. For example, Amazon Web Services .

User Interfaces and Access to Servers. Ideally, a public IaaS provider must provide multiple access means to its cloud, thus catering for various users and their preferences. Different types of user interfaces (UI) provide different levels of abstraction, the most

common being graphical user interfaces (GUI), command-line tools (CLI), and Web service (WS) APIs.

GUIs are preferred by end users who need to launch, customize, and monitor a few virtual servers and do not necessary need to repeat the process several times.

Advance Reservation of Capacity. Advance reservations allow users to request for an IaaS provider to reserve resources for a specific time frame in the future, thus ensuring that cloud resources will be available at that time.

Amazon Reserved Instances is a form of advance reservation of capacity, allowing users to pay a fixed amount of money in advance to guarantee resource availability at anytime during an agreed period and then paying a discounted hourly rate when resources are in use.

Automatic Scaling and Load Balancing. Applications often need to scale up and down to meet varying load conditions. Automatic scaling is a highly desirable feature of IaaS clouds. It allow users to set conditions for when they want their applications to scale up and down, based on application-specific metrics such as transactions per second, number of simultaneous users, request latency, and so forth.

When the number of virtual servers is increased by automatic scaling, incoming traffic must be automatically distributed among the available servers.

Service-Level Agreement. Service-level agreements (SLAs) are offered by IaaS providers to express their commitment to delivery of a certain QoS. To customers it serves as a warranty.

An SLA usually include availability and performance guarantees. Additionally, metrics must be agreed upon by all parties as well as penalties for violating these expectations.

Hypervisor and Operating System Choice. Traditionally, IaaS offerings have been based on heavily customized open-source Xen deployments. IaaS providers needed expertise in Linux, networking, virtualization, metering, resource management, and many other low-level aspects to successfully deploy and maintain their cloud offerings.

PLATFORM AS A SERVICE PROVIDERS

Public Platform as a Service providers commonly offer a development and deployment environment that allow users to create and run their applications with little or no concern to low-level details of the platform.

1.Features

Programming Models,Languages, and Frameworks.

Programming models made available by IaaS providers define how users can express their applications using higher levels of abstraction and efficiently run them on the cloud platform. Each model aims at efficiently solving a particular problem.

For user convenience, PaaS providers usually support multiple programming languages. Most commonly used languages in platforms include Python and Java (e.g., Google AppEngine), .NET languages (e.g., Microsoft Azure), and Ruby (e.g., Heroku). Force.com has devised its own programming language (Apex) and an Excel-like query language, which provide higher levels of abstraction to key platform functionalities.

Persistence Options. A persistence layer is essential to allow applications to record their state and recover it in case of crashes, as well as to store user data.

Traditionally, Web and enterprise application developers have chosen relational databases as the preferred persistence method. These databases offer fast and reliable structured data storage and transaction processing, but may lack scalability to handle several petabytes of data stored in commodity computers .

In the cloud computing domain, distributed storage technologies have emerged, which seek to be robust and highly scalable, at the expense of relational structure and convenient query languages. For example, Amazon SimpleDB and Google AppEngine datastore offer schema-less, automatically indexed database services .

CHALLENGES AND RISKS

Providers, developers, and end users must consider these challenges and risks to take good advantage of cloud computing. Issues to be faced include user privacy, data security, data lock-in, availability of service, disaster recovery, performance, scalability, energy-efficiency, and programmability.

1.Security, Privacy, and Trust

Information security as a main issue: “current cloud offerings are essentially public exposing the system to more attacks.” For this reason there are potentially additional challenges to make cloud computing environments as secure as in-house IT systems. At the same time, existing, well-understood technologies can be leveraged, such as data encryption, VLANs, and firewalls.

Security and privacy affect the entire cloud computing stack, since there is a massive use of third-party services and infrastructures that are used to host important data or to perform critical operations.

. When data are moved into the Cloud, providers may choose to locate them anywhere on the planet. The physical location of data centers determines the set of laws that can be applied to the management of data. For example, specific cryptography techniques could not be used because they are not allowed in some countries. Similarly, country laws can impose that sensitive data, such as patient health records, are to be stored within national borders.

2.Data Lock-In and Standardization

A major concern of cloud computing users is about having their data locked-in by a certain provider. Users may want to move data and applications out from a provider that does not meet their requirements. However, in their current form, cloud computing infrastructures and platforms do not employ standard methods of storing user data and applications. Consequently, they do not interoperate and user data are not portable.

The answer to this concern is standardization. The Cloud Computing Interoperability Forum (CCIF) was formed by organizations such as Intel, Sun, and Cisco in order to “enable a global cloud computing ecosystem whereby organizations are able to seamlessly work together for the purposes for wider industry adoption of cloud computing technology.” The development of the Unified Cloud Interface (UCI) by CCIF aims at creating a standard programmatic point of access to an entire cloud infrastructure.

3.Availability, Fault-Tolerance, and Disaster Recovery

It is expected that users will have certain expectations about the service level to be provided once their applications are moved to the cloud. These expectations include availability of the service, its overall performance, and what measures are to be taken when something goes wrong in the system or its components. In summary,

users seek for a warranty before they can comfortably move their business to the cloud.

SLAs, which include QoS requirements, must be ideally set up between customers and cloud computing providers to act as warranty.

4.Resource Management and Energy-Efficiency

One important challenge faced by providers of cloud computing services is the efficient management of virtualized resource pools. Physical resources such as CPU cores, disk space, and network bandwidth must be sliced and shared among virtual machines running potentially heterogeneous workloads.

Dimensions to be considered include: number of CPUs, amount of memory, size of virtual disks, and network bandwidth.

Another challenge concerns the outstanding amount of data to be managed in various VM management activities. Such data amount is a result of particular abilities of virtual machines, including the ability of traveling through space (i.e., migration) and time (i.e., checkpointing and rewinding) ,operations that may be required in load balancing, backup, and recovery scenarios.

BROAD APPROACHES TO MIGRATING INTO THE CLOUD

1. Why Migrate?

There are economic and business reasons why an enterprise application can be migrated into the cloud, and there are also a number of technological reasons. Many of these efforts come up as initiatives in adoption of cloud technologies in the enterprise, resulting in integration of enterprise applications running off the captive data centers with the new ones that have been developed on the cloud. Adoption of or integration with cloud computing services is a use case of migration.

In brief, migration can happen at one of the five levels of application, code, design, architecture, and usage.

With due simplification, the migration of an enterprise application is best captured by the following:

$$P-P_C^0 \ 1 \ P_1^0 - P_{OFC}^0 \ 1 \ P_1^0$$

where P is the application before migration running in captive data center, P_C^0 is the application part after migration either into a (hybrid) cloud, P_1^0 is the part of application being run in the captive local data center, and P_{OFC}^0 is the application part optimized for cloud.

If an enterprise application cannot be migrated fully, it could result in some parts being run on the captive local data center while the rest are being migrated into the cloud—essentially a case of a hybrid cloud usage. However, when the entire application is migrated onto the cloud, then P_1^0 is null. Indeed, the migration of the enterprise application P can happen at the five levels of application, code, design, architecture, and usage. It can be that the P_C^0 migration happens at any of the five levels without any P_1^0 component. Many of these best practices are specialized at the level of the components of an enterprise application—like migrating application servers or the enterprise databases.

Cloudonomics. Invariably, migrating into the cloud is driven by economic reasons of cost cutting in both the IT capital expenses (Capex) as well as operational expenses (Opex). There are both the short-term benefits of opportunistic migration to offset seasonal and highly variable IT loads as well as the long-term benefits to leverage the cloud. For the long-term sustained usage, as of 2009, several impediments and shortcomings of the cloud computing services need to be addressed.

At the core of the cloudonomics, as articulated in Ambrust et al. [2], is the expression of when a migration can be economically feasible or tenable. If the average costs of using an enterprise application on a cloud is substantially lower than the costs of using it in one's captive data center and if the cost of migration does not add to the burden on ROI, then the case for migration into the cloud is strong.

2. Deciding on the Cloud Migration

In fact, several proof of concepts and prototypes of the enterprise application are experimented on the cloud to take help in making a sound decision on migrating into the cloud.

Post migration, the ROI on the migration should be positive for a broad range of pricing variability. Arriving at a decision for

undertaking migration demands that either the compelling factors be clearly understood or the pragmatic approach of consulting a group of experts be constituted.

THE SEVEN-STEP MODEL OF MIGRATION INTO A CLOUD

Cloud migration assessments comprise assessments to understand the issues involved in the specific case of migration at the application level or the code, the design, the architecture, or usage levels. In addition, migration assessments are done for the tools being used, the test cases as well as configurations, functionalities, and NFRs of the enterprise application. This results in a meaningful formulation of a comprehensive migration strategy.

The first step of the iterative process of the seven-step model of migration is basically at the assessment level. Proof of concepts or prototypes for various approaches to the migration along with the leveraging of pricing parameters enables one to make appropriate assessments.

These assessments are about the cost of migration as well as about the ROI that can be achieved in the case of production version. The next process step is in isolating all systemic and environmental dependencies of the enterprise

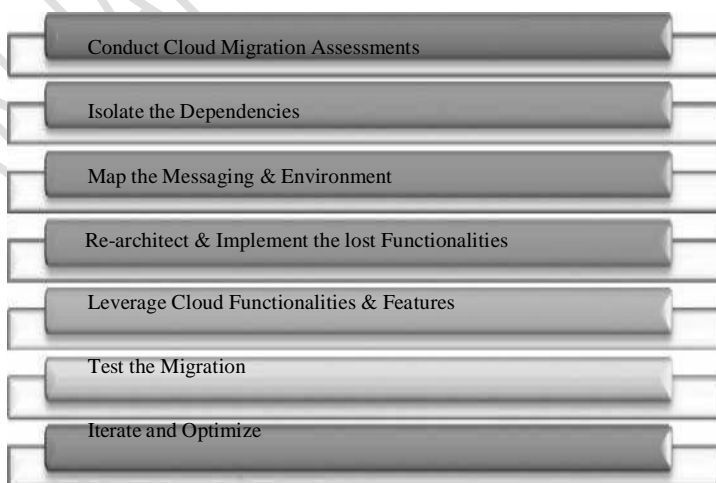


FIGURE : The Seven-Step Model of Migration into the Cloud. (Source: Infosys Research.)

ANNAL WOMEN'S COLLEGE

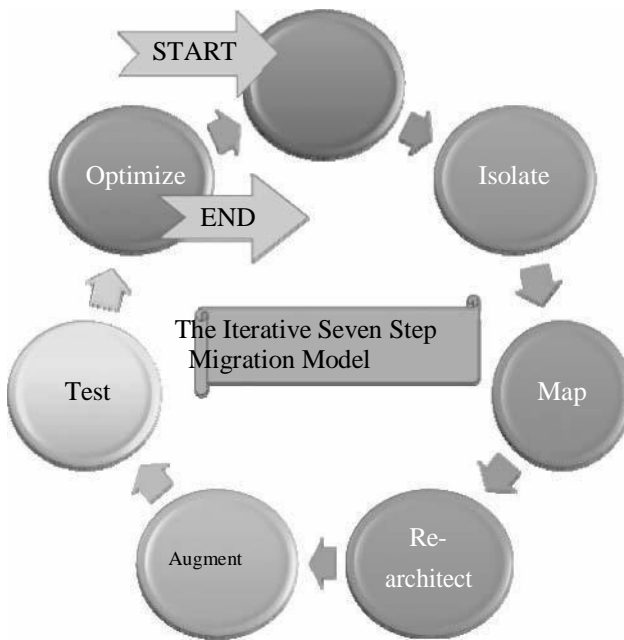


FIGURE : The iterative Seven-step Model of Migration into the Cloud. (Source: Infosys Research.)

application components within the captive data center. This, in turn, yields a picture of the level of complexity of the migration. After isolation is complete, one then goes about generating the mapping constructs between what shall possibly remain in the local captive data center and what goes onto the cloud.

In the next process step we leverage the intrinsic features of the cloud computing service to augment our enterprise application in its own small ways. Compared with the typical approach to migration into the Amazon AWS, our Seven-step model is more generic, versatile, and comprehensive. The typical migration into the Amazon AWS is a phased over several steps. It is about six steps as discussed in several white papers in the Amazon website and is as follows: The first phase is the cloud migration assessment phase wherein dependencies are isolated and strategies worked out to handle these dependencies.

The next phase is in trying out proof of concepts to build a reference migration architecture.

The third phase is the data migration phase wherein database data segmentation and cleansing is completed. This phase also tries to leverage the various cloud storage options as best suited.

The fourth phase comprises the application migration wherein either a “forklift strategy” of migrating the key enterprise application along with its dependencies (other applications) into the cloud is pursued.

UNIT II

THE EVOLUTION OF SaaS

SaaS paradigm is on fast track due to its innate powers and potentials. Executives, entrepreneurs, and end-users are ecstatic about the tactic as well as strategic success of the emerging and evolving SaaS paradigm.

Newer resources and activities are being consistently readied to be delivered as a service. Experts and evangelists are in unison that cloud is to rock the total IT community as the best possible infrastructural solution for effective service delivery.

IT as a Service (ITaaS) is the most recent and efficient delivery method in the decisive IT landscape. With the meteoric and mesmerizing rise of the service orientation principles, every single IT resource, activity and infrastructure is being viewed and visualized as a service that sets the tone for the grand unfolding of the dreamt service era. These days, systems are designed and engineered as elegant collections of enterprising and evolving services. Infrastructures are service-enabled to be actively participative and collaborative. In the same tenor, the much-maligned delivery aspect too has gone through several transformations and today the whole world has solidly settled for the green paradigm 'IT as a service (ITaaS)'.

Integration as a service (IaaS) is the budding and distinctive capability of clouds in fulfilling the business integration requirements. Increasingly business applications are deployed in clouds to reap the business and technical benefits. On the other hand, there are still innumerable applications and data sources locally stationed and sustained primarily due to the security reason. The question here is how to create a seamless connectivity between those hosted and on-premise applications to empower them to work together. IaaS over-comes these challenges by smartly utilizing the time-tested business-to-business (B2B) integration technology as the value-added bridge between SaaS solutions and in-house business applications.

The use of hub & spoke (H&S) architecture further simplifies the implementation and avoids placing an excessive processing burden on the customer sides. The hub is installed at the SaaS provider's

cloud center to do the heavy lifting such as reformatting files. Clouds, being the Web-based infrastructures are the best fit for hosting scores of unified and utility-like platforms to take care of all sorts of brokering needs among connected and distributed ICT systems.

1. The Web is the largest digital information superhighway
2. The Web is the largest repository of all kinds of resources such as web pages, applications comprising enterprise components, business services, beans, POJOs, blogs, corporate data, etc.
3. The Web is turning out to be the open, cost-effective and generic business execution platform (E-commerce, business, auction, etc. happen in the web for global users) comprising a wider variety of containers, adaptors, drivers, connectors, etc.
4. The Web is the global-scale communication infrastructure (VoIP, Video conferencing, IP TV etc.)
5. The Web is the next-generation discovery, Connectivity, and integration middleware

Thus the unprecedented absorption and adoption of the Internet is the key driver for the continued success of the cloud computing.

THE CHALLENGES OF SaaS PARADIGM

SaaS and cloud concepts too suffer a number of limitations. Loss or lack of the following features deters the massive adoption of clouds

1. Controllability
2. Visibility & flexibility
3. Security and Privacy
4. High Performance and Availability
5. Integration and Composition
6. Standards

A number of approaches are being investigated for resolving the identified issues and flaws. Private cloud, hybrid and the latest community cloud are being prescribed as the solution for most of these inefficiencies and deficiencies. As rightly pointed out by someone in his weblogs, still there are miles to go.

Integration Conundrum. While SaaS applications offer outstanding value in terms of features and functionalities relative to cost, they have introduced several challenges specific to integration. The first issue is that the majority of SaaS applications are point solutions and service one line of business. As a result, companies without a method of synchronizing data between multiple lines of businesses are at a serious disadvantage in terms of maintaining accurate data, forecasting, and automating key business processes. Real-time data and functionality sharing is an essential ingredient for clouds.

APIs are Insufficient. Many SaaS providers have responded to the integration challenge by developing application programming interfaces (APIs). Unfortunately, accessing and managing data via an API requires a significant amount of coding as well as maintenance due to frequent API modifications and updates.

Data Transmission Security. SaaS providers go to great length to ensure that customer data is secure within the hosted environment. However, the need to transfer data from on-premise systems or applications behind the firewall with SaaS applications hosted outside of the client's data center poses new challenges that need to be addressed by the integration solution of choice. It is critical that the integration solution is able to synchronize data bi-directionally from SaaS to on-premise without opening the firewall. Best-of-breed integration providers can offer the ability to do so by utilizing the same security as when a user is manually typing data into a web browser behind the firewall.

The Impacts of Clouds :. On the infrastructural front, in the recent past, the clouds have arrived onto the scene powerfully and have extended the horizon and the boundary of business applications, events and data. That is, business applications, development platforms etc. are getting moved to elastic, online and on-demand cloud infrastructures.

APPROACHING THE SaaS INTEGRATION ENIGMA

Integration as a Service (IaaS) is all about the migration of the functionality of a typical enterprise application integration (EAI) hub / enterprise service bus (ESB) into the cloud for providing for smooth

data transport between any enterprise and SaaS applications. Users subscribe to IaaS as they would do for any other SaaS application.

Cloud middleware is the next logical evolution of traditional middleware solutions. That is, cloud middleware will be made available as a service. Due to varying integration requirements and scenarios, there are a number of middleware technologies and products such as JMS-compliant message queues and integration backbones such as EAI, ESB, EII, EDB, CEP, etc. For performance sake, clusters, fabrics, grids, and federations of hubs, brokers, and buses are being leveraged.

For service integration, it is enterprise service bus (ESB) and for data integration, it is enterprise data bus (EDB). Besides there are message oriented middleware (MOM) and message brokers for integrating decoupled applications through message passing and pick up.

Why SaaS Integration is hard?. As indicated in the white paper, there is a mid-sized paper company that recently became a Salesforce.com CRM customer. The company currently leverages an on-premise custom system that uses an Oracle database to track inventory and sales.

The use of the Salesforce.com system provides the company with a significant value in terms of customer and sales management.

Integration is not easier either to implement as successful untangling from the knotty situation is a big issue. The web of application and data silos really makes the integration task difficult and hence choosing a best-in class scheme for flexible and futuristic integration is insisted very frequently.

Limited Access. Access to cloud resources (SaaS, PaaS, and the infrastructures) is more limited than local applications. Accessing local applications is quite simple and faster. Imbedding integration points in local as well as custom applications is easier.

Even with the commercial applications, it is always possible to slip in database-triggers to raise events and provide hooks for integration access. Once applications move to the cloud, custom applications must be designed to support integration because there is no longer that low-level of access.

Enterprises putting their applications in the cloud or those subscribers of cloud-based business services are dependent on the vendor to provide the integration hooks and APIs. For example, the

SalesForce.com web services API does not support transactions against multiple records, which means integration code has to handle that logic.

Dynamic Resources. Cloud resources are virtualized and service-oriented. That is, everything is expressed and exposed as a service. Due to the dynamism factor that is sweeping the whole cloud ecosystem, application versioning and infrastructural changes are liable for dynamic changes.

Performance. Clouds support application scalability and resource elasticity. However the network distances between elements in the cloud are no longer under our control. Bandwidth is not the limiting factor in most integration scenarios but the round trip latency is an issue not to be sidestepped. Because of the latency aggravation, the cloud integration performance is bound to slow down.

NEW INTEGRATION SCENARIOS

Before the cloud model, we had to stitch and tie local systems together. With the shift to a cloud model is on the anvil, we now have to connect local applications to the cloud, and we also have to connect cloud applications to each other, which add new permutations to the complex integration channel matrix

Cloud Integration Scenarios. We have identified three major integration scenarios as discussed below.

Within a Public Cloud (figure 1). Two different applications are hosted in a cloud. The role of the cloud integration middleware (say cloud-based ESB or internet service bus (ISB)) is to seamlessly enable these applications to talk to each other. The possible sub-scenarios include these applications can be owned

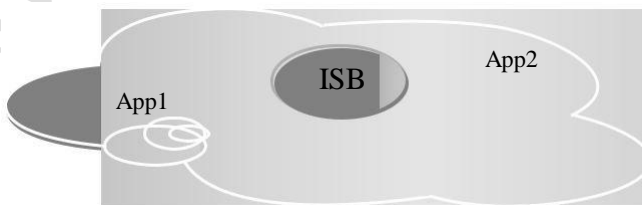


FIGURE :Within a Public Cloud.

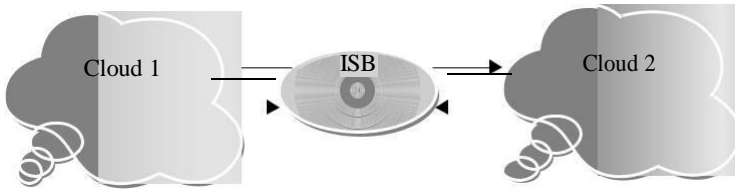


FIGURE :Across Homogeneous Clouds.

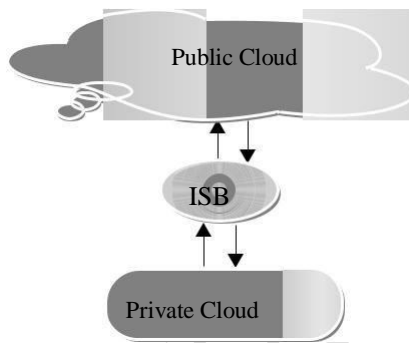


FIGURE :Across Heterogeneous Clouds.

by two different companies. They may live in a single physical server but run on different virtual machines.

Homogeneous Clouds (figure 2). The applications to be integrated are posited in two geographically separated cloud infrastructures. The integration middleware can be in cloud 1 or 2 or in a separate cloud.

There is a need for data and protocol transformation and they get done by the ISB. The approach is more or less compatible to enterprise application integration procedure.

Heterogeneous Clouds (figure 3). One application is in public cloud and the other application is private cloud.

As described above, this is the currently dominating scene for cloud integration. That is, businesses are subscribing to popular on-demand enter-prise packages from established providers such as Salesforce.com..

THE INTEGRATION METHODOLOGIES

Excluding the custom integration through hand-coding, there are three types for cloud integration

1. Traditional Enterprise Integration Tools can be empowered with special connectors to access Cloud-located Applications— This is the most likely approach for IT organizations, which have already invested a lot in integration suite for their application integration needs.
2. Traditional Enterprise Integration Tools are hosted in the Cloud—This approach is similar to the first option except that the integration software suite is now hosted in any third-party cloud infrastructures so that the enterprise does not worry about procuring and managing the hardware or installing the integration software. This is a good fit for IT organizations that outsource the integration projects to IT service organizations and systems integrators, who have the skills and resources to create and deliver integrated systems.
3. Integration-as-a-Service (IaaS) or On-Demand Integration Offerings—
These are SaaS applications that are designed to deliver the integration service securely over the Internet and are able to integrate cloud applications with the on-premise systems, cloud-to-cloud applications. This approach is a good fit for companies who insist about the ease of use, ease of maintenance, time to deployment, and are on a tight budget. SaaS administrator or business analyst as the primary resource for managing and maintaining their integration work. A good example is Informatica On-Demand Integration Services.

In a nutshell, the integration requirements can be realised using any one of the following methods and middleware products.

1. Hosted and extended ESB (Internet service bus / cloud integration bus)

2. Online Message Queues, Brokers and Hubs
3. Wizard and configuration-based integration platforms (Niche integration solutions)
4. Integration Service Portfolio Approach
5. Appliance-based Integration (Standalone or Hosted)

With the emergence of the cloud space, the integration scope grows further and hence people are looking out for robust and resilient solutions and services that would speed up and simplify the whole process of integration.

Characteristics of Integration Solutions and Products. The key attributes of integration platforms and backbones gleaned and gained from integration projects experience are connectivity, semantic mediation, Data mediation, integrity, security, governance etc

Connectivity refers to the ability of the integration engine to engage with both the source and target systems using available native interfaces. This means leveraging the interface that each provides, which could vary from standards-based interfaces, such as Web services, to older and proprietary interfaces.

Semantic Mediation refers to the ability to account for the differences between application semantics between two or more systems. Semantics means how information gets understood, interpreted and represented within information systems.

Data Mediation converts data from a source data format into destination data format. Coupled with semantic mediation, data mediation or data transformation is the process of converting data from one native format on the source system, to another data format for the target system.

Data Migration is the process of transferring data between storage types, formats, or systems. Data migration means that the data in the old system is mapped to the new systems, typically leveraging data extraction and data loading technologies.

Data Security means the ability to insure that information extracted from the source systems has to securely be placed into target systems. The integration method must leverage the native security systems of the source and target systems, mediate the differences,

and provide the ability to transport the information safely between the connected systems.

Data Integrity means data is complete and consistent. Thus, integrity has to be guaranteed when data is getting mapped and maintained during integration operations, such as data synchronization between on-premise and SaaS-based systems.

Governance refers to the processes and technologies that surround a system or systems, which control how those systems are accessed and leveraged. Within the integration perspective, governance is about managing changes to core information resources, including data semantics, structure, and interface

SaaS INTEGRATION PRODUCTS AND PLATFORMS

Cloud-centric integration solutions are being developed and demonstrated for showcasing their capabilities for integrating enterprise and cloud applications. The integration puzzle has been the toughest assignment for long due to heterogeneity and multiplicity-induced complexity. But as the days go by, there will be a huge market for application and service integration. Interoperability will become the most fundamental thing. Composition and collaboration will become critical and crucial for the mass adoption of clouds, which are prescribed and proclaimed as the next-generation infrastructure for creating, deploying and delivering hordes of ambient, artistic, adaptive, and agile services. Cloud interoperability is the prime demand and the figure 3.4 for creating cloud peers, clusters, fabrics, and grids.

1.Jitterbit

Force.com is a Platform as a Service (PaaS), enabling developers to create and deliver any kind of on-demand business application. However, in order to take advantage of this breakthrough cloud technology, there is a need for a flexible and robust integration solution to synchronize force.com with any on-demand or on-premise enterprise applications, databases, and legacy systems.

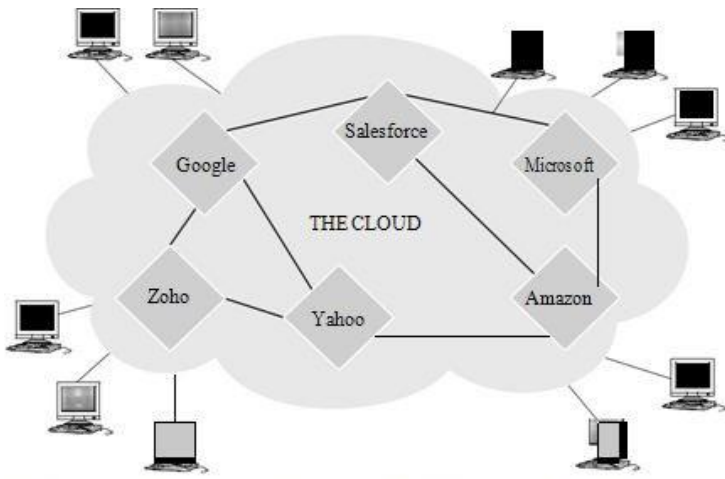


FIGURE: The Smooth and Spontaneous Cloud Interaction via Open Clouds.

Until now, integrating force.com applications with other on-demand applications and systems within an enterprise has seemed like a daunting and doughty task that required too much time, money, and expertise.

Jitterbit is a fully graphical integration solution that provides users a versatile platform and a suite of productivity tools to reduce the integration efforts sharply. Jitterbit can be used standalone or with existing EAI infra-structures, enabling users to create new projects or consume and modify existing ones offered by the open source community or service provider.

The Jitterbit solution enables the cool integration among confidential and corporate data, enterprise applications, web services, XML data sources, legacy systems, simple and complex flat files.

Jitterbit Integration Environment An intuitive point-and-click graphical UI that enables to quickly configure, test, deploy and manage integration projects on the Jitterbit server.

Jitterbit Integration Server A powerful and scalable run-time engine that processes all the integration operations, fully configurable and manage-able from the Jitterbit application.

Jitterbit is making integration easier, faster, and more affordable than ever before. Using Jitterbit, one can connect force.com with a wide variety

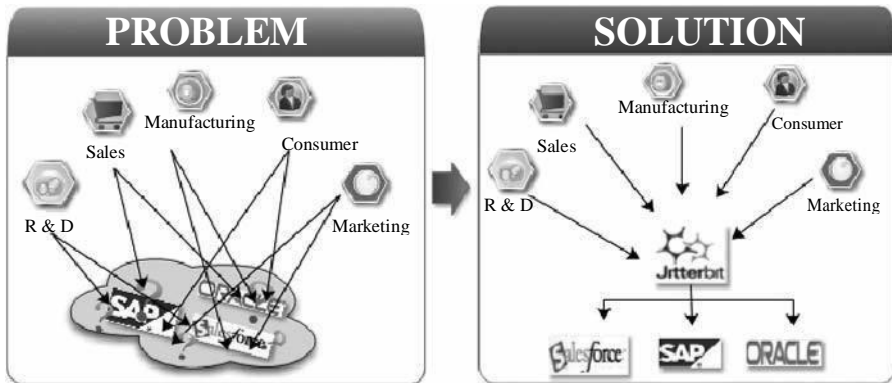


FIGURE :Linkage of On-Premise with Online and On-Demand Applications.

of on-premise systems including ERP, databases, flat files and custom applications. The figure illustrates how Jitterbit links a number of functional and vertical enterprise systems with on-demand applications.

SaaS INTEGRATION SERVICES

We have seen the state-of-the-art cloud-based data integration platforms for real-time data sharing among enterprise information systems and cloud applications. Another fast-emerging option is to link enterprise and cloud systems via messaging. This has forced vendors and service organizations to take message oriented middleware (MoM) to the all-powerful cloud infrastructures. Going forward, there are coordinated and calculated efforts for taking the standards-compatible enterprise service bus (ESB) to clouds in order to guarantee message enrichment, mediation, content and context-based message routing.

Thus both loosely or lightly coupled and decoupled cloud services and applications will become a reality soon with the maturity and durability of message-centric and cloud-based service bus suites. We can still visualise the deployment of complex event processing (CEP) engines in clouds in order to capture and capitalise streams of events from diverse sources in different formats and forms in order to infer the existing and emerging situation precisely and concisely.

Further on, all kinds of risks, threats, vulnerabilities, opportunities, trends, tips, associations, patterns, and other tactical as well as strategic insights and actionable insights can be deduced to act upon confidently and at real time.

1. Informatica On-Demand

Informatica offers a set of innovative on-demand data integration solutions called Informatica On-Demand Services. This is a cluster of easy-to-use SaaS offerings, which facilitate integrating data in SaaS applications, seamlessly and securely across the Internet with data in on-premise applications. There are a few key benefits to leveraging this maturing technology.

Rapid development and deployment with zero maintenance of the integration technology.

Automatically upgraded and continuously enhanced by vendor.

Proven SaaS integration solutions, such as integration with Salesforce.com, meaning that the connections and the metadata understanding are provided.

Proven data transfer and translation technology, meaning that core integration services such as connectivity and semantic mediation are built into the technology.

Informatica On-Demand has taken the unique approach of moving its industry leading PowerCenter Data Integration Platform to the hosted model and then configuring it to be a true multi-tenant solution. That means that when developing new features or enhancements, they are immediately made available to all of their customers transparently. That means, no complex software upgrades required and no additional fee is demanded. Fixing, patching, versioning, etc are taken care of by the providers at no cost for the subscribers. Still the service and operation level agreements are being fully met. And the multi-tenant architecture means that bandwidth and scalability are shared resources so meeting different capacity demands becomes smoother and simpler.

2. Microsoft Internet Service Bus (ISB)

Azure is an upcoming cloud operating system from Microsoft. This makes development, depositing and delivering Web and Windows application on cloud centers easier and cost-effective. Developers' productivity shoots up, customers' preferences are being provided, the enterprise goal of "more with less" gets achieved, etc. Azure is being projected as the comprehensive yet compact cloud framework that comprises a wider variety of enabling tools for a slew of tasks and a growing service portfolio. The primary components are explained below.

Microsoft .NET Services. is a set of Microsoft-built and hosted cloud infrastructure services for building Internet-enabled applications and the ISB acts as the cloud middleware providing diverse applications with a common infrastructure to name, discover, expose, secure and orchestrate web services. The following are the three broad areas.

.NET Service Bus. The .NET Service Bus (figure) provides a hosted, secure, and broadly accessible infrastructure for pervasive communication,

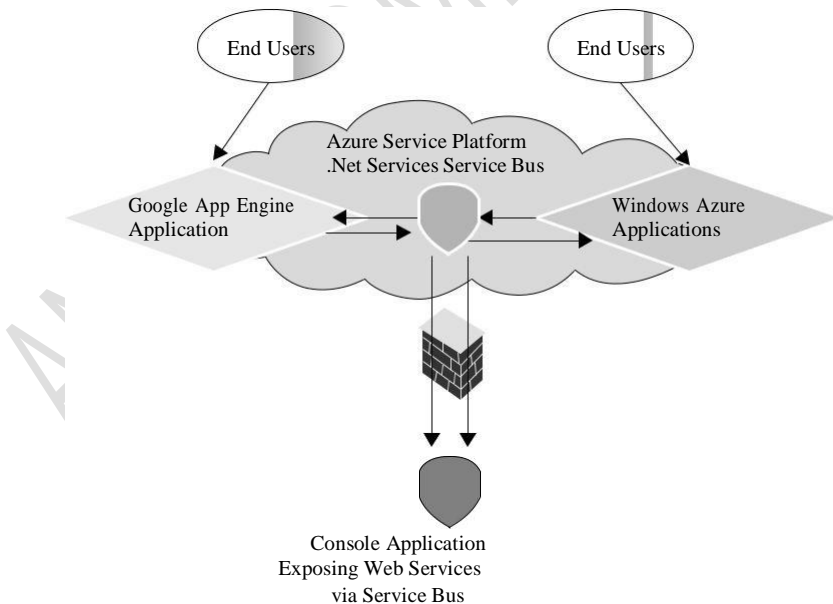


FIGURE .NET Service Bus.

large-scale event distribution, naming, and service publishing. Services can be exposed through the Service Bus Relay, providing connectivity options for service endpoints that would otherwise be difficult or impossible to reach. Endpoints can be located behind network address translation (NAT) boundaries or bound to frequently changing, dynamically assigned IP addresses, or both.

.NET Access Control Service. The .NET Access Control Service is a hosted, secure, standards-based infrastructure for multiparty, federated authentication, rules-driven, and claims-based authorization. The Access Control Service's capabilities range from simple, one-step, user name/password-based authentication and authorization with Web-style HTTP requests to sophisticated WS-Federation scenarios that employ two or more collaborating WS-Trust Security Token Services. The Access Control Service allows applications to rely on

.NET Services solution credentials for simple scenarios or on on-premise enterprise accounts managed in Microsoft Active Directory and federated with the Access Control Service via next-generation Microsoft Active Directory Federation Services.

.NET Workflow Service. The .NET Workflow Service provide a hosted environment for service orchestration based on the familiar Windows Work-flow Foundation (WWF) development experience. The Workflow services will provide a set of specialized activities for rules-based control flow, service invocation, as well as message processing and correlation that can be executed on demand, on schedule, and at scale inside the.NET Services environment.

Relay Services. Often when we connect a service, it is located behind the firewall and behind the load balancer. Its address is dynamic and can be

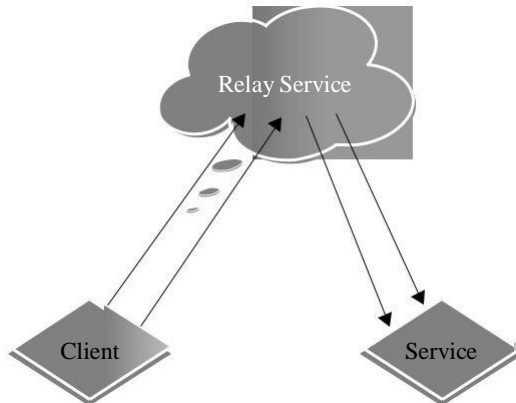


FIGURE :The .NET Relay Service.

resolved only on local network. When we are having the service call-backs to the client, the connectivity challenges lead to scalability, availability and security issues. The solution to Internet connectivity challenges is instead of connecting client directly to the service we can use a relay service as pictorially represented in the relay service figure.

The Relay service is a service residing in the cloud whose job is to assist the connectivity and relaying the calls to the service. Relay Service solution require both the client and the service intranets to allow connections to the cloud.

BUSINESSES-TO-BUSINESS INTEGRATION (B2Bi) SERVICES

B2Bi has been a mainstream activity for connecting geographically distributed businesses for purposeful and beneficial cooperation. Products vendors have come out with competent B2B hubs and suites for enabling smooth data sharing in standards-compliant manner among the participating enterprises. Now with the surging popularity of clouds, there are serious and sincere efforts to posit these products in clouds in order to deliver B2Bi as a service with very least investment and maintenance costs. The cloud ideas and ideals lay the strong and stimulating foundation for transitioning from the capital expenditure to operational expenditure and for sustaining the transformed.

There are several proven integration methods in the B2Bi space and they can be captured and capitalized for achieving quicker success and better return and value in the evolving IaaS landscape. B2Bi systems are good candidate for IaaS as they are traditionally employed to automate business processes between manufacturers and their external trading partners such as retail, warehouse, transport, and inventory systems.

This means that they provide application-to-application (A2A) connectivity along with functionality that is crucial to linking internal and external software: i.e. secure data exchange across the corporate firewall. Unlike pure EAI solutions designed only for internal data sharing, B2Bi platforms have the ability to encrypt files for safe passage across the public network, manage large data volumes, transfer batch files, convert disparate file formats and

guarantee data accuracy, integrity, confidentiality, and delivery. Just as these abilities ensure smooth communication between manufacturers and their external suppliers or customers, they also enable reliable interchange between hosted and installed applications.

The IaaS model also leverages the adapter libraries developed by B2Bi vendors to provide rapid integration with various business systems. Because the B2Bi partners have the expertise and experience and can supply pre-built connectors for major ERP, CRM, SCM and other packaged business applications as well as legacy systems from AS400 to MVS and mainframe.

The use of a hub-and-spoke centralised architecture further simplifies implementation and provides a good control and grip on the system management and finally this avoids placing an excessive processing burden on the customer side.

The hub is installed at the SaaS provider's cloud center to do the heavy lifting such as reformatting files. A spoke unit, typically consisting of a small downloadable Java client, is then deployed at each user site to handle basic tasks such as data transfer.

This also eliminates the need for an expensive server-based solution, data mapping and other tasks at the customer location. As the Internet is the principal communication infrastructure, enterprises can leverage the IaaS to sync up with their partners across the continents towards smart and systematic collaboration.

Cloud- based Enterprise Mashup Integration Services for B2B Scenarios . There is a vast need for infrequent, situational and ad-hoc B2B applications desired by the mass of business end-users. Enterprise mashup and lightweight composition approaches and tools are promising methods to unleash the huge and untapped potential of empowering end-users to develop or assemble aligned and aware composite services in order to overcome the “long-tail” dilemma. Currently available solutions to support B2B collaborations focus on the automation of long-term business relationships and still lack to provide their users intuitive ways to modify or to extend them according to their ad-hoc or situational needs. Conventional proceeding in the development of such applications directs to an immense use of time and work due to long development cycles and a lack of required business knowledge.

Especially in the area of applications to support B2B collaborations, current offerings are characterized by a high richness but low reach, like B2B hubs that focus on many features enabling electronic collaboration, but lack availability for especially small organizations or even individuals. The other extreme solutions with a low reach but high richness such as web sites, portals and emails, lack standardization and formularization which makes them inappropriate for automated or special enterprises’ needs. New development approaches are hence needed to overcome these hurdles and hitches to involve non-technical business users into the development process in order to address this long tail syndrome, to realize cost-effectiveness and efficiency gains, and to overcome the traditional constrictions between IT department and business units.

Enterprise Mashups, a kind of new-generation Web-based applications, seem to adequately fulfill the individual and heterogeneous requirements of end-users and foster End User Development (EUD). To shorten the traditional and time-consuming development process, these new breed of applications are developed by non-professional programmers, often in a non-formal, iterative, and collaborative way by assembling existing building blocks.

Another challenge in B2B integration is the ownership of and responsibility for processes. In many inter-organizational settings, business processes are only sparsely structured and formalized, rather loosely coupled and/or based on ad-hoc cooperation. Inter-organizational collaborations tend to involve more and more participants and the growing number of participants also draws a huge amount of differing requirements. Also, the participants may act according to different roles, controls and priorities. Historically, the focus for collaboration was participation within teams which were managed according to one set of rules.

Now, in supporting supplier and partner co-innovation and customer co-creation, the focus is shifting to collaboration which has to embrace the participants, who are influenced yet restricted by multiple domains of control and disparate processes and practices. This represents the game-changing shift from static B2B approaches to new and dynamic B2B integration, which can adaptively act and react to any unexpected disruptions, can allow a rapid configuration

and customization and can manage and moderate the rising complexity by the use of end-to-end business processes.

Both Electronic data interchange translators (EDI) and Managed file transfer (MFT) have a longer history, while B2B gateways only have emerged during the last decade. However, most of the available solutions aim at supporting medium to larger companies, resulting from their high costs and long implementation cycles and times, which make them unaffordable and unattractive to smaller organizations. Consequently, these offerings are not suitable for short-term collaborations, which need to be set up in an ad hoc manner.

Enterprise Mashup Platforms and Tools. Mashups are the adept combination of different and distributed resources including content, data or application functionality. Resources represent the core building blocks for mashups. Resources can be accessed through APIs, which encapsulate the resources and describe the interface through which they are made available. Widgets or gadgets primarily put a face on the underlying resources by providing a graphical representation for them and piping the data received from the resources. Piping can include operators like aggregation, merging or filtering. Mashup platform is a Web based tool that allows the creation of Mashups by piping resources into Gadgets and wiring Gadgets together.

Mashups can help adding richness to existing lightweight solutions such as Websites or Portals by adding a certain level of formalization and standardization. Mashups facilitate the ease of mixing and transforming various sources of information internally and from business partners. Complexity in B2B operations is often linked with heterogeneous systems and platforms. The tedious integration process and requirements of various support and maintenance for the software is a major hindrance to today's dynamic B2B integration, especially for the small and medium enterprises.

The Mashup integration services are being implemented as a prototype in the FAST project. The layers of the prototype are illustrated in figure 3.9 illustrating the architecture, which describes how these services work together. The authors of this framework have given an outlook on the technical realization of the services using cloud infrastructures and services.

Prototype architecture shows the services and their relations to each other. The core services are shown within the box in the middle. The external services shown under the box are attached via APIs to allow the usage of third-party offerings to realize their functionality. Users access the services through a Mashup platform of their choice. The Mashup platforms are connected via APIs to the Mashup integration services.

To use the services, users have to identify themselves against the user-access control service. This service is connected to a user management service, which controls the users and their settings. The

user management service is connected via an API to allow the usage of external services, e.g. a corporate user database. All data coming from the users go through a translation engine to unify the data objects and protocols, so that different Mashup platforms can be integrated. The translation engine has an interface which allows connecting other external translation engines to add support for additional protocol and data standards.

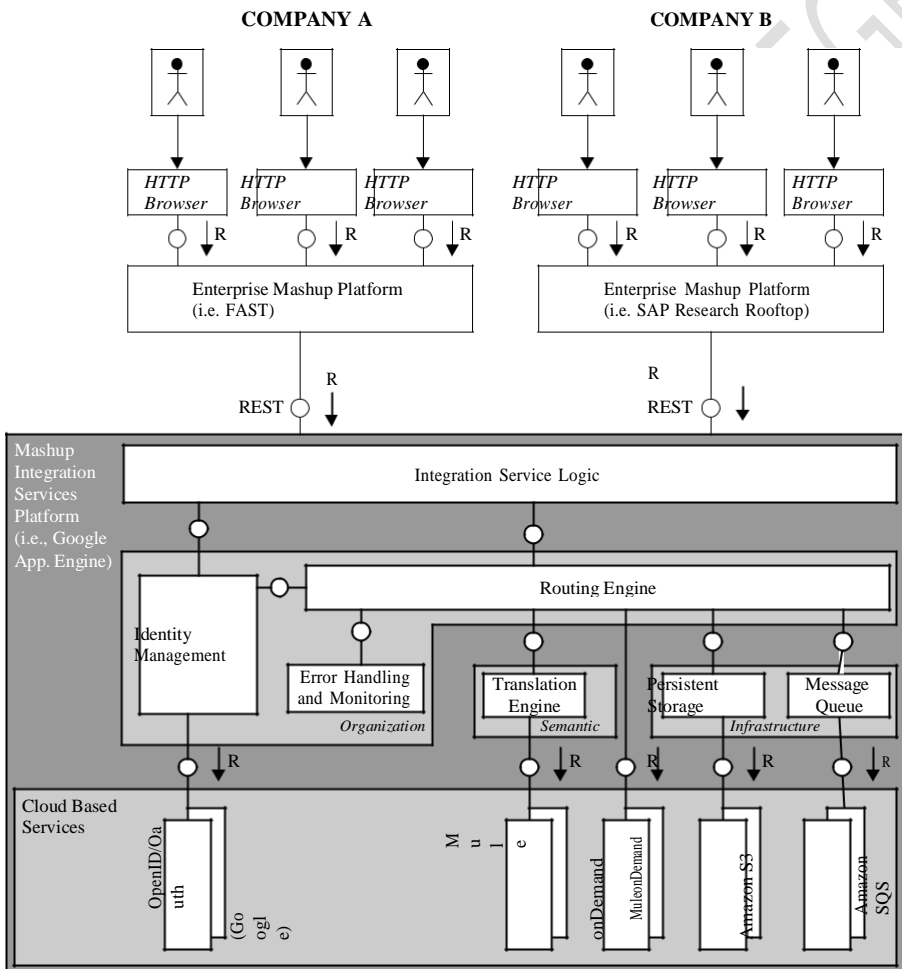


FIGURE: Cloud- based Enterprise Mashup Integration Platform Architecture.

To simplify this, a Gadget could be provided for the end-user. The routing engine is also connected to a message queue via an API. Thus, different message queue engines are attachable. The message queue is responsible for storing and forwarding the messages controlled by the routing engine. Beneath the message queue, a persistent storage, also connected via an API to allow exchangeability, is available to store large data. The error handling and monitoring service allows tracking the message-flow to detect errors and to collect statistical data. The Mashup integration service is hosted as a cloud-based service. Also, there are cloud-based services available which provide the functionality required by the integration service. In this way, the Mashup integration service can reuse and leverage the existing cloud services to speed up the implementation.

Message Queue. The message queue could be realized by using Amazon's Simple Queue Service (SQS). SQS is a web-service which provides a queue for messages and stores them until they can be processed. The Mashup integration services, especially the routing engine, can put messages into the queue and recall them when they are needed.

Persistent Storage. Amazon Simple Storage Service⁵ (S3) is also a web-service. The routing engine can use this service to store large files.

Translation Engine. This is primarily focused on translating between different protocols which the Mashup platforms it connects can understand, e.g. REST or SOAP web services. However, if the need of translation of the objects transferred arises, this could be attached to the translation engine.

A company requiring such a service could on the one hand develop such a service and connect it to the Mashup integration services. Another possibility for this would be to connect existing translation services, e.g., the services by Mule on Demand, which is also a cloud-based offering.

Interaction between the Services. The diagram describes the process of a message being delivered and handled by the Mashup Integration Services Platform. The precondition for this process is that a user already established a route to a recipient.

After having received a message from an Enterprise Mashup tool via an API, the Integration Services first check the access rights of the sender of the message against an external service. An incoming message is processed only if sender of the message is authorized, that is, he has the right to deliver the message to the recipient and to use the Mashup integration services.

If he is not authorized, the processing stops, and an error message gets logged. The error log message is written into a log file, which could reside on Amazon's Simple Storage Service (S3). If the message has been accepted, it is put in the message queue in Amazon's SQS service.

If required, the message is being translated into another format, which can also be done by an external, cloud-based service. After that, the services can begin trying delivering the message to a recipient.

Evaluating the recipients of the message is based on the rules stored in the routing engine which have been configured by a user before. Finally, the successful delivery of the message can be logged, or an error if one occurred.

THE ENTERPRISE CLOUD COMPUTING PARADIGM

INTRODUCTION

Cloud computing is still in its early stages and constantly undergoing changes as new vendors, offers, services appear in the cloud market. This evolution of the cloud computing model is driven by cloud providers bringing new services to the ecosystem or revamped and efficient exiting services primarily triggered by the ever changing requirements by the consumers.

Enterprise cloud computing is the alignment of a cloud computing model with an organization's business objectives (profit, return on investment, reduction of operations costs) and processes. This chapter explores this paradigm with respect to its motivations, objectives, strategies and methods.

BACKGROUND

According to NIST , cloud computing is composed of five essential characteristics: on-demand self-service, broad network access, resource pool-ing, rapid elasticity, and measured service. The ways in which these character-istics are manifested in an enterprise context vary according to the deployment model employed.

Relevant Deployment Models for Enterprise Cloud Computing

There are some general cloud deployment models that are accepted by the majority of cloud stakeholders today, as suggested by the references and discussed in the following:

Public clouds are provided by a designated service provider for general public under a utility based pay-per-use consumption model. The cloud resources are hosted generally on the service provider's premises. Popular examples of public clouds are Amazon's AWS (EC2, S3 etc.), Rackspace Cloud Suite, and Microsoft's Azure Service Platform.

Private clouds are built, operated, and managed by an organization for its internal use only to support its business operations exclusively. Public, private, and government organizations worldwide are adopting this model to exploit the cloud benefits like flexibility, cost reduction, agility and so on.

Virtual private clouds are a derivative of the private cloud deployment model but are further characterized by an isolated and secure segment of resources, created as an overlay on top of public cloud infrastructure using advanced network virtualization capabilities.

Some of the public cloud vendors that offer this capability include Amazon Virtual Private Cloud , OpSource Cloud , and Skytap Virtual Lab .Community clouds are shared by several organizations and support a specific community that has shared concerns (e.g., mission, security requirements, policy, and compliance considerations). They may be managed by the organizations or a third party and may exist on premise or off premise . One example of this is OpenCirrus formed by HP, Intel, Yahoo, and others.

Managed clouds arise when the physical infrastructure is owned by and/or physically located in the organization's data centers with an

extension of management and security control plane controlled by the managed service provider . This deployment model isn't widely agreed upon, however, some vendors like and NaviSite's NaviCloud offers claim to be managed cloud offerings.

Hybrid clouds are a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load-balancing between clouds) . Recently some cloud vendors have started offering solutions which can be used to enable these hybrid cloud deployment models. Some examples of these offerings include Amazon Virtual Private Cloud, Skytap Virtual Lab , and CohesiveFT VPN-Cubed . These solutions work by creating IPsec VPN tunneling capabilities to connect the public cloud infrastructure to the on-premise cloud resources.

The selection of a deployment model depends on the opportunities to increase earnings and reduce costs i.e. capital expenses (CAPEX) and operating expenses (OPEX). Such opportunities can also have an element of timeliness associated with it, in that decisions that lead to losses today could be done with a vision of increased earnings and cost reductions in a foreseeable future.

Adoption and Consumption Strategies

The selection of strategies for enterprise cloud computing is critical for IT capability as well as for the earnings and costs the organization experiences, motivating efforts toward convergence of business strategies and IT.

Some critical questions toward this convergence in the enterprise cloud paradigm are as follows:

Will an enterprise cloud strategy increase overall business value?

Are the effort and risks associated with transitioning to an enterprise cloud strategy worth it?

Which areas of business and IT capability should be considered for the enterprise cloud?

Which cloud offerings are relevant for the purposes of an organization? How can the process of transitioning to an enterprise cloud strategy be piloted and systematically executed?

These questions are addressed from two strategic perspectives: (1) adoption and (2) consumption. Figure illustrates a framework for enterprise cloud adoption strategies, where an organization makes a decision to adopt a cloud computing model based on fundamental drivers for cloud computing— scalability, availability, cost and convenience. The notion of a Cloud Data Center (CDC) is used, where the CDC could be an external, internal or federated provider of infrastructure, platform or software services.

An optimal adoption decision cannot be established for all cases because the types of resources (infrastructure, storage, software) obtained from a CDC depend on the size of the organisation understanding of IT impact on business, predictability of workloads, flexibility of existing IT landscape and available budget/resources for testing and piloting. The strategic decisions using these four basic drivers are described in following, stating objectives, conditions and actions.

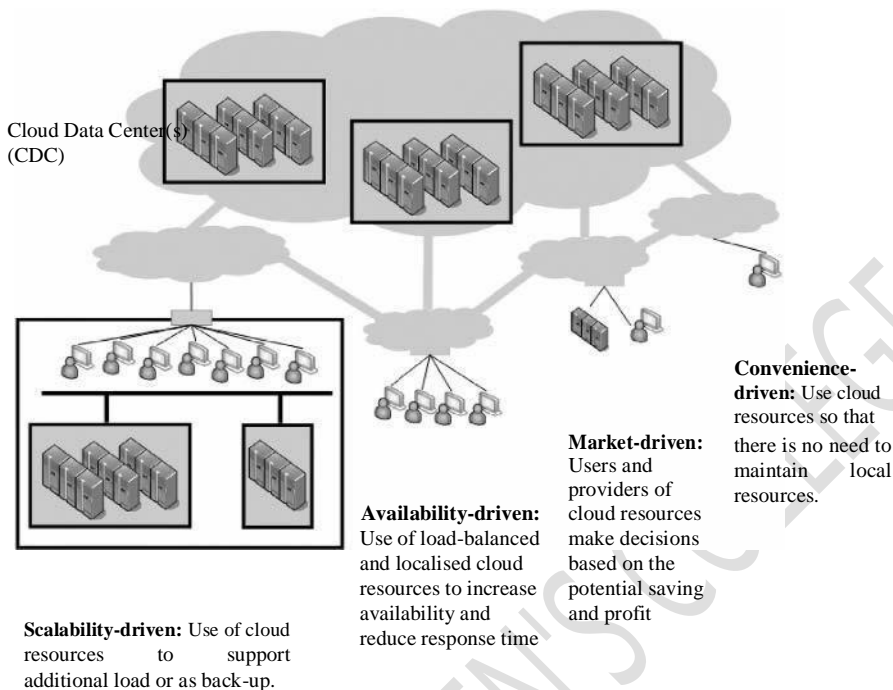
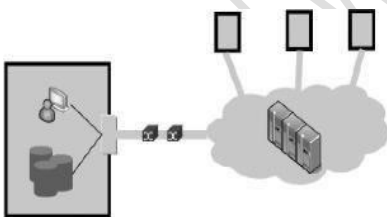


FIGURE : Enterprise cloud adoption strategies using fundamental cloud drivers.

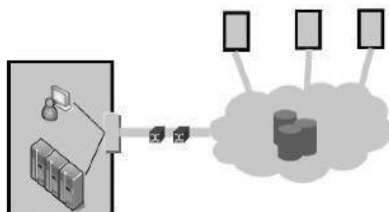
1. Scalability-Driven Strategy. The objective is to support increasing work-loads of the organization without investment and expenses exceeding returns. The conditions are that the effort, costs (CAPEX and OPEX) and time involved in accessing and installing IT capability on a CDC are less than going through a standard hardware and software procurement and licensing process. Scalability will often make use of the IaaS delivery model because the fundamental need of the organization is to have compute power or storage capacity readily available.

2. Availability-Driven Strategy. Availability has close relations to scalability but is more concerned with the assurance that IT capabilities and functions are accessible, usable and acceptable by the standards of users. This is hence the objective of this basic enterprise cloud strategy. The conditions of this strategy are that there exist unpredictable usage peaks and locales, yet the risks (probability and impact) of not being able to satisfy demand outweigh the costs of acquiring the IT capability from a CDC.

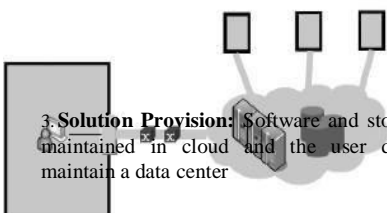
3. Market-Driven Strategy. This strategy is more attractive and viable for small, agile organizations that do not have (or wish to have) massive investments in their IT infrastructure. The objective here is to identify and acquire the “best deals” for IT capabilities as demand and supply change, enabling ongoing reductions in OPEX and CAPEX. There is however always the need to support customer-driven service management based



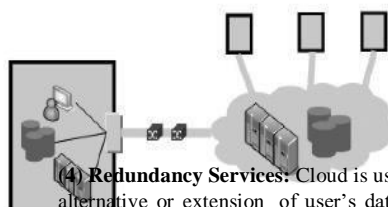
(1) **Software Provision:** Cloud provides instances of software but data is maintained within user's data center



(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

on their profiles and requests service requirements . The conditions for this strategy would be the existence of standardized interfaces between and across CDCs, where the means by which customers access their resources on the CDC, deploy software/data and migrate software/data are uniformed. Ongoing efforts in the Open Cloud Computing Interface (OCCI) Working Group and the Open Cloud Consortium (OCC) are steps toward achieving these standards. Other features such as bidding, negotiation, service discovery and brokering would also be required at communal, regional or global scales.

4. Convenience-Driven Strategy. The objective is to reduce the load and need for dedicated system administrators and to make access to IT capabilities by users easier, regardless of their location and connectivity (e.g. over the Internet). The expectation is that the cost of obtaining IT capabilities from a CDC and making them accessible to users is significantly lower than the cost of having a dedicated administrator. However, it should be noted that, according to a recent Gartner study, the major reason for discontinuing with cloud-related strategies is the difficulty with integration, ahead of issues with the costs of services.

The consumption strategies make a distinction between data and application logic because there are questions of programming models used, data sensitivity, software licensing and expected response times that need to be considered. Figure illustrates a set of enterprise cloud consumption strategies, where an organization makes decisions about how to best deploy its data and software using its internal resources and those of a selected CDC.

There are four consumption strategies identified, where the differences in objectives, conditions and actions reflect the decision of an organization to trade-off hosting costs, controllability and resource elasticity of IT resources for software and data. These are discussed in the following.

1. Software Provision. This strategy is relevant when the elasticity requirement is high for software and low for data, the controllability concerns are low for software and high for data, and the cost reduction concerns for software are high, while cost reduction is not a priority for data, given the high controllability concerns for data, that is, data are highly sensitive. Implementing this strategy sees an organization requesting either software to be delivered as a service (SaaS) by the CDC or access to some portion of the CDC's compute infrastructure as a service (IaaS), such that it can deploy its application software on the provisioned resources. However, the organization chooses to maintain its data internally and hence needs to provide a means for the software running in the CDC to access data within its domain.

This will entail changing some properties at the firewall or maintaining additional, supplementary software for secure access such as VPN, application-level proxy/gateway or wrapper software that could make the data base accessible via a remote messaging or service interface. According to a recent Gartner survey [10], the major hindrance to SaaS adoption is still the pricing and the lack of compelling indicators that the long-term investment in SaaS will be more cost-effective than traditional on-site maintenance of software.

1.Storage Provision. This strategy is relevant when the elasticity requirements is high for data and low for software, while the controllability of software is more critical than for data. This can be the case for data intensive applications, where the results from processing in the application are more critical and sensitive than the data itself. Furthermore, the cost reduction for data resources is a high concern, whereas cost for software, given its criticality, is not an issue for the organization within reasonable means. Other advantages of this strategy include the ease of sharing data between organizations, availability, fast provisioning, and management of storage utilization, because storage is a resource that is constantly in demand. Hasan, Yurcik and Myagmar [11] show in their study of storage service providers that reputation as storage vendors and the existence of established business relationships are major success and sustainability factors in this market.

2.Solution Provision. This strategy is relevant when the elasticity and cost reduction requirements are high for software and data, but the controllability requirements can be entrusted to the CDC. It is not the case that controllability is an insignificant requirement; it is rather the case that the organization trusts the CDC sufficiently to manage access and usage control of its software and data. In some cases the organization might have greater trust in the CDC maintaining and securing its applications and data than it does in its own administrative capabilities.

In other words, there are perceived gains in controllability for placing the entire IT solution (software and data) in the domain of the CDC. Solution provision also seemed like a more viable strategy than software or storage provision strategies, given the limitations of bandwidth between software and data that persists, especially for query-intensive solutions. Such a strategy is also attractive for testing systems, because these generally will not contain sensitive data (i.e., only test data) and are not the production-time versions of the software.

3.Redundancy Services. This strategy can be considered as a hybrid enterprise cloud strategy, where the organization switches between traditional, software, storage or solution management based on changes in its operational conditions and business demands. The trade-offs between controllability and cost reduction will therefore vary based on changes in load experienced by the organization. The strategy is referred to as the “redundancy strategy” because the CDC is used for situations such as disaster recovery, fail-over and load-balancing.

Software, storage or solution services can be implemented using redundancy, such that users are redirected for the purpose of maintaining availability of functionality or performance/response times experienced by the user of the service. Business continuity is then the objective of this strategy, given that downtime and degradation of QoS can result in massive losses. There is however a cost for redundancy, because the subscription and access to redundant services needs to be maintained.

Even though an organization may find a strategy that appears to provide it significant benefits, this does not mean that immediate adoption of the strategy is advised or that the returns on investment will be observed immediately. There are still many issues to be considered when moving enterprise applications to the cloud paradigm.

ISSUES FOR ENTERPRISE APPLICATIONS ON THE CLOUD

Enterprise Resource Planning (ERP) is the most comprehensive definition of enterprise application today. The purpose of ERP solutions is to equip enterprises with a tool to optimize their underlying business processes with a seamless, integrated information flow from suppliers through to manufacturing and distribution and the ability to effectively plan and control all resources, necessary in the face of growing consumer demands, globalization and competition . For these reasons, ERP solutions have emerged as the core of successful information management and the enterprise backbone of nearly any organization.

Organizations that have successfully implemented the ERP systems are reaping the benefits of having integrating working environment, standardized process and operational benefits to the organization .However, as the market rapidly changes, organizations need new solutions for remaining competitive, such that they will constantly need to improve their business practices and procedures. For this reason the enterprise cloud computing paradigm is becoming

attractive as a potential ERP execution environment. Nevertheless, such a transition will require a balance of strategic and operational steps guided by socio-technical considerations, continuous evaluation, and tracking mechanisms .

One of the first issues is that of infrastructure availability. Al-Mashari and Yasser argued that adequate IT infrastructure, hardware and network-ing are crucial for an ERP system's success. It is clear that ERP implementation involves a complex transition from legacy information systems and business processes to an integrated IT infrastructure and common business process throughout the organization. Hardware selection is driven by the organiza-tion's choice of an ERP software package. The ERP software vendor generally certifies which hardware (and hardware configurations) must be used to run the ERP system. This factor has always been considered critical [17]. The IaaS offerings hence bear promising, but also challenging future scenarios for the implementation of ERP systems.

One of the ongoing discussions concerning future scenarios considers varying infrastructure requirements and constraints given different workloads and development phases. Recent surveys among companies in North America and Europe with enterprise-wide IT systems showed that nearly all kinds of workloads are seen to be suitable to be transferred to IaaS offerings. Interest in use for production applications is nearly as high as for test and development use. One might think that companies will be much more comfortable with test and development workloads at an external service provider than with produc-tion workloads, where they must be more cautious.

However, respondents in surveys said they were either just as comfortable, or only up to 8% less comfortable, deploying production workloads on “the cloud” as they were deploying test and development workloads. When the responses for all work-load types are aggregated together, two-thirds or more of firms are willing to put at least one workload type into an IaaS offering at a service provider [21]. More technical issues for enterprise cloud computing adoption arise when considering the operational characteristics and behaviors of transactional and analytical applications [22], which extend and underlie the capabilities of ERP.

Considering Transactional and Analytical Capabilities

Transactional type of applications or so-called OLTP (On-line Transaction Processing) applications, refer to a class of systems that manage transaction-oriented applications, typically using relational databases. These applications rely on strong ACID (atomicity, consistency, isolation, durability) properties and are relatively write/update-intensive. Typical OLTP-type ERP components are sales and distributions (SD), banking and financials, customer relationship management (CRM) and supply chain management (SCM).

These applications face major technical and non-technical challenges to deploy in cloud environments. For instance, they provide mission-critical functions and enterprises have clear security and privacy concerns. The classical transactional systems typically use a shared-everything architecture, while cloud platforms mostly consist of shared-nothing commodity hardware.

ACID properties are also difficult to guarantee given the concurrent cloud-based data management and storage systems. Opportunities arise while the highly complex enterprise applications are decomposed into simpler functional components, which are characterized and engineered accordingly.

For example, salesforce.com focuses on CRM-related applications and provides both a hosted software and development platform. Companies such as taleo.com offer on-demand Human Relationship (HR) applications and are gaining momentum in the SaaS market.

A suite of core business applications as managed services can also be an attractive option, especially for small and medium companies. Despite the big engineering challenges, leading software providers are offering tailored business suite solutions as hosted services (e.g. SAP Business ByDesign).

Secondly, analytical types of applications or so-called OLAP (On-line Analytical Processing) applications, are used to efficiently answer multi-dimensional queries for analysis, reporting, and decision support. Typical OLAP applications are business reporting, marketing, budgeting and forecast-ing, to name a few, which belong to the larger Business Intelligence (BI) category . These systems tend to be read-most or read-only, and ACID guarantees are typically not required.

Because of its data-intensive and data-parallel nature, this type of applications can benefit greatly from the elastic compute and storage available in the cloud. Business Intelligence and analytical applications are relatively better suited to run in a cloud platform with a shared-nothing architecture and commodity hardware. Opportunities arise in the vision of Analytics as a Service, or Agile

Analytics . Data sources residing within private or public clouds, can be processed using elastic computing resources on-demand, accessible via APIs, web services, SQL, BI, and data mining tools.

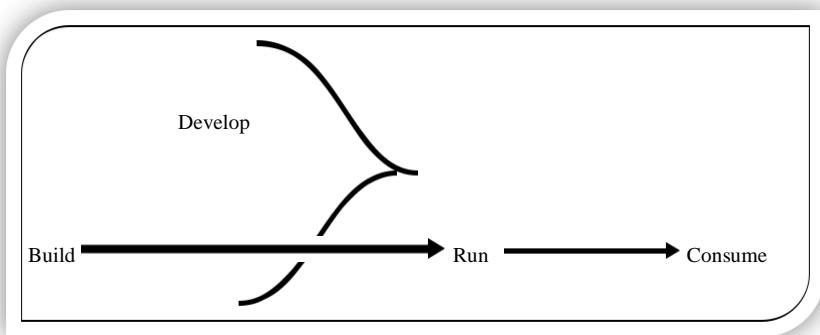
Of course security, data integrity, and other issues can not be overlooked, but a cloud way offers a direction with unmatched performance and TCO (total cost of ownership) benefits toward large-scale analytic processing. Leading providers have been offering on-demand BI and analytics services (e.g. BusinessObjects' ondemand.com and Cognos Now!). Startup companies and niche players (e.g. Brist, PivotLink, Oco) provide a range of SaaS BI products from reporting to ETL (Extract, Transform, Load).

One can conclude that analytical applications will benefit more than their transactional counterparts from the opportunities created by cloud computing, especially on compute elasticity and efficiency. The success of separate functional components such as CRM and HR offered as hosted services has been observed, such that predictions of an integrated suite of core enterprise functionalities emerging as on-demand solutions for small and medium enterprises can be made, given that the transition challenges can be overcome.

TRANSITION CHALLENGES

The very concept of cloud represents a leap from traditional approach for IT to deliver mission critical services. With any leap comes the gap of risk and challenges to overcome. These challenges can be classified in five different categories, which are the five aspects of the enterprise cloud stages: build, develop, migrate, run, and consume (Figure).

The first immediate challenge facing organizations, embarking on this transition, is the understanding of the state of their own IT assets and what is already, can, and cannot be sublimed (the process of transitioning from physical to less visible vapor). Based on the information gathered by this audit they need to evaluate what can be salvaged from the existing infrastructure and how high in the cloud stack they should venture. Most companies are likely to stick to IaaS. However, major development shops may envisage delving into the PaaS and SaaS sphere. Shifting the current architecture requires us to scrap a good chunk of it, which should be taken literally. However, we already see a sprawl of small cloud island appearing within corporations. As this unplanned cloud spreads throughout the organization, coherency becomes a challenge. The requirement for a company-wide cloud approach should then become the number one priority of the CIO, especially when it comes to having a coherent and cost effective development and migration of services on this architecture.



Migrate

FIGURE : Five stages of the cloud.

A second challenge is migration of existing or “legacy” applications to “the cloud.” The expected average lifetime of ERP product is 15 years, which means that companies will need to face this aspect sooner than later as they try to evolve toward the new IT paradigm. An applications migration is not a straightforward process.

It is risky, and doesn’t always guarantee a better service delivery. Firstly, the guarantee that the migration process can be agnostic of the underlying, chosen cloud technology must be provided. If such a process can be automated, a company will still face the same amount of planning, negotiation and testing required for risk mitigation as classical software. It is yet to be proven that companies will be able to balance such expense with the cost cutting, scalability and performance promised by the cloud.

Because migrating to the cloud depends on the concept of decoupling of processes, work needs to be organized using a process (or service) centric model, rather than the standard “silo” one commonly used in IT: server, network, storage, database, and so on.

Not all applications will be able to handle such migration without a tedious and costly overall reengineering. However, if companies decide to (re-) develop from scratch, they will face a completely different kind of hurdle: governance, reliability, security/trust, data management, and control/predictability .

The ownership of enterprise data conjugated with the integration with others applications integration in and from outside the cloud is one of the key challenges.

Future enterprise application development frameworks will need to enable the separation of data management from ownership. From this, it can be extrapolated that SOA, as a style, underlies the architecture and, moreover, the operation of the enterprise cloud.

Challenges for cloud operations can be divided into running the enterprise cloud and running applications on the enterprise cloud. In the first case, companies face difficulties in terms of the changing IT operations of their day today operation. It requires upgrading and updating all the IT department's components. One of these has been notoriously hard to upgrade: the human factor; bringing staff up to speed on the requirements of cloud computing with respect to architecture, implementation, and operation has always been a tedious task.

Once the IT organization has either been upgraded to provide cloud or is able to tap into cloud resource, they face the difficulty of maintaining the services in the cloud. The first one will be to maintain interoperability between in-house infrastructure and service and the CDC (Cloud Data Center).

Furthermore, inasmuch as elasticity is touted as the killer features for enterprise applications, most of the enterprise applications do not really face such wild variations in load to date, such that they need to resort to the cloud for on-demand capacity. More fundamentally, most enterprise apps don't support such features (apart from the few ones built from the ground up for clouds). Before leveraging such features, much more basic functionalities are problematic: monitoring, troubleshooting, and comprehensive capacity planning are actually missing in most offers. Without such features it becomes

very hard to gain visibility into the return on investment and the consumption of cloud services.

Today there are two major cloud pricing models: Allocation based and Usage based .

The first one is provided by the poster child of cloud computing, namely, Amazon. The principle relies on allocation of resource for a fixed amount of time.

The second model does not require any reservation of resource, and the cloud would simply allocate them as a per need basis. When this model combine two typical pricing models: Utility (pay-per-use) and subscription based (fixed per duration charge)—we see the number of variation of offers exploding. Finding the right combination of billing and consumption model for the service is a daunting task. However, the challenge doesn't just stop there. As companies need to evaluate the offers they need to also include the hidden costs such as lost IP, risk, migration, delays and provider overheads. This combination can be compared to trying to choose a new mobile with carrier plan. Not to mention that some providers are proposing to introduce a subscription scheme in order to palliate with their limited resource within their unlimited offer.

This is similar to what ISPs would have done with their content rationing strategies. The market dynamics will hence evolve alongside the technology for the enterprise cloud computing paradigm.

ENTERPRISE CLOUD TECHNOLOGY AND MARKET EVOLUTION

Today's enterprise landscapes to the enterprise computing paradigm, featuring the convergence of business and IT and an open, service oriented marketplace.

1. Technology Drivers for Enterprise Cloud Computing Evolution

One of the main factors driving this evolution is the concern by all the stakeholders in the cloud ecosystem of vendor lock-in, which includes the barriers of proprietary interfaces, formats, and protocols employed by the cloud vendors. As an increasing number of organizations and enterprises formulate cloud adoption strategies and execution plans, requirements of open, inter-operable standards for cloud management interfaces and protocols, data formats and so on will emerge. This will put pressure on cloud providers to build their offering on open interoperable standards to be considered as a candidate by enterprises. The big cloud vendors like Amazon, Google, and Microsoft, who currently do not actively participate in these efforts. True interoperability across the board in the near future seems unlikely. However, if achieved, it could lead to facilitation of advanced scenarios and thus drive the mainstream adoption of the enterprise cloud computing paradigm.

Another reason standards-based cloud offers are critical for the evolution and spread of this paradigm is the fact that standards drive choice and choice drives the market. From another perspective, in the presence of standards-based cloud offers, third party vendors will be able to develop and offer value added management capabilities in the form of independent cloud management tools.

Moreover, vendors with existing IT management tools in the market would be able to extend these tools to manage cloud solutions, hence facilitating organizations to preserve their existing investments in IT management solutions and use them for managing their hybrid cloud deployments.

Part of preserving investments is maintaining the assurance that cloud resources and services powering the business operations perform according to the business requirements. Underperforming resources or service disruptions lead to business and financial loss, reduced business credibility, reputation, and marginalized user productivity.

In the face of lack of control over the environment in which the resources and services are operating, enterprise would like sufficient assurances and guarantees to eliminate performance issues, and lack of compliance to security or governance standards (e.g. PCI, HIPAA, SOX, etc.) which can potentially lead to service disruptions, business loss, or damaged reputation. Service level agreements (SLA) can prove to be a useful instrument in facilitating enterprises' trust in cloud-based services.

Currently, the cloud solutions come with primitive or non existing SLAs. This is surely bound to change; as the cloud market gets crowded with increasing number of cloud offers, providers have to gain some competitive differentiation to capture larger share of the market. This is particularly true for market segments represented by enterprises and large organizations. Enterprise will be particularly interested to choose the offering with sophisticated SLAs providing assurances for the issues mentioned above.

Another important factor in this regard is lack of insights into the performance and health of the resources and service deployed on the cloud, such that this is another area of technology evolution that will be pushed. Currently, cloud providers don't offer sophisticated monitoring and reporting capabilities which can allow customers to comprehend and analyze the operations of these resources and services. However, recently, solutions have started to emerge to address this issue [32 34]. Nonetheless, this is one of the areas where cloud providers need to improve their offerings. It is believed that the situation will then improve because the enterprise cloud adoption phenomenon will make it imperative for the cloud providers to deliver sophisticated monitoring and reporting capabilities for the customers. This requirement would become ever more critical with the introduction of sophisticated SLAs, because customers would like to get insights into the service and resource behaviors for detecting SLA compliance violations. Moreover, cloud providers would need to expose this information through a standardized programmatic interface so customers can feed this information into their planning tools. Another important advancement that would emerge is to enable third-party independent vendors to measure the performance and health of resources and services deployed on cloud. This would prove to be a critical capability empowering third-party organizations to act as independent auditors especially with respect to SLA compliance auditing and for mediating the SLA penalty related issues.

Looking into the cloud services stack (IaaS, PaaS, SaaS) , the applications space or SaaS has the most growth potential. As forecasted by the analyst IDC , applications will account for 38% of \$44.2 billion cloud services market by 2103. Enterprises have already started to adopt some SaaS based solutions; however, these are primarily the edge applications like supplier management, talent management, performance management and so on as compared to the core business processes.

These SaaS based applications need to be integrated to the backed applications located on-premise. These integration capabilities would drive the mainstream SaaS adoption by enterprises. Moreover, organizations would opt for SaaS applications from multiple service providers to cater for various operational segments of an enterprise. This adds an extra dimension of complexity because the integration mechanisms need to weave SaaS application from various providers and eventually integrate them to the on-premise core business applications seamlessly.

Another emerging trend in the cloud application space is the divergence from the traditional RDBMS based data store backend. Cloud computing has given rise to alternative data storage technologies (Amazon Dynamo, Facebook Cassandra, Google BigTable, etc.) based on key-type storage models as compared to the relational model, which has been the mainstream choice for data storage for enterprise applications.

Recently launched NoSQL movement is gaining momentum, and enterprise application developers will start adopting these alternative data storage technologies as a data layer for these enterprise applications.

The platform services segment of the cloud market is still in its early phases. Currently, PaaS is predominantly used for developing and deploying situational applications to exploit the rapid development cycles especially to cope with the scenarios that are constrained by limited timeframe to bring the solutions to the market.

However, most of the development platforms and tools addressing this market segment are delivered by small startups and are proprietary technologies. Since the technologies are still evolving, providers are focusing on innovation aspects and gaining competitive edge over other providers. As these technologies evolve into maturity, the PaaS market will consolidate into a smaller number of service providers.

Moreover, big traditional software vendors will also join this market which will potentially trigger this consolidation through acquisitions and mergers. These views are along the lines of the research published by Gartner [36]. Key findings published in this report were that through 2011, development platforms and tools targeting cloud deployment will remain highly proprietary and until then, the focus of these service providers would be on innovation over market viability. Gartner predicts that from 2011 to 2015 market competition and maturing developer practises will drive consolidation around a small group of industry-dominant cloud technology providers.

The IaaS segment is typically attractive for small companies or startups that don't have enough capital and human resources to afford internal infrastructures. However, enterprises and large

organizations are experimenting with external cloud infrastructure providers as well. According to a Forrester report published last year , enterprises were experimenting with IaaS in various contexts for examples R&D-type projects for testing new services and applications and low-priority business applications.

The report also quotes a multi-national telecommunication company running an internal cloud for wikis and intranet sites and was beginning to test mission critical applications. The report also quotes the same enterprise to have achieved 30% cost reduction by adopting the cloud computing model.

However, we will see this trend adopted by an increasing number of enterprises opting for IaaS services. A recent Forrester report published in May 2009 supports this claim as according to the survey, 25% enterprises are either experimenting or thinking about adopting external cloud providers various types of enterprise applications and workloads.

As more and more vendors enter the IaaS cloud segment, cloud providers will strive to gain competitive advantage by adopting various optimization strategies or value added services to the customers. Open standards based cloud interfaces will gain attraction for increasing the like-lihood of being chosen by enterprises.

Cloud providers will provide transpar-ency into their operations and environments through sophisticated monitoring and reporting capabilities for the consumer to track and control their costs based on the consumption and usage information.

A recent report published by Gartner presents an interesting perspective on cloud evolution. The report argues that as cloud services proliferate, services would become complex to be handled

directly by the consumers. To cope with these scenarios, meta-services or cloud brokerage services will emerge. These brokerages will use several types of brokers and platforms to enhance service delivery and, ultimately service value.

According to Gartner, before these scenarios can be enabled, there is a need for brokerage business to use these brokers and platforms. According to Gartner, the following types of cloud service brokerages (CSB) are foreseen:

Cloud Service Intermediation. An intermediation broker provides a service that directly enhances a given service delivered one or more service consumers, essentially on top of a given service to enhance a specific capability.

Aggregation. An aggregation brokerage service combines multiple services into one or more new services.

Cloud Service Arbitrage. These services will provide flexibility and opportunistic choices for the service aggregator.

The above shows that there is potential for various large, medium, and small organizations to become players in the enterprise cloud marketplace. The dynamics of such a marketplace are still to be explored as the enabling technologies and standards continue to mature.

BUSINESS DRIVERS TOWARD A MARKETPLACE FOR ENTERPRISE CLOUD COMPUTING

In order to create an overview of offerings and consuming players on the market, it is important to understand the forces on the market and motivations of each player. Porter offers a framework for the

industry analysis and business strategy development. Within this framework the actors, products, and business models are clarified and structured.

The Porter model consists of five influencing factors/views (forces) on the market (Figure). In the traditional economic model, competition among rival companies drives profits to zero, thus forcing companies to strive for a competitive advantage over their rivals. The intensity of rivalry on the market is traditionally influenced by industry-specific characteristics :

Rivalry: The amount of companies dealing with cloud and virtualization technology is quite high at the moment; this might be a sign for high

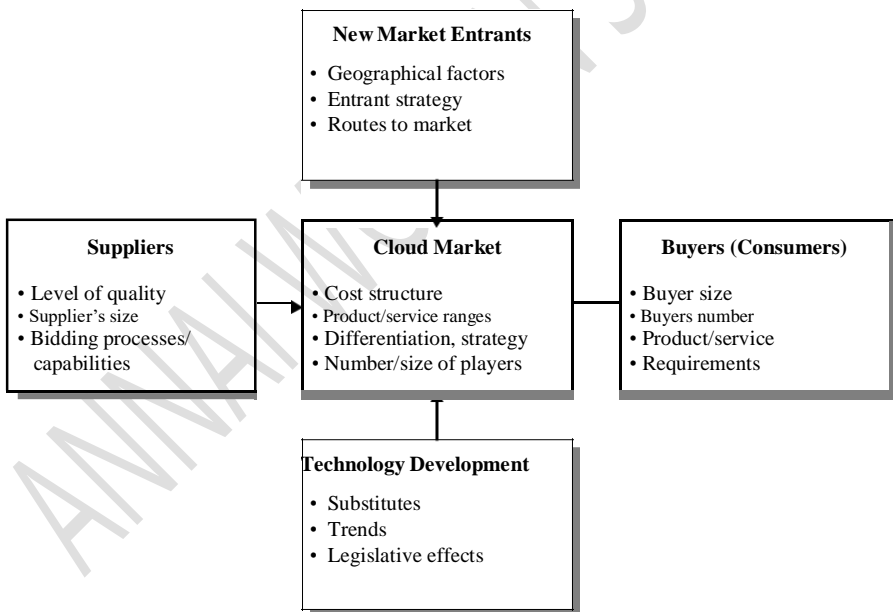


FIGURE .. Porter's five forces market model (adjusted for the cloud market) .

rivalry. But also the products and offers are quite various, so many niche products tend to become established.

Obviously, the cloud-virtualization market is presently booming and will keep growing during the next years. Therefore the fight for customers and struggle for market share will begin once the market becomes saturated and companies start offering comparable products.

The initial costs for huge data centers are enormous. By building up federations of computing and storing utilities, smaller companies can try to make use of this scale effect as well.

Low switching costs or high exit barriers influence rivalry. When a customer can freely switch from one product to another, there is a greater struggle to capture customers. From the opposite point of view high exit barriers discourage customers to buy into a new technology. The trends towards standardization of formats and architectures try to face this problem and tackle it. Most current cloud providers are only paying attention to standards related to the interaction with the end user. However, standards for clouds interoperability are still to be developed .

Monitoring the cloud market and observing current trends will show when the expected shakeout will take place and which companies will have the most accepted and economic offers by then . After this shakeout, the whole buzz and hype around cloud computing is expected to be over and mature solutions will evolve.

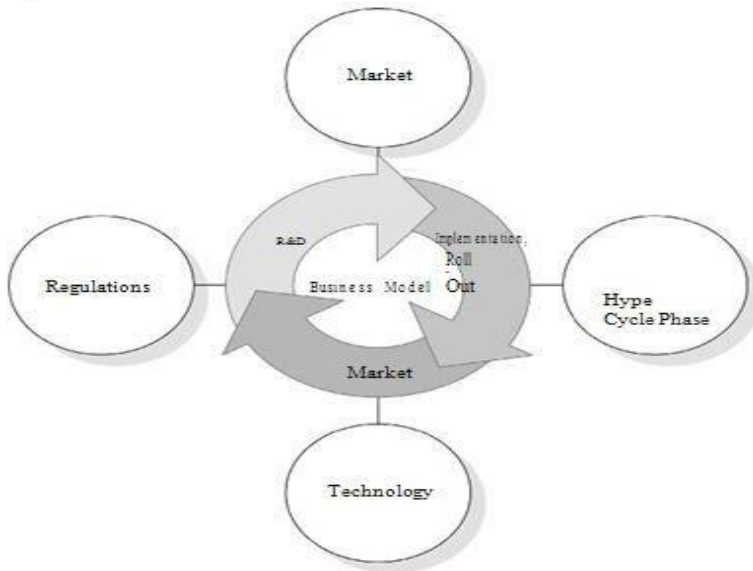
It is then that concrete business models will emerge. These business models will consider various fields, including e-business, strategy, supply chain management and information systems , but will now need to

emphasize the value of ICT-driven innovations for organizations and users

Furthermore, static perspectives on business models will not be viable in such an ICT-centric environment, given that organizations often have to review their business model in order to keep in line with fast changing environments like the cloud market for the ICT sector, from development to exploitation . With a few exceptions, most literature has taken a fairly static perspective on business models.

For dynamic business models for ICT, it is important to incorporate general phases of a product development. Thus, phasing models help to understand how innovation and change affect the evolution of the markets, and its consequences for company strategies and business models [50]. As argued by Kijl [51], the three main phases are R&D, implementation/roll-out, and market phase, which include the subphases of market offerings, maturity, and decline. These three main phases, influencing the business model, are used in a framework, visualized in Figure 4.5.

Figure also outlines which external drivers are expected to play a dominant role throughout the phases. Technology is the most important driver for the development of new business models in the ICT sector and will undoubtedly continue to be a major influencer of the enterprise cloud computing evolution. However, it can be assumed that market developments and regulation can also trigger opportunities for the development of new products and services in this paradigm. Changes in market opportunities or regulation enable new product and/or service definitions as well as underlying business models. There are already various players in the cloud computing market offering various services



Dynamic business models (based on [49] extend by influence factors identified).

However, they still struggle for market share and it is very likely that they will diversify their offers in order to meet all the market requirements. During these efforts, some of them will reach the mainstream and achieve a critical mass for the market while others will pass away or exist as niche offers after the shakeout. It is increasingly necessary to have a comprehensive model of drivers for business model dynamics , including knowledge of actors, products and market. This is also motivated by Porter , Kijl, and Bouwman and MacInnes. How then would such a business model be manifested?

THE CLOUD SUPPLY CHAIN

One indicator of what such a business model would look like is in the complexity of deploying, securing, interconnecting and maintaining enterprise landscapes and solutions such as ERP, The concept of a Cloud Supply Chain (C-SC) and hence Cloud Supply Chain Management (C-

SCM) appear to be viable future business models for the enterprise cloud computing paradigm. The idea of C-SCM represents the management of a network of interconnected businesses involved in the end-to-end provision of product and service packages required by customers. The established understanding of a supply chain is two or more parties linked by a flow of goods, information, and funds .

A specific definition for a C-SC is hence: “two or more parties linked by the provision of cloud services, related information and funds.” Figure represents a concept for the C-SC, showing the flow of products along different organizations such as hardware suppliers, soft-ware component suppliers, data center operators, distributors and the end customer.

Figure also makes a distinction between innovative and functional products in the C-SC. Fisher classifies products primarily on the basis of their demand patterns into two categories: primarily functional or primarily innovative [57]. Due to their stability, functional products favor competition, which leads to low profit margins and, as a consequence of their properties, to low inventory costs, low product variety, low stockout costs, and low obsolescence [58], [57]. Innovative products are characterized by additional (other) reasons for a customer in addition to basic needs that lead to purchase, unpredictable demand (that is high uncertainties, difficult to forecast and variable demand), and short product life cycles (typically 3 months to 1 year). Cloud services

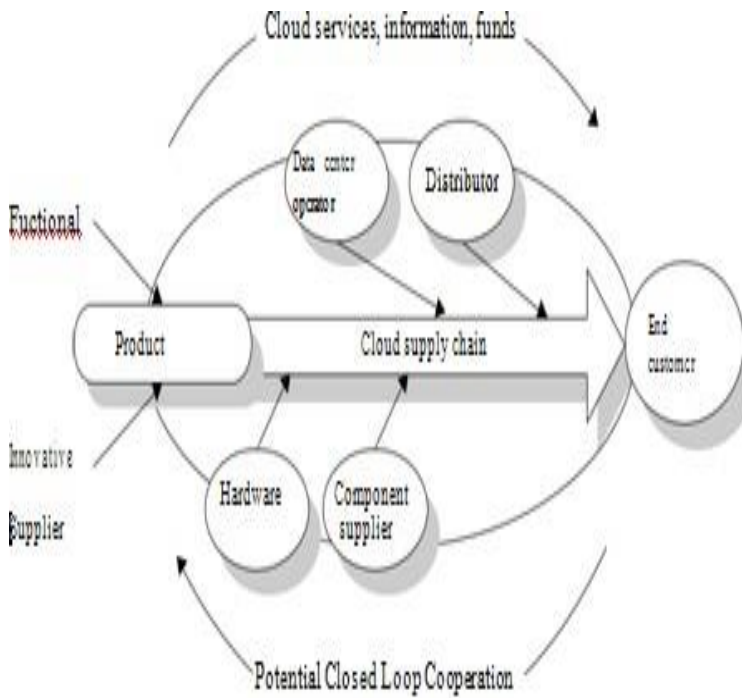


FIGURE : Cloud supply chain (C-SC).

should fulfill basic needs of customers and favor competition due to their reproducibility. They however also show characteristics of innovative products as the demand is in general unpredictable (on-demand business model) and have due to adjustments to competitors and changing market requirements very short development circles. Table presents a comparison of Traditional

TABLE . Comparison of Traditional and Emerging ICT Supply Chains^a

	Traditional Supply Chain Concepts		Emerging ICT Concepts
	Efficient SC	Responsive SC	Cloud SC
Primary goal	Supply demand at the lowest level of Cost	Respond quickly to demand (changes)	Supply demand at the lowest level of costs and respond quickly to demand
Product design Strategy	Maximize performance at the minimum product Cost	Create modularity to allow postponement of product differentiation	Create modularity to allow individual setting while maximizing the performance of services
Pricing strategy	Lower margins because price is a prime customer Driver	Higher margins, because price is not a prime customer driver	Lower margins, as high competition and comparable products
Manufacturing Strategy	Lower costs through high Utilization	Maintain capacity flexibility to meet unexpected demand	High utilization while flexible reaction on demand
Inventory Strategy	Minimize inventory to lower cost	Maintain buffer inventory to meet unexpected demand	Optimize of buffer for unpredicted demand, and best utilization
Lead time Strategy	Reduce but not at the expense of Costs	Aggressively reduce even if the costs are significant	Strong service-level agreements (SLA) for ad hoc provision
Supplier Strategy	Select based on cost and quality	Select based on speed, flexibility, and quantity	Select on complex optimum of speed, cost, and flexibility
Transportation Strategy	Greater reliance on low cost modes	Greater reliance on responsive modes	Implement highly responsive and low cost modes

Supply Chain concepts such as the efficient SC and responsive SC and a new concept for emerging ICT as the cloud computing area with cloud services as traded products.

This mixed characterization is furthermore reflected when it comes to the classification of efficient vs. responsive Supply Chains. Whereas functional products would preferable go into efficient Supply Chains, the main aim of responsive Supply Chains fits the categorization of innovative product. Cachon and Fisher show that within the supply chain the sharing of information (e.g. accounting and billing) is not the only contributor to SC cost, but it is the management and restructuring of services, information, and funds for an optimization of the chain that are expensive .

UNIT III

THE ANATOMY OF CLOUD INFRASTRUCTURES

There are many commercial IaaS cloud providers in the market, such as those cited earlier, and all of them share five characteristics: (i) They provide on-demand provisioning of computational resources; (ii) they use virtualization technologies to lease these resources; (iii) they provide public and simple remote interfaces to manage those resources; (iv) they use a pay-as-you-go cost model, typically charging by the hour; and (v) they operate data centers large enough to provide a seemingly unlimited amount of resources to their clients (usually touted as “infinite capacity” or “unlimited elasticity”).

Private and hybrid clouds share these same characteristics but, instead of selling capacity over publicly accessible interfaces, focus on providing capacity to an organization’s internal users.

Virtualization technologies have been the key enabler of many of these salient characteristics of IaaS clouds by giving providers a more flexible and generic way of managing their resources. Thus, virtual infrastructure (VI) management—the management of virtual machines distributed across a pool of physical resources—becomes a key concern when building an IaaS cloud and poses a number of challenges.

Like traditional physical resources, virtual machines require a fair amount of configuration, including preparation of the machine’s software environment and network configuration. However, in a virtual infrastructure, this configuration must be done on-the-fly, with

as little time between the time the VMs are requested and the time they are available to the user.

This is further complicated by the need to configure groups of VMs that will provide a specific service (e.g., an application requiring a Web server and a database server). Additionally, a virtual infrastructure manager must be capable of allocating resources efficiently, taking into account an organization's goals (such as minimizing power consumption and other operational costs) and reacting to changes in the physical infrastructure.

Virtual infrastructure management in private clouds has to deal with an additional problem: Unlike large IaaS cloud providers, such as Amazon, private clouds typically do not have enough resources to provide the illusion of "infinite capacity." The immediate provisioning scheme used in public clouds, where resources are provisioned at the moment they are requested, is ineffective in private clouds. Support for additional provisioning schemes, such as best-effort provisioning and advance reservations to guarantee quality of service (QoS) for applications that require resources at specific times (e.g., during known "spikes" in capacity requirements), is required. Thus, efficient resource allocation algorithms and policies and the ability to combine both private and public cloud resources, resulting in a hybrid approach, become even more important.

However, managing virtual infra-structures in a private/hybrid cloud is a different, albeit similar, problem than managing a virtualized data center, and existing tools lack several features that are required for building IaaS clouds. Most notably, they exhibit

monolithic and closed structures and can only operate, if at all, with some preconfigured placement policies, which are generally simple (round robin, first fit, etc.) and based only on CPU speed and utilization of a fixed and predetermined number of resources, such as memory and network bandwidth. This precludes extending their resource management strategies with custom policies or integration with other cloud systems, or even adding cloud interfaces.

Thus, there are still several gaps in existing VI solutions. Filling these gaps will require addressing a number of research challenges over the next years, across several areas, such as virtual machine management, resource scheduling, SLAs, federation of resources, and security. In this chapter, we focus on three problems addressed by the Virtual Machine Management Activity of RESER-VOIR: distributed management of virtual machines, reservation-based provisioning of virtualized resource, and provisioning to meet SLA commitments.

1. Distributed Management of Virtual Machines

The first problem is how to manage the virtual infrastructures themselves. Although resource management has been extensively studied, particularly for job management in high-performance computing, managing VMs poses additional problems that do not arise when managing jobs, such as the need to set up custom software environments for VMs, setting up and managing networking for interrelated VMs, and reducing the various overheads involved in using VMs. Thus, VI managers must be able to efficiently orchestrate all these different tasks.

The problem of efficiently selecting or scheduling computational resources is well known. However, the state of the art in VM-based resource scheduling follows a static approach, where resources are initially selected using a greedy allocation strategy, with minimal or no support for other placement policies. To efficiently schedule resources, VI managers must be able to support flexible and complex scheduling policies and must leverage the ability of VMs to suspend, resume, and migrate.

This complex task is one of the core problems that the RESERVOIR project tries to solve. In Section 6.2 we describe the problem of how to manage VMs distributed across a pool of physical resources and describe OpenNebula, the virtual infrastructure manager developed by the RESERVOIR project.

2.Reservation-Based Provisioning of Virtualized Resources

A particularly interesting problem when provisioning virtual infrastructures is how to deal with situations where the demand for resources is known before-hand—for example, when an experiment depending on some complex piece of equipment is going to run from 2 pm to 4 pm, and computational resources must be available at exactly that time to process the data produced by the equipment. Commercial cloud providers, such as Amazon, have enough resources to provide the illusion of infinite capacity, which means that this situation is simply resolved by requesting the resources exactly when needed; if capacity is “infinite,” then there will be resources available at 2 pm.

3.Provisioning to Meet SLA Commitments

IaaS clouds can be used to deploy services that will be consumed by users other than the one that deployed the services. For example, a company might depend on an IaaS cloud provider to deploy three-tier applications (Web front-end, application server, and database server) for its customers. In this case, there is a distinction between the cloud consumer (i.e., the service owner; in this case, the company that develops and manages the applications) and the end users of the resources provisioned on the cloud (i.e., the service user; in this case, the users that access the applications). Furthermore, service owners will enter into service-level agreements (SLAs) with their end users, covering guarantees such as the timeliness with which these services will respond.

However, cloud providers are typically not directly exposed to the service semantics or the SLAs that service owners may contract with their end users. The capacity requirements are, thus, less predictable and more elastic. The use of reservations may be insufficient, and capacity planning and optimizations are required instead. The cloud provider's task is, therefore, to make sure that resource allocation requests are satisfied with specific probability and timeliness. These requirements are formalized in infrastructure SLAs between the service owner and cloud provider, separate from the high-level SLAs between the service owner and its end users.

In many cases, either the service owner is not resourceful enough to perform an exact service sizing or service workloads are hard to

anticipate in advance. Therefore, to protect high-level SLAs, the cloud provider should cater for elasticity on demand. We argue that scaling and de-scaling of an application is best managed by the application itself. The reason is that in many cases, resources allocation decisions are application-specific and are being driven by the application level metrics. These metrics typically do not have a universal meaning and are not observable using black box monitoring of virtual machines comprising the service.

RESERVOIR proposes a flexible framework where service owners may register service-specific elasticity rules and monitoring probes, and these rules are being executed to match environment conditions. We argue that elasticity of the application should be contracted and formalized as part of capacity availability SLA between the cloud provider and service owner. This poses interesting research issues on the IaaS side, which can be grouped around two main topics:

SLA-oriented capacity planning that guarantees that there is enough capacity to guarantee service elasticity with minimal over-provisioning. Continuous resource placement and scheduling optimization that lowers operational costs and takes advantage of available capacity transparently to the service while keeping the service SLAs.

DISTRIBUTED MANAGEMENT OF VIRTUAL INFRASTRUCTURES

Managing VMs in a pool of distributed physical resources is a key concern in IaaS clouds, requiring the use of a virtual infrastructure manager. To address some of the shortcomings in existing VI solutions, we have developed the open source OpenNebula¹ virtual infrastructure engine. OpenNebula is capable of managing groups of interconnected VMs—with support for the Xen, KVM, and VMWare platforms—within data centers and private clouds that involve a large amount of virtual and physical servers. OpenNebula can also be used to build hybrid clouds by interfacing with remote cloud sites .

1. VM Model and Life Cycle

The primary target of OpenNebula is to manage VMs. Within OpenNebula, a VM is modeled as having the following 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. In general it might be necessary to transfer some of these image files to/from the physical machine the VM will be running in. A state file (optional) or recovery file that contains the memory image of a running VM plus some hypervisor-specific information.

The life cycle of a VM within OpenNebula follows several stages:

Resource Selection. Once a VM is requested to OpenNebula, a feasible placement plan for the VM must be made. OpenNebula's default scheduler provides an implementation of a rank scheduling policy, allowing site administrators to configure the scheduler to prioritize the resources that are more suitable for the VM, using information from the VMs and the physical hosts. As we will describe in Section 6.3, OpenNebula can also use the Haizea lease manager to support more complex scheduling policies.

Resource Preparation. The disk images of the VM are transferred to the target physical resource. During the boot process, the VM is contextualized, a process where the disk images are specialized to work in a given environment. For example, if the VM is part of a group of VMs offering a service (a compute cluster, a DB-based application, etc.), contextualization could involve setting up the network and the machine hostname, or registering the new VM with a service (e.g., the head node in a compute cluster). Different techniques are available to contextualize a worker node, including use of an automatic installation system (for instance, Puppet or Quattor), a context server (see reference 15), or access to a disk image with the context data for the worker node (OVF recommendation).

VM Creation. The VM is booted by the resource hypervisor.

VM Migration. The VM potentially gets migrated to a more suitable resource (e.g., to optimize the power consumption of the physical resources). **VM Termination.** When the VM is going to shut down, OpenNebula can transfer back its disk images to a known location. This way, changes in the VM can be kept for a future use.

2. VM Management

OpenNebula manages a VMs life cycle by orchestrating three different management areas: virtualization by interfacing with a physical resource's hypervisor, such as Xen, KVM, or VMWare, to control (e.g., boot, stop, or shutdown) the VM; image management by transferring the VM images from an image repository to the selected resource and by creating on-the-fly temporary images; and networking by creating local area networks (LAN) to interconnect the VMs and tracking the MAC addresses leased in each network.

Virtualization. OpenNebula manages VMs by interfacing with the physical resource virtualization technology (e.g., Xen or KVM) using a set of pluggable drivers that decouple the managing process from the underlying technology. Thus, whenever the core needs to manage a VM, it uses high-level commands such as “start VM,” “stop VM,” and so on, which are translated by the drivers into commands that the virtual machine manager can understand. By decoupling the OpenNebula core from the virtualization technologies through the use of a driver-based architecture, adding support for additional virtual machine managers only requires writing a driver for it.

Image Management. VMs are supported by a set of virtual disks or images, which contains the OS and any other additional software needed by the VM. OpenNebula assumes that there is an image repository that can be any storage medium or service, local or remote, that holds the base image of the VMs. There are a number of different possible configurations depending on the user's needs. For example, users may want all their images placed on a separate repository with

only HTTP access. Alternatively, images can be shared through NFS between all the hosts. OpenNebula aims to be flexible enough to support as many different image management configurations as possible. OpenNebula uses the following concepts for its image management model (Figure):

Image Repositories refer to any storage medium, local or remote, that hold the base images of the VMs. An image repository can be a dedicated file server or a remote URL from an appliance provider, but they need to be accessible from the OpenNebula front-end.

Virtual Machine Directory is a directory on the cluster node where a VM is running. This directory holds all deployment files for the hypervisor to boot the machine, checkpoints, and images being used or saved—all of them specific to that VM. This directory should be shared for most hypervisors to be able to perform live migrations. Any given VM image goes through the following steps along its life cycle:

Preparation implies all the necessary changes to be made to the machine's image so it is prepared to offer the service to which it is intended. OpenNebula assumes that the images that conform to a particular VM are prepared and placed in the accessible image repository.

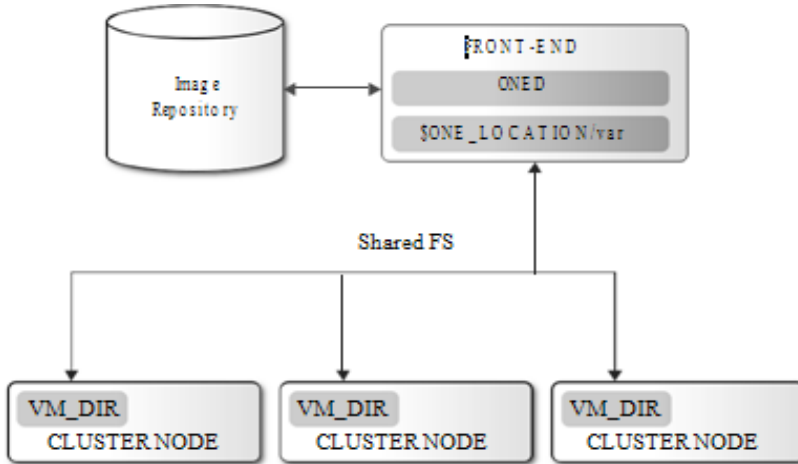


FIGURE . Image management in OpenNebula.

Cloning the image means taking the image from the repository and placing it in the VM's directory in the physical node where it is going to be run before the VM is actually booted. If a VM image is to be cloned, the original image is not going to be used, and thus a copy will be used. There is a qualifier (clone) for the images that can mark them as targeting for cloning or not.

Save/remove. If the save qualifier is disabled, once the VM has been shut down, the images and all the changes thereof are going to be disposed of. However, if the save qualifier is activated, the image will be saved for later use.

Networking. In general, services deployed on a cloud, from a computing cluster to the classical three-tier business application, require several inter-related VMs, with a virtual application network (VAN) being the primary link between them. OpenNebula dynamically creates these VANs and tracks the MAC addresses

leased in the network to the service VMs. Note that here we refer to layer 2 LANs; other TCP/IP services such as DNS, NIS, or NFS are the responsibility of the service (i.e., the service VMs have to be configured to provide such services).

The physical hosts that will co-form the fabric of our virtual infrastructures will need to have some constraints in order to effectively deliver virtual networks to our virtual machines. Therefore, from the point of view of networking, we can define our physical cluster as a set of hosts with one or more network interfaces, each of them connected to a different physical network.

3. Further Reading on OpenNebula

There are a number of scholarly publications that describe the design and architecture of OpenNebula in more detail, including papers showing performance results obtained when using OpenNebula to deploy and manage the back-end nodes of a Sun Grid Engine compute cluster [14] and of a NGINX Web server [16] on both local resources and an external cloud. The Open-Nebula virtual infrastructure engine is also available for download at <http://www.opennebula.org/>, which provides abundant documentation not just on how to install and use OpenNebula, but also on its internal architecture.

SCHEDULING TECHNIQUES FOR ADVANCE RESERVATION OF CAPACITY

While a VI manager like OpenNebula can handle all the minutiae of managing VMs in a pool of physical resources, scheduling these VMs efficiently is a different and complex matter. Commercial cloud providers, such as Amazon, rely on an immediate provisioning model where VMs are provisioned right away, since their data centers' capacity is assumed to be infinite.

Thus, there is no need for other provisioning models, such as best-effort provisioning where requests have to be queued and prioritized or advance provisioning where resources are pre-reserved so they will be guaranteed to be available at a given time period; queuing and reservations are unnecessary when resources are always available to satisfy incoming requests.

1. Existing Approaches to Capacity Reservation

Efficient reservation of resources in resource management systems has been studied considerably, particularly in the context of job scheduling. In fact, most modern job schedulers support advance reservation of resources, but their implementation falls short in several aspects. First of all, they are constrained by the job abstraction;

when a user makes an advance reservation in a job-based system, the user does not have direct and unfettered access to the resources, the way a cloud users can access the VMs they requested, but, rather, is only allowed to submit jobs to them. For example, PBS Pro creates a new queue that will be bound to the reserved resources,

guaranteeing that jobs submitted to that queue will be executed on them (assuming they have permission to do so).

Maui and Moab, on the other hand, simply allow users to specify that a submitted job should use the reserved resources (if the submitting user has permission to do so). There are no mechanisms to directly login to the reserved resources, other than through an interactive job, which does not provide unfettered access to the resources.

However, although many modern schedulers support at least checkpointing-based preemption, this requires the job's executable itself to be checkpointable. An application can be made checkpointable by explicitly adding that functionality to an application (application-level and library-level checkpointing) or transparently by using OS-level checkpointing, where the operating system (such as Cray, IRIX, and patched versions of Linux using BLCR [17]) checkpoints a process, without rewriting the program or relinking it with checkpointing libraries. However, this requires a checkpointing-capable OS to be available.

Thus, a job scheduler capable of checkpointing-based preemption and migration could be used to checkpoint jobs before the start of an advance reservation, minimizing their impact on the schedule. However, the application- and library-level checkpointing approaches burden the user with having to modify their applications to make them checkpointable, imposing a restriction on the software environment.

OS-level checkpointing, on the other hand, is a more appealing option, but still imposes certain software restrictions on resource consumers. Systems like Cray and IRIX still require applications to be compiled for their respective architectures, which would only allow a small fraction of existing applications to be supported within leases, or would require existing applications to be ported to these architectures. This is an excessive restriction on users, given the large number of clusters and applications that depend on the x86 architecture. Although the BLCR project does provide a checkpointing x86 Linux kernel, this kernel still has several limitations, such as not being able to properly checkpoint network traffic and not being able to checkpoint MPI applications unless they are linked with BLCR-aware MPI libraries.

An alternative approach to supporting advance reservations was proposed by Nurmi et al. [18], which introduced “virtual advance reservations for queues” (VARQ). This approach overlays advance reservations over traditional job schedulers by first predicting the time a job would spend waiting in a scheduler’s queue and then submitting a job (representing the advance reservation) at a time such that, based on the wait time prediction, the probability that it will be running at the start of the reservation is maximized.

Since no actual reservations can be done, VARQ jobs can run on traditional job schedulers, which will not distinguish between the regular best-effort jobs and the VARQ jobs. Although this is an interesting approach that can be realistically implemented in practice (since it does not require modifications to existing scheduler), it still depends on the job abstraction.

Hovesta proposed a planning-based (as opposed to queuing-based) approach to job scheduling, where job requests are immediately planned by making a reservation (now or in the future), instead of waiting in a queue.

Thus, advance reservations are implicitly supported by a planning-based system. Additionally, each time a new request is received, the entire schedule is reevaluated to optimize resource usage. For example, a request for an advance reservation can be accepted without using preemption, since the jobs that were originally assigned to those resources can be assigned to different resources (assuming the jobs were not already running).

2. Reservations with VMs

As we described earlier, virtualization technologies are a key enabler of many features found in IaaS clouds. Virtual machines are also an appealing vehicle for implementing efficient reservation of resources due to their ability to be suspended, potentially migrated, and resumed without modifying any of the applications running inside the VM. However, virtual machines also raise additional challenges related to the overhead of using VMs:

Preparation Overhead. When using VMs to implement reservations, a VM disk image must be either prepared on-the-fly or transferred to the physical node where it is needed. Since a VM disk image can have a size in the order of gigabytes, this preparation overhead can significantly delay the starting time of leases. This delay may, in some cases, be unacceptable for advance reservations that must start at a specific time.

Runtime Overhead. Once a VM is running, scheduling primitives such as checkpointing and resuming can incur in significant overhead

since a VM's entire memory space must be saved to disk, and then read from disk. Migration involves transferring this saved memory along with the VM disk image. Similar to deployment overhead, this overhead can result in noticeable delays.

3. Leasing Model

We define a lease as “a negotiated and renegotiable agreement between a resource provider and a resource consumer, where the former agrees to make a set of resources available to the latter, based on a set of lease terms presented by the resource consumer.”

The terms must encompass the following: the hardware resources required by the resource consumer, such as CPUs, memory, and network bandwidth; a software environment required on the leased resources; and an availability period during which a user requests that the hardware and software resources be available.

Thus, we consider the following availability terms:

Start time may be unspecified (a best-effort lease) or specified (an advance reservation lease). In the latter case, the user may specify either a specific start time or a time period during which the lease start may occur.

Maximum duration refers to the total maximum amount of time that the leased resources will be available.

Leases can be preemptable. A preemptable lease can be safely paused without disrupting the computation that takes place inside the lease.

Haizea's resource model considers that it manages W physical nodes capable of running virtual machines. Each node i has CPUs, megabytes (MB) of memory, and MB of local disk storage. We assume that all disk images required to run virtual machines are

available in a repository from which they can be transferred to nodes as needed and that all are connected at a bandwidth of B MB/sec by a switched network.

A lease is implemented as a set of N VMs, each allocated resources described by a tuple (p, m, d, b) , where p is number of CPUs, m is memory in MB, d is disk space in MB, and b is network bandwidth in MB/sec.

A disk image I with a size of $\text{size}(I)$ MB must be transferred from the repository to a node before the VM can start. When transferring a disk image to multiple nodes, we use multi-casting and model the transfer time as $\text{size}(I)/B$.

If a lease is preempted, it is suspended by suspending its VMs, which may then be either resumed on the same node or migrated to another node and resumed there. Suspending a VM results in a memory state image file (of size m that can be saved to either a local filesystem or a global filesystem ($f \in \{\text{local}, \text{global}\}$)).

Resumption requires reading that image back into memory and then discarding the file. Suspension of a single VM is done at a rate of s megabytes of VM memory per second, and we define r similarly for VM resumption.

4. Lease Scheduling

Haizea is designed to process lease requests and determine how those requests can be mapped to virtual machines, leveraging their suspend/resume/migrate capability, in such a way that the leases' requirements are satisfied. The scheduling component of Haizea uses classical backfilling algorithms, extended to allow best-effort leases to be preempted if resources have to be freed up for advance reservation requests.

Additionally, to address the pre-paration and runtime overheads mentioned earlier, the scheduler allocates resources explicitly for the overhead activities (such as transferring disk images or suspending VMs) instead of assuming they should be deducted from the lease's allocation.

Besides guaranteeing that certain operations complete on time (e.g., an image transfer before the start of a lease), the scheduler also attempts to minimize this overhead whenever possible, most notably by reusing disk image transfers and caching disk images on the physical nodes.

Best-effort leases are scheduled using a queue. When a best-effort lease is requested, the lease request is placed at the end of the queue, which is periodically evaluated using a backfilling algorithm—both aggressive and conservative backfilling strategies are supported—to determine if any leases can be scheduled.

The scheduler does this by first checking the earliest possible starting time for the lease on each physical node, which will depend on the required disk images. For example, if some physical nodes have cached the required disk image, it will be possible to start the lease earlier on those nodes. Once these earliest starting times have been determined, the scheduler chooses the nodes that allow the lease to start soonest.

The use of VM suspension/resumption allows the best-effort leases to be scheduled even if there are not enough resources available for their full requested duration. If there is a “blocking” lease in the future, such as an advance reservation lease that would prevent the best-effort lease to run to completion before the blocking lease starts,

the best-effort lease can still be scheduled; the VMs in the best-effort lease will simply be suspended before a blocking lease. The remainder of a suspended lease is placed in the queue, according to its submission time, and is scheduled like a regular best-effort lease (except a resumption operation, and potentially a migration operation, will have to be scheduled too).

Advance reservations, on the other hand, do not go through a queue, since they must start at either the requested time or not at all. Thus, scheduling this type of lease is relatively simple, because it mostly involves checking if there are enough resources available during the requested interval.

However, the scheduler must also check if any associated overheads can be scheduled in such a way that the lease can still start on time. For preparation overhead, the scheduler determines if the required images can be transferred on time.

These transfers are scheduled using an earliest deadline first (EDF) algorithm, where the deadline for the image transfer is the start time of the advance reservation lease. Since the start time of an advance reservation lease may occur long after the lease request, we modify the basic EDF algorithm so that transfers take place as close as possible to the deadline, preventing images from unnecessarily consuming disk space before the lease starts. For runtime overhead, the scheduler will attempt to schedule the lease without having to preempt other leases; if preemption is unavoidable, the necessary suspension operations are scheduled if they can be performed on time.

For both types of leases, Haizea supports pluggable policies, allowing system administrators to write their own scheduling policies without having to modify Haizea's source code. Currently, three policies are pluggable in Haizea: determining whether a lease is accepted or not, the selection of physical nodes, and determining whether a lease can preempt another lease.

Our main results so far have shown that, when using workloads that combine best-effort and advance reservation lease requests, a VM-based approach with suspend/resume/migrate can overcome the utilization problems typically associated with the use of advance reservations. Even in the presence of the runtime overhead resulting from using VMs, a VM-based approach results in consistently better total execution time than a scheduler that does not support task preemption, along with only slightly worse performance than a scheduler that does support task preemption.

Measuring the wait time and slowdown of best-effort leases shows that, although the average values of these metrics increase when using VMs, this effect is due to short leases not being preferentially selected by Haizea's backfilling algorithm, instead of allowing best-effort leases to run as long as possible before a preempting AR lease (and being suspended right before the start of the AR).

In effect, a VM-based approach does not favor leases of a particular length over others, unlike systems that rely more heavily on backfilling. Our results have also shown that, although supporting the deployment of multiple software environments, in the form of multiple VM images, requires the transfer of potentially large disk

image files, this deployment overhead can be minimized through the use of image transfer scheduling and caching strategies.

5. Further Reading on Lease-Based Resource Management

There are several scholarly publications available for download at the Haizea Web site (<http://haizea.cs.uchicago.edu/>) describing Haizea's design and algorithms in greater detail and showing performance results obtained when using Haizea's lease-based model.

RVWS DESIGN

While Web services have simplified resource access and management, it is not possible to know if the resource(s) behind the Web service is (are) ready for requests. Clients need to exchange numerous messages with required Web services to learn the current activity of resources and thus face significant overhead loss if most of the Web services prove ineffective.

Furthermore, even in ideal circumstances where all resources behind Web services are the best choice, clients still have to locate the services themselves. Finally, the Web services have to be stateful so that they are able to best reflect the current state of their resources.

This was the motivation for creating the RVWS framework. The novelty of RVWS is that it combines dynamic attributes, stateful Web services (aware of their past activity), stateful and dynamic WSDL documents, and brokering into a single, effective, service-based framework. Regardless of clients accessing services directly or discovering them via a broker, clients of RVWS-based distributed systems spend less time learning of services.

1. Dynamic Attribute Exposure

There are two categories of dynamic attributes addressed in the RVWS framework: state and characteristic. State attributes cover the current activity of the service and its resources, thus indicating readiness. For example, a Web service that exposes a cluster (itself a complex resource) would most likely have a dynamic state attribute that indicates how many nodes in the cluster are busy and how many are idle.

Characteristic attributes cover the operational features of the service, the resources behind it, the quality of service (QoS), price and provider information. Again with the cluster Web service example, a possible characteristic is an array of support software within the cluster. This is important information as cluster clients need to know what software libraries exist on the cluster.

Figure shows the steps on how to make Web services stateful and how the dynamic attributes of resources are presented to clients via the WSDL document.

To keep the stateful Web service current, a Connector [2] is used to detect changes in resources and then inform the Web service. The Connector has three logical modules: Detection, Decision, and Notification. The Detection module routinely queries the resource for attribute information (1 2). Any changes in the attributes are passed to the Decision module (3) that decides if the attribute change is large enough to warrant a notification. This prevents excessive communication with the Web service. Updated attributes are passed on to the Notification module (4), which informs the stateful Web

service (5) that updates its internal state. When clients requests the stateful WSDL document

(6), the Web service returns the WSDL document with the values of all attributes (7) at the request time.

2.Stateful WSDL Document Creation

When exposing the dynamic attributes of resources, the RVWS framework allows Web services to expose the dynamic attributes through the WSDL documents of Web services. The Web Service Description Language (WSDL)

governs a schema that describes a Web service and a document written in the schema. In this chapter, the term WSDL refers to the stateless WSDL document. Stateful WSDL document refers to the WSDL document created by RVWS Web services.

All information of service resources is kept in a new WSDL section called Resources. Figure shows the structure of the Resources section with the rest of the WSDL document. For each resource behind the Web service, a ResourceInfo section exists.

Each ResourceInfo section has a resource-id attribute and two child sections: state and characteristic. All resources behind the Web service have unique identifiers. When the Connector learns of the resource for the first time, it publishes the resource to the Web service.

Both the state and characteristics elements contain several description elements, each with a name attribute and (if the provider

wishes) one or more attributes of the service. Attributes in RVWS use the {name: op value} notations. An example attribute is {cost: .5 \$5}.

The state of a resource could be very complex and cannot be described in just one attribute. For example, variations in each node in the cluster all contribute significantly to the state of the cluster. Thus the state in RVWS is described via a collection of attributes, all making up the whole state.

The characteristics section describes near-static attributes of resources such as their limitations and data parameters. For example, the type of CPU on a node in a cluster is described in this section.

3.Publication in RVWS

While the stateful WSDL document eliminates the overhead incurred from manually learning the attributes of the service and its resource(s), the issues behind discovering services are still unresolved.

To help ease the publication and discovery of required services with stateful WSDL documents, a Dynamic Broker was proposed (Figure 7.3) [17]. The goal of the Dynamic Broker is to provide an effective publication and discovery service based on service, resource, and provider dynamic attributes.

When publishing to the Broker (1), the provider sends attributes of the Web service to the Dynamic Broker. The dynamic attributes indicate the functionality, cost, QoS, and any other attributes the provider wishes to have published about the service. Furthermore, the

provider is able to publish information about itself, such as the provider's contact details and reputation.

After publication (1), the Broker gets the stateful WSDL document from the Web service (2). After getting the stateful WSDL document, the Dynamic Broker extracts all resource dynamic attributes from the stateful WSDL documents and stores the resource attributes in the resources store.

The Dynamic Broker then stores the (stateless) WSDL document and service attributes from (1) in the service store. Finally, all attributes about the provider are placed in the providers store.

As the Web service changes, it is able to send a notification to the Broker (3) which then updates the relevant attribute in the relevant store. Had all information about each service been kept in a single stateful WSDL document, the dynamic broker would have spent a lot of time load, thereby editing and saving huge XML documents to the database.

4. Automatic Discovery and Selection

The automatic service discovery that takes into consideration dynamic attributes in their WSDL documents allows service (e.g., a cluster) discovery.

When discovering services, the client submits to the Dynamic Broker three groups of requirements :service, resource, and provider. The Dynamic Broker compares each requirement group on the related data store (2).

Then, after getting matches, the Broker applies filtering (3). As the client using the Broker could vary from human operators to other software units, the resulting matches have to be filtered to suit the client. Finally, the filtered results are returned to the client (4).

The automatic service selection that takes into consideration dynamic attributes in their WSDL documents allows for both a single service (e.g., a cluster) selection and an orchestration of services to satisfy workflow requirements .

The SLA (service-level agreement) reached by the client and cloud service provider specifies attributes of services that form the client's request or workflow.

This is followed by the process of services' selection using Brokers. Thus, selection is carried out automatically and transparently.

In a system comprising many clouds, the set of attributes is partitioned over many distributed service databases, for autonomy, scalability, and performance.

The automatic selection of services is performed to optimize a function reflecting client requirements.

Time-critical and high-throughput tasks benefit by executing a computing intensive application on multiple clusters exposed as services of one or many clouds.

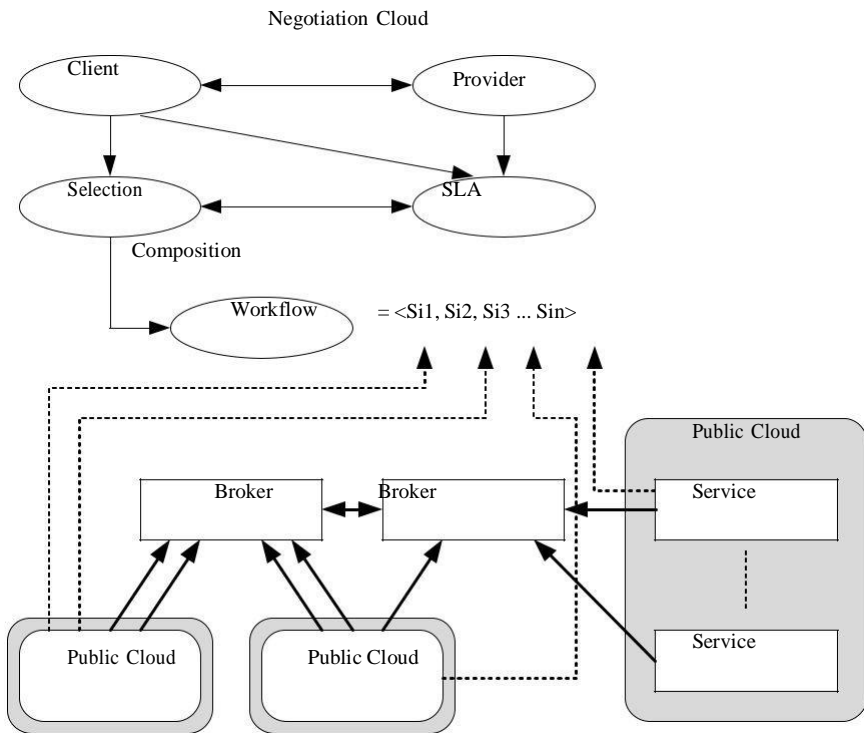


FIGURE 10. Dynamic discovery and selection.

The dynamic attribute information only relates to clients that are aware of them. Human clients know what the attributes are, owing to the section being clearly named. Software-client-designed pre-RVWS ignore the additional information as they follow the WSDL schema that we have not changed.

CLUSTER AS A SERVICE: THE LOGICAL DESIGN

Simplification of the use of clusters could only be achieved through higher layer abstraction that is proposed here to be implemented using the service-based Cluster as a Service (CaaS) Technology. The purpose of the CaaS Technology is to ease the publication, discovery, selection, and use of existing computational clusters.

1.CaaS Overview

The exposure of a cluster via a Web service is intricate and comprises several services running on top of a physical cluster. Figure 7.6 shows the complete CaaS technology.

A typical cluster is comprised of three elements: nodes, data storage, and middleware. The middleware virtualizes the cluster into a single system image; thus resources such as the CPU can be used without knowing the organization of the cluster. Of interest to this chapter are the components that manage the allocation of jobs to nodes (scheduler) and that monitor the activity of the cluster (monitor). As time progresses, the amount of free memory, disk space, and CPU usage of each cluster node changes. Information about how quickly the scheduler can take a job and start it on the cluster also is vital in choosing a cluster.

To make information about the cluster publishable, a Publisher Web service and Connector were created using the RVWS

framework. The purpose of the publisher Web service was to expose the dynamic attributes of the cluster via the stateful WSDL document. Furthermore, the Publisher service is published to the Dynamic Broker so clients can easily discover the cluster.

To find clusters, the CaaS Service makes use of the Dynamic Broker. While the Broker is detailed in returning dynamic attributes of matching services, the results from the Dynamic Broker are too detailed for the CaaS Service. Thus another role of the CaaS Service is to “summarize” the result data so that they convey fewer details.

Ordinarily, clients could find required clusters but they still had to manually transfer their files, invoke the scheduler, and get the results back. All three tasks require knowledge of the cluster and are conducted using complex tools. The role of the CaaS Service is to (i) provide easy and intuitive file transfer tools so clients can upload jobs and download results and (ii) offer an easy to use interface for clients to monitor their jobs. The CaaS Service does this by

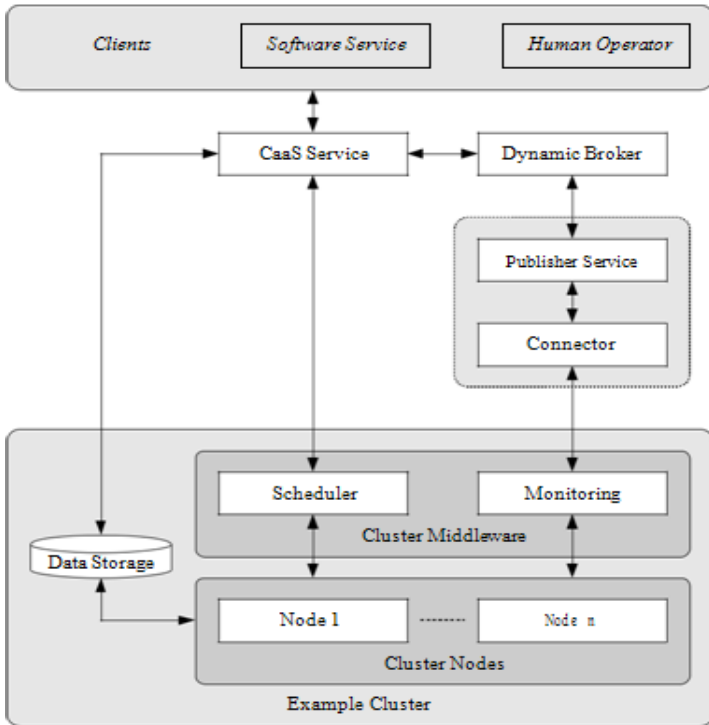


FIGURE Complete CaaS system.

allowing clients to upload files as they would any Web page while carrying out the required data transfer to the cluster transparently.

Because clients to the cluster cannot know how the data storage is managed, the CaaS Service offers a simple transfer interface to clients while addressing the transfer specifics. Finally, the CaaS Service communicates with the cluster's scheduler, thus freeing the client from needing to know how the scheduler is invoked when submitting and monitoring jobs.

2.Cluster Stateful WSDL Document

The purpose of the Publisher Web service is to expose the dynamic attributes of a cluster via a stateful WSDL document. Figure 7 shows the resources section to be added to the WSDL of the Publisher Web service.

Inside the state and characteristic elements, an XML element for each cluster node was created. The advantage of the XML structuring of our cluster intercommunicating modules.

Each module in the CaaS Service encapsulates one of the tasks and is able to communicate with other modules to extend its functionality.

Figure presents the modules with the CaaS Service and illustrates the dependencies between them.

The modules inside the CaaS Web service are only accessed through an interface. The use of the interface means the Web service can be updated over time without requiring clients to be updated nor modified.

Invoking an operation on the CaaS Service Interface (discovery, etc.) invokes operations on various modules. Thus, to best describe the role each module plays, the following sections outline the various tasks that the CaaS Service carries out.

3.CaaS Service Design

The CaaS service can be described as having four main tasks: cluster discovery and selection, result organization, job management, and file management. Based on these tasks, the CaaS Service has been designed using

TABLE . Cluster 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		
	node-count	Total number of nodes in the cluster. Node count differs from core-count as each node in a cluster can have many cores.	Generated	
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 Discovery. Before a client uses a cluster, a cluster must be discovered and selected first. Figure shows the workflow on finding a required cluster. To start, clients submit cluster requirements in the form of attribute values to the CaaS Service Interface (1). The requirements range from the number of nodes in the cluster to the installed software (both operating systems and software APIs). The CaaS Service Interface invokes the Cluster Finder module (2) that communicates with the Dynamic Broker (3) and returns service matches (if any).

To address the detailed results from the Broker, the Cluster Finder module invokes the Results Organizer module (4) that takes the Broker results and returns an organized version that is returned to the client (5 6). The organized

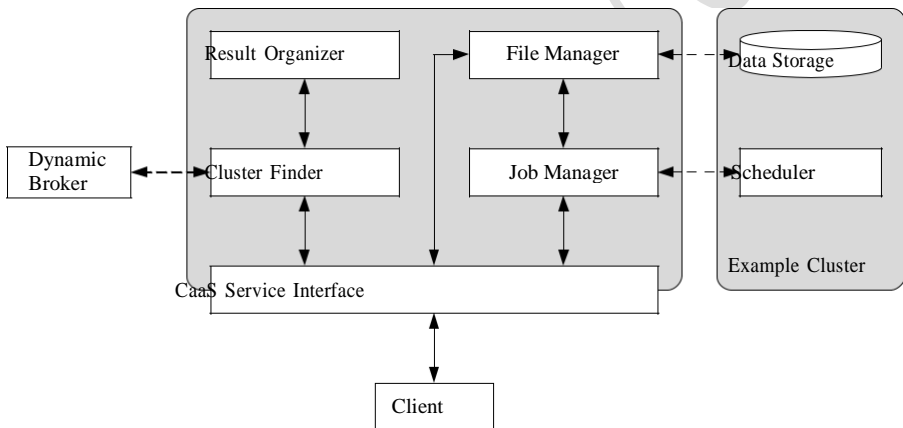


FIGURE . CaaS Service design.

results instruct the client what clusters satisfy the specified requirements. After reviewing the results, the client chooses a cluster.

Job Submission. After selecting a required cluster, all executables and data files have to be transferred to the cluster and the job submitted to the scheduler for execution. As clusters vary significantly in the software middleware used to create them, it can

be difficult to place jobs on the cluster. To do so requires knowing how jobs are stored and how they are queued for execution on the cluster. Figure shows how the CaaS Service simplifies the use of a cluster to the point where the client does not have to know about the underlying middleware.

All required data, parameters, such as estimated runtime, are uploaded to the CaaS Service (1). Once the file upload is complete, the Job Manager is invoked (2). It resolves the transfer of all files to the cluster by invoking the File Manager (3) that makes a connection to the cluster storage and commences the transfer of all files (4).

Upon completion of the transfer (4), the outcome is reported back to the Job Manager (5). On failure, a report is sent and the client can decide on the appropriate action to take. If the file transfer was successful, the Job Manager invokes the scheduler on the cluster (6).

The same parameters the client gave to the CaaS Service Interface are submitted to the scheduler; the only difference being that the Job Manager also informs the scheduler where the job is kept so it can be started. If the outcome of the scheduler (6) is successful, the client is then informed . The outcome includes the response from the scheduler, the job identifier the scheduler gave to the job, and any other information the scheduler provides.

Job Monitoring. During execution, clients should be able to view the execution progress of their jobs. Even though the cluster is not the owned by the client, the job is. Thus, it is the right of the client to see how the job is progressing and (if the client decides) terminate the job and remove it from the cluster.

Figure outlines the workflow the client takes when querying about job execution.

First, the client contacts the CaaS service interface (1) that invokes the Job Manager module (2). No matter what the operation is (check, pause, or terminate), the Job Manager only has to communicate with the scheduler (3) and reports back a successful outcome to the client (4 5).

Result Collection. The final role of the CaaS Service is addressing jobs that have terminated or completed their execution successfully. In both cases, error or data files need to be transferred to the client. Figure presents the workflow and CaaS Service modules used to retrieve error or result files from the cluster.

Clients start the error or result file transfer by contacting the CaaS Service Interface (1) that then invokes the File Manager (2) to retrieve the files from the cluster's data storage (3). If there is a transfer error, the File Manager attempts to resolve the issue first before informing the client. If the transfer of files (3) is successful, the files are returned to the CaaS Service Interface (4) and then the client (5). When returning the files, URL link or a FTP address is provided so the client can retrieve the files.

4. User Interface: CaaS Web Pages

The CaaS Service has to support at least two forms of client: software clients and human operator clients. Software clients could be other software applications or services and thus are able to communicate with the CaaS Service Interface directly.

For human operators to use the CaaS Service, a series of Web pages has been designed. Each page in the series covers a step in the process of discovering, selecting, and using a cluster. Figure 7.13 shows the Cluster Specification Web page where clients can start the discovery of a required cluster.

In Section A the client is able to specify attributes about the required cluster. Section B allows specifying any required software the cluster job needs. Afterwards, the attributes are then given to the CaaS service that performs a search for possible clusters and the results are displayed in a Select Cluster Web page .

Next, the client goes to the job specification page, Figure 7.15. Section A allows specifying the job. Section B allows the client to specify and upload all data files and job executables. If the job is complex, Section B also allows specifying a job script. Job scripts are script files that describe and manage

Section A: Hardware

Number of Nodes:

Amount of Memory: GB

Free Memory: GB

Disk Free: GB

CPU: GHz

Section B: Software

Operating System:

FIGURE : Web page for cluster specification.

	Cluster A <u>select</u>	Cluster B <u>select</u>
Hardware		
Number of Nodes :	<input checked="" type="checkbox"/>	
Amount of Memory :	<input checked="" type="checkbox"/>	
Free Memory :	<input checked="" type="checkbox"/>	
Disk Free :		<input checked="" type="checkbox"/>
CPU :	<input checked="" type="checkbox"/>	
Architecture :	<input checked="" type="checkbox"/>	
Speed		<input checked="" type="checkbox"/>
Software		
Operating System :	<input checked="" type="checkbox"/>	
Architecture :	<input checked="" type="checkbox"/>	
Version :	<input checked="" type="checkbox"/>	

FIGURE .: Web page for showing matching clusters.

various stages of a large cluster job. Section C allows specifying an estimated time the job would take to complete.

Afterward, the CaaS Service attempts to submit the job; the outcome is shown in the Job Monitoring page, Figure 7.16. Section A tells the client whether the job is submitted successfully. Section B offers commands to allow the client to take an appropriate action.

Section A: Identification

Job Name:

Job Owner:

Section B: Job File Specification

Executable:

Script:

Data files:

Proven.dat
Control.dat
Recent.dat

Output Filename:

Section C: Execution Specification

Estimated Tme:

FIGURE . Web page for job specification.

Section A: Submission Outcome

Outcome:

Job ID:

Report:

Delegating Submission request.... Request Accepted.
 Job has been started.

Section B: Job Control

Collect Results ->

FIGURE :Web page for monitoring job execution.

CLOUD STORAGE: FROM LANs TO WANs

Cloud computing has been viewed as the future of the IT industry. It will be a revolutionary change in computing services. Users will be allowed to purchase CPU cycles, memory utilities, and information storage services conveniently just like how we pay our monthly water and electricity bills. However, this image will not become realistic until some challenges have been addressed. In this section, we will briefly introduce the major difference brought by distributed data storage in cloud computing environment. Then, vulnerabilities in today's cloud computing platforms are analyzed and illustrated.

.1.Moving From LANs to WANs

Most designs of distributed storage take the form of either storage area networks (SANs) or network-attached storage (NAS) on the LAN level, such

as the networks of an enterprise, a campus, or an organization. SANs are constructed on top of block-addressed storage units connected through dedicated high-speed networks. In contrast, NAS is implemented by attaching specialized file servers to a TCP/IP network and providing a file-based interface to client machine [6]. For SANs and NAS, the distributed storage nodes are managed by the same authority. The system administrator has control over each node, and essentially the security level of data is under control. The reliability of such systems is often achieved by redundancy, and the storage security is highly dependent on the security of the system against the attacks and intrusion from outsiders. The confidentiality and integrity of data are mostly achieved using robust cryptographic schemes.

However, such a security system would not be robust enough to secure the data in distributed storage applications at the level of wide area net-works, specifically in the cloud computing environment. The recent progress of network technology enables global-scale collaboration over heterogeneous networks under different authorities. For instance, in a peer-to-peer (P2P) file sharing environment, or the distributed storage in a cloud computing environment, the specific data storage strategy is transparent to the user [3]. Furthermore, there is no approach to guarantee that the data host nodes are under robust security protection. In addition, the activity of the medium owner is not controllable to the data owner. Theoretically speaking, an attacker can do whatever she wants to the data stored in a storage node once the node is compromised. Therefore, the confidentiality and the integrity of the data would be violated when an adversary controls a node or the node administrator becomes malicious.

.2 Existing Commercial Cloud Services

As shown in Figure 8.1, data storage services on the platform of cloud computing are fundamentally provided by applications/software based on the Internet. Although the definition of cloud computing is not clear yet, several pioneer commercial implementations have been constructed and opened to the public, such as Amazon's Computer Cloud AWS (Amazon Web service) [7], the Microsoft Azure Service Platform [8], and the Google App Engine (GAE) [9].

In normal network-based applications, user authentication, data confidentiality, and data integrity can be solved through IPSec proxy using encryption and digital signature. The key exchanging issues can be solved by SSL proxy. These methods have been applied to today's cloud computing to secure the data on the cloud and also secure the communication of data to and from the cloud. The service providers claim that their services are secure. This section describes three secure methods used in three commercial cloud services and discusses their vulnerabilities.

Amazon's Web Service. Amazon provides Infrastructure as a Service (IaaS) with different terms, such as Elastic Compute Cloud (EC2), SimpleDB, Simple

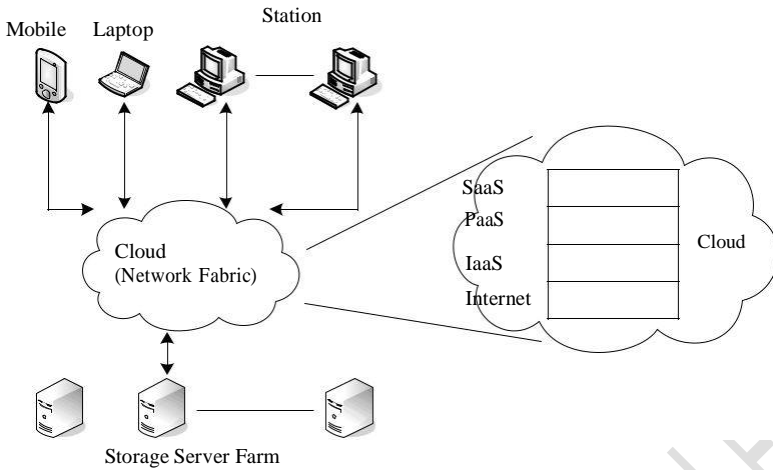


FIGURE : Illustration of cloud computing principle.

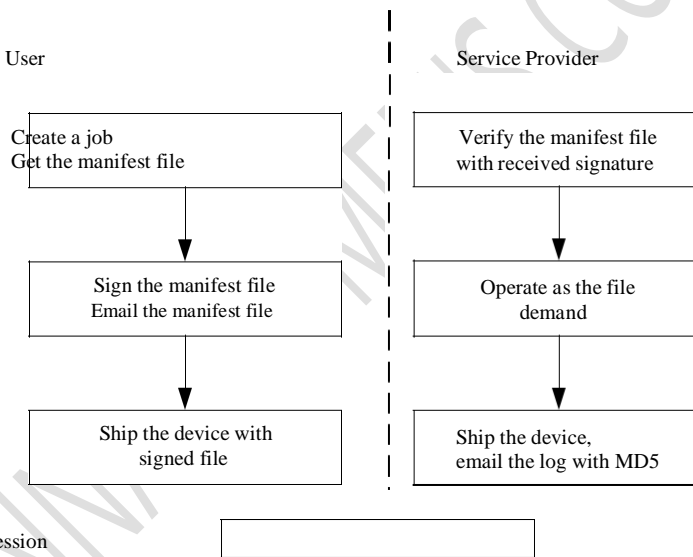


FIGURE . AWS data processing procedure.

Storage Service (S3), and so on. They are supposed to ensure the confidentiality, integrity, and availability of the customers' applications and data. Figure 8.2 presents one of the data processing methods adopted in Amazon's AWS [7], which is used to transfer large amounts of data between the AWS cloud and portable storage devices.

When the user wants to upload the data, he/she stores some parameters such as AccessKeyID, DeviceID, Destination, and so on, into an import metadata file called the manifest file and then signs the manifest file and e-mails the signed manifest file to Amazon. Another metadata file named the signature file is used by AWS to describe the cipher algorithm that is adopted to encrypt the job ID and the bytes in the manifest file. The signature file can uniquely identify and authenticate the user request.

The signature file is attached with the storage device, which is shipped to Amazon for efficiency. On receiving the storage device and the signature file, the service provider will validate the signature in the device with the manifest file sent through the email. Then, Amazon will e-mail management information back to the user including the number of bytes saved, the MD5 of the bytes, the status of the load, and the location on the Amazon S3 of the AWS Import Export Log. This log contains details about the data files that have been uploaded, including the key names, number of bytes, and MD5 checksum values.

The downloading process is similar to the uploading process. The user creates a manifest and signature file, e-mails the manifest file, and ships the storage device attached with signature file. When Amazon receives these two files, it will validate the two files, copy the data into the storage device, ship it back, and e-mail to the user with the status including the MD5 checksum of the data. Amazon claims that the maximum security is obtained via SSL endpoints.

Microsoft Windows Azure. The Windows Azure Platform (Azure) is an Internet-scale cloud services platform hosted in Microsoft data centers, which provides an operating system and a set of developer services that can be used individually or together [8]. The platform also provides scalable storage service. There are three basic data items: blobs (up to 50 GB), tables, and queues (,8k). In the Azure Storage, based on the blob, table, and queue structures, Microsoft promises to achieve confidentiality of the users' data. The procedure shown in Figure 8.3 provides security for data accessing to ensure that the data will not be lost.

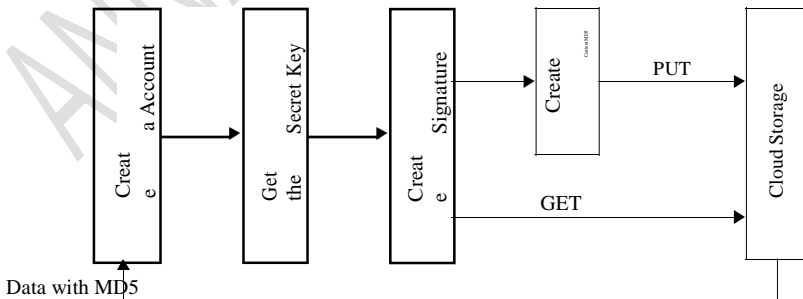


FIGURE Security data access procedure.

```

PUT http://jerry.blob.core.windows.net/movie/mov.avi
?comp=block &blockid=BlockId1 &timeout=30
HTTP/1.1 Content-Length: 2174344
Content-MD5: FJXZLUNMuI/KZ5KDcJPcOA==
Authorization:SharedKeyjerry:F5a+dUDvef+PfMb4T8Rc2jHcwfK58KecSZY+l2naIao=
x-ms-date: Sun, 13 Sept 2009 22:30:25 GMT
x-ms-version: 2009-04-14

GET http://jerry.blob.core.windows.net/movies/mov.avi
HTTP/1.1
Authorization:SharedKeyjerry:ZF3IJMtkOMi4y/nedSk5Vn74IU6/fRMwiPsL+uYSDjY=
x-ms-date: Sun, 13 Sept 2009 22:40:34 GMT
x-ms-version: 2009-04-14

```

FIGURE Example of a REST request.

To use Windows Azure Storage service, a user needs to create a storage account, which can be obtained from the Windows Azure portal web interface. After creating an account, the user will receive a 256-bit secret key. Each time when the user wants to send the data to or fetch the data from the cloud, the user has to use his secret key to create a HMAC SHA256 signature for each individual request for identification. Then the user uses his signature to authenticate request at server. The signature is passed with each request to authenticate the user requests by verifying the HMAC signature.

The example in Figure 8.4 is a REST request for a PUT/GET block operation [10]. Content-MD5 checksums can be provided to guard against network transfer errors and data integrity. The Content-MD5 checksum in the PUT is the MD5 checksum of the data block in the request. The MD5 checksum is checked on the server. If it does not match, an error is returned. The content length specifies the size of the data block contents. There is also an authorization header inside the HTTP request header as shown above in Figure 8.4.

At the same time, if the Content-MD5 request header was set when the blob has been uploaded, it will be returned in the response header. Therefore, the user can check for message content integrity. Additionally, the secure HTTP connection is used for true data integrity [7].

Google App Engine (GAE). The Google App Engine (GAE) [9] provides a powerful distributed data storage service that features a query engine and transactions. An independent third-party auditor, who claims that GAE can be secure under the SAS70 auditing industry standard, issued Google Apps an unqualified SAS70 Type II certification. However, from its on-line storage

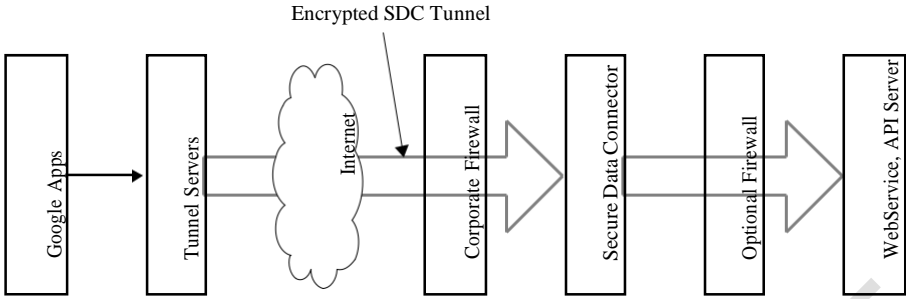


FIGURE Illustration of Google SDC working flow.

technical document of lower API [9], there are only some functions such as GET and PUT. There is no content addressing the issues of securing storage services. The security of data storage is assumed guaranteed using techniques such as by SSL link, based on our knowledge of security method adopted by other services.

Figure 8.5 is one of the secure services, called Google Secure Data Connector (SDC), based on GAE [9]. The SDC constructs an encrypted connection between the data source and Google Apps. As long as the data source is in the Google Apps domain to the Google tunnel protocol servers, when the user wants to get the data, he/she will first send an authorized data requests to Google Apps, which forwards the request to the tunnel server. The tunnel servers validate the request identity. If the identity is valid, the tunnel protocol allows the SDC to set up a connection, authenticate, and encrypt the data that flows across the Internet. At the same time, the SDC uses resource rules to validate whether a user is authorized to access a specified resource. When the request is valid, the SDC performs a network request. The server validates the signed request, checks the credentials, and returns the data if the user is authorized.

The SDC and tunnel server are like the proxy to encrypt connectivity between Google Apps and the internal network. Moreover, for more security, the SDC uses signed requests to add authentication information to requests that are made through the SDC. In the signed request, the user has to submit identification information including the `owner_id`, `viewer_id`, `instance_id`, `app_id`, `public_key`, `consumer_key`, `nonce`, `token`, and `signature` within the request [9] to ensure the integrity, security, and privacy of the request.

3 Vulnerabilities in Current Cloud Services

Previous subsections describe three different commercial cloud computing secure data storage schemes. Storage services that accept

a large amount of data (.1 TB) normally adopt strategies that help make the shipment more convenient, just as the Amazon AWS does. In contrast, services that only accept a smaller data amount (#50 GB) allow the data to be uploaded or downloaded via the Internet, just as the Azure Storage Service does. To provide data integrity, the Azure Storage Service stores the uploaded data MD5 checksum in the database and returns it to the user when the user wants to retrieve the data. Amazon AWS computes the data MD5 checksum and e-mails it to the user for integrity checking. The SDC is based on GAE's attempt to strengthen Internet authentication using a signed request. If these services are grouped together, the following scheme can be derived.

As shown in Figure 8.6, when user_1 stores data in the cloud, she can ship or send the data to the service provider with MD5_1. If the data are transferred through the Internet, a signed request could be used to ensure the privacy, security, and integrity of the data. When the service provider receives the data and the MD5 checksum, it stores the data with the corresponding checksum (MD5_1). When the service provider gets a verified request to retrieve the data from another user or the original user, it will send/ship the data with a MD5 checksum to the user. On the Azure platform, the original checksum MD5_1 will be sent, in contrast, a re-computed checksum MD5_2 is sent on Amazon's AWS.

The procedure is secure for each individual session. The integrity of the data during the transmission can be guaranteed by the SSL protocol applied. However, from the perspective of cloud storage

services, data integrity depends on the security of operations while in storage in addition to the security of the uploading and downloading sessions. The uploading session can only ensure that the data received by the cloud storage is the data that the user uploaded; the downloading session can guarantee the data that the user retrieved is the data cloud storage recorded. Unfortunately, this procedure applied on cloud storage services cannot guarantee data integrity.

To illustrate this, let's consider the following two scenarios. First, assume that Alice, a company CFO, stores the company financial data at a cloud storage service provided by Eve. And then Bob, the company administration chairman, downloads the data from the cloud. There are three important concerns in this simple procedure:

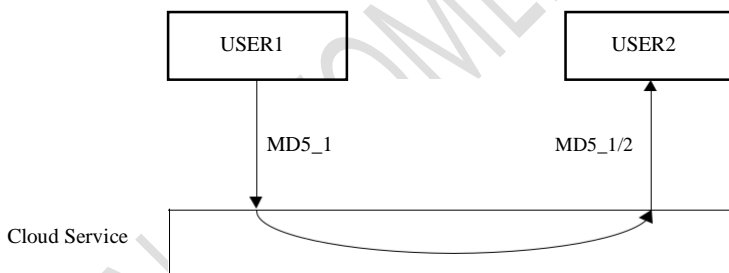


FIGURE . Illustration of potential integrity problem.

1. Confidentiality. Eve is considered as an untrustworthy third party, Alice and Bob do not want reveal the data to Eve.
2. Integrity. As the administrator of the storage service, Eve has the capability to play with the data in hand. How can Bob be confident that the data he fetched from Eve are the same as what was sent by Alice? Are there any measures to guarantee that the data have not been tampered by Eve?
3. Repudiation. If Bob finds that the data have been tampered with, is there any evidence for him to demonstrate that it is Eve who should be responsible for the fault? Similarly, Eve also needs certain evidence to prove her innocence.

Recently, a potential customer asked a question on a cloud mailing-group regarding data integrity and service reliability. The reply from the developer was “We won’t lose your data—we have a robust backup and recovery strategy — but we’re not responsible for you losing your own data . . .” [11]. Obviously, it is not persuasive to the potential customer to be confident with the service.

The repudiation issue opens a door for potentially blackmailers when the user is malicious. Let’s assume that Alice wants to blackmail Eve. Eve is a cloud storage service provider who claims that data integrity is one of their key features. For that purpose, Alice stored some data in the cloud, and later she downloaded the data. Then, she reported that her data were incorrect and that it is the fault of the storage provider. Alice claims compensation for her so-called loss. How can the service provider demonstrate her innocence?

Confidentiality can be achieved by adopting robust encryption schemes. However, the integrity and repudiation issues are not handled well on the current cloud service platform. One-way SSL session only guarantees one-way integrity. One critical link is missing between the uploading and downloading sessions: There is no mechanism for the user or service provider to check whether the record has been modified in the cloud storage. This vulnerability leads to the following questions:

Upload-to-Download Integrity. Since the integrity in uploading and down-loading phase are handled separately, how can the

user or provider know the data retrieved from the cloud is the same data that the user uploaded previously?

Repudiation Between Users and Service Providers. When data errors happen without transmission errors in the uploading and downloading sessions, how can the user and service provider prove their innocence?

4 Bridge the Missing Link

This section presents several simple ideas to bridge the missing link based on digital signatures and authentication coding schemes. According to whether there is a third authority certified (TAC) by the user and provider and whether the user and provider are using the secret key sharing technique (SKS), there are four solutions to bridge the missing link of data integrity between the uploading and downloading procedures. Actually, other digital signature technologies can be adopted to fix this vulnerability with different approaches.

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.

Once the uploading operation finished, both sides agreed on the integrity of the uploaded data, and each side owns the MD5 checksum and MD5 signature generated by the opposite site.

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.

When disputation happens, the user or the service provider can check the MD5 checksum and the signature of MD5 checksum generated by the opposite side to prove its innocence. However, there are some special cases that exist. When the service provider is trustworthy, only MSU is needed; when the user is trustworthy, only MSP is needed; if each of them trusts the other side, neither MSU nor MSP is needed. Actually, that is the current method adopted in cloud computing platforms. Essentially, this approach implies that when the identity is authenticated that trust is established.

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.

Then, both sides agree on the integrity of the uploaded data, and they share the agreed MD5 checksum, which is used when disputation happens.

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.

When disputation happens, the user or the service provider can take the shared MD5 together, recover it, and prove his/her innocence.

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.

On finishing the uploading phase, both sides agree on the integrity of the uploaded data, and TAC owns their agreed MD5 signature.

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.

When disputation happens, the user or the service provider can prove his innocence by presenting the MSU and MSP stored at the TAC.

Similarly, there are some special cases. When the service provider is trustworthy, only the MSU is needed; when the user is trustworthy, only the MSP is needed; if each of them trusts the other, the TAC is not needed. Again, the last case is the method adopted in the current cloud computing platforms. When the identity is authenticated, trust is established.

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.

Both sides agree on the integrity of the uploaded data and share the same MD5 checksum by SKS, and the TAC own their agreed MD5 signatures.

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.

When disputation happens, the user or the service provider can prove their innocence by checking the shared MD5 checksum together. If the disputation cannot be resolved, they can seek further help from the TAC for the MD5 checksum.

Here are the special cases. When the service provider is trustworthy, only the user needs the MD5 checksum; when the user is trustworthy, only the service provider needs MD5 checksum; if both of them can be trusted, the TAC is not needed. This is the method used in the current cloud computing platform.

TECHNOLOGIES FOR DATA SECURITY IN CLOUD COMPUTING

This section presents several technologies for data security and privacy in cloud computing. Focusing on the unique issues of the cloud data storage platform, this section does not repeat the normal approaches that provide confidentiality, integrity, and availability in distributed data storage applications. Instead, we select to illustrate the unique requirements for cloud computing data security from a few different perspectives:

Database Outsourcing and Query Integrity Assurance. Researchers have pointed out that storing data into and fetching data from devices and machines behind a cloud are essentially a novel form of database outsourcing. Section 8.3.1 introduces the technologies of Database Out-sourcing and Query Integrity Assurance on the clouding computing platform.

Data Integrity in Untrustworthy Storage. One of the main challenges that prevent end users from adopting cloud storage services is the fear of losing data or data corruption. It is critical to relieve the users' fear by providing technologies that enable users to check the integrity of their data. Section 8.3.2 presents two approaches that allow users to detect whether the data has been touched by unauthorized people.

Web-Application-Based Security. Once the dataset is stored remotely, a Web browser is one of the most convenient approaches that end users can use to access their data on remote services. In the era of cloud computing, Web security plays a more important role than ever. Section 8.3.3 discusses the most important concerns in Web security and analyzes a couple of widely used attacks.

Multimedia Data Security. With the development of high-speed network technologies and large bandwidth connections, more and more multi-media data are being stored and shared in cyber space. The security requirements for video, audio, pictures, or images are different from other applications. Section 8.3.4 introduces the requirements for multimedia data security in the cloud.

1. Database Outsourcing and Query Integrity Assurance

In recent years, database outsourcing has become an important component of cloud computing. Due to the rapid advancements in network technology, the cost of transmitting a terabyte of data over long distances has decreased significantly in the past decade. In addition, the total cost of data management is five to ten times higher than the initial acquisition costs.

As a result, there is a growing interest in outsourcing database management tasks to third parties that can provide these tasks for a much lower cost due to the economy of scale.

This new outsourcing model has the benefits of reducing the costs for running Database Management Systems (DBMS) independently and enabling enter-prises to concentrate on their main businesses [12]. Figure 8.7 demonstrates the general architecture of a database outsourcing environment with clients.

The database owner outsources its data management tasks, and clients send queries to the untrusted service provider.

Let T denote the data to be outsourced. The data T are is preprocessed, encrypted, and stored at the service provider. For evaluating queries, a user rewrites a set of queries Q against T to queries against the encrypted database.

The outsourcing of databases to a third-party service provider was first introduced by Hacigu`mu`s et al. [13]. Generally, there are two security concerns

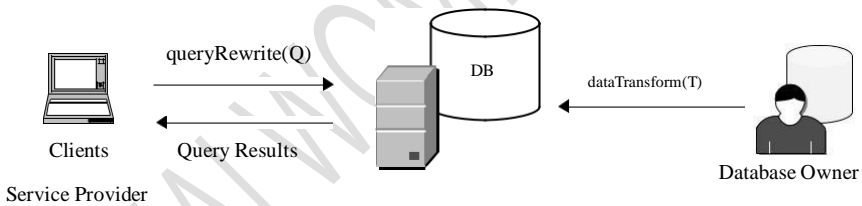


FIGURE . The system architecture of database outsourcing.

in database outsourcing. These are data privacy and query integrity. The related research is outlined below.

Data Privacy Protection. Hacigu"mu"s et al. [37] proposed a method to execute SQL queries over encrypted databases. Their strategy is to process as much of a query as possible by the service providers, without having to decrypt the data. Decryption and the remainder of the query processing are performed at the client side. Agrawal et al. [14] proposed an order-preserving encryption scheme for numeric values that allows any comparison operation to be directly applied on encrypted data. Their technique is able to handle updates, and new values can be added without requiring changes in the encryption of other values. Generally, existing methods enable direct execution of encrypted queries on encrypted datasets and allow users to ask identity queries over data of different encryptions. The ultimate goal of this research direction is to make queries in encrypted databases as efficient as possible while preventing adversaries from learning any useful knowledge about the data. However, researches in this field did not consider the problem of query integrity.

Query Integrity Assurance. In addition to data privacy, an important security concern in the database outsourcing paradigm is query integrity. Query integrity examines the trustworthiness of the hosting environment. When a client receives a query result from the service provider, it wants to be assured that the result is both correct and complete, where correct means that the result must originate in the owner's data and not has been tampered with, and complete means that the result includes all records satisfying the query. Devanbu et al. [15] authenticate data records using the Merkle hash tree [16], which is based on the idea of using a signature on the root of the Merkle hash tree to generate a proof of correctness. Mykletun et al. [17] studied and compared several signature methods that can be utilized in data authentication, and they identified the problem of completeness but did not provide a solution. Pang et al. [18] utilized an aggregated signature to sign each record with the information from neighboring records by assuming that all the records are sorted with a certain order.

The method ensures the completeness of a selection query by checking the aggregated signature. But it has difficulties in handling multipoint selection query of which the result tuples occupy a noncontinuous region of the ordered sequence.

The work in Li et al. [19] utilizes Merkle hash tree-based methods to audit the completeness of query results, but since the Merkle hash tree also applies the signature of the root Merkle tree node, a similar difficulty exists. Besides, the network and CPU overhead on the client side can be prohibitively high for some types of queries. In some extreme cases, the overhead could be as high as processing these queries locally, which can undermine the benefits of database outsourcing. Sion [20] proposed a mechanism called the challenge token and uses it as a probabilistic proof that the server has executed the query over the entire database. It can handle arbitrary types of queries including joins and does not assume that the underlying data is ordered. However, the approach is not applied to the adversary model where an adversary can first compute the complete query result and then delete the tuples specifically corresponding to the challenge tokens [21]. Besides, all the aforementioned methods must modify the DBMS kernel in order to provide proof of integrity.

Recently, Wang et al. [22] proposed a solution named dual encryption to ensure query integrity without requiring the database engine to perform any special function beyond query processing. Dual encryption enables cross-examination of the outsourced data, which consist of (a) the original data stored under a certain encryption scheme and (b) another small percentage of the original data stored under a different encryption scheme. Users generate queries against the additional piece of data and analyze their results to obtain integrity assurance.

For auditing spatial queries, Yang et al [23] proposed the MR-tree, which is an authenticated data structure suitable for verifying queries executed on outsourced spatial databases. The authors also designed a caching technique to reduce the information sent to the client for verification purposes. Four spatial transformation mechanisms are presented in Yiu et al. [24] for protect-ing the privacy of outsourced private spatial data.

The data owner selects transformation keys that are shared with trusted clients, and it is infeasible to reconstruct the exact original data points from the transformed points without the key. However, both aforementioned researches did not consider data privacy protection and query integrity auditing jointly in their design.

The state-of-the-art technique that can ensure both privacy and integrity for outsourced spatial data is proposed in Ku et al. [12]. In particular, the solution first employs a one-way spatial transformation method based on Hilbert curves, which encrypts the spatial data before outsourcing and hence ensures its privacy. Next, by probabilistically replicating a portion of the data and encrypting it with a different encryption key, the authors devise a mechanism for the client to audit the trustworthiness of the query results.

2 Data Integrity in Untrustworthy Storage

While the transparent cloud provides flexible utility of network-based resources, the fear of loss of control on their data is one of the major concerns that prevent end users from migrating to cloud storage services. Actually it is a potential risk that the storage infrastructure providers become self-interested, untrustworthy, or even malicious. There are different motivations whereby a storage service provider could become untrustworthy—for instance, to cover the consequence of a mistake in operation, or deny the vulnerability in the system after the data have been stolen by an adversary. This section introduces two technologies to enable data owners to verify the data integrity while the files are stored in the remote untrustworthy storage services.

Actually, before the term “cloud computing” appears as an IT term, there are several remote data storage checking protocols that have been suggested [25], [26]. Later research has summarized that in practice a remote data possession checking protocol has to satisfy the following five requirements [27].

Note that the verifier could be either the data owner or a trusted third party, and the prover could be the storage service provider or storage medium owner or system administrator.

Requirement #1. It should not be a pre-requirement that the verifier has to possess a complete copy of the data to be checked. And in practice, it does not make sense for a verifier to keep a duplicated copy of the content to be verified. As long as it serves the purpose well, storing a more concise contents digest of the data at the verifier should be enough.

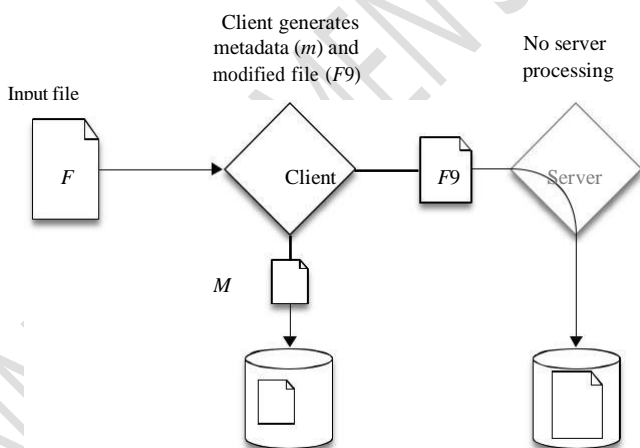
Requirement #2. The protocol has to be very robust considering the untrustworthy prover. A malicious prover is motivated to hide the violation of data integrity. The protocol should be robust enough that such a prover ought to fail in convincing the verifier.

Requirement #3. The amount of information exchanged during the verification operation should not lead to high communication overhead. Requirement #4. The protocol should be computationally efficient.

Requirement #5. It ought to be possible to run the verification an unlimited number of times.

A PDP-Based Integrity Checking Protocol. Ateniese et al. [28] proposed a protocol based on the provable data possession (PDP) technology, which allows users to obtain a probabilistic proof from the storage service providers. Such a proof will be used as evidence that their data have been stored there. One of the advantages of this protocol is that the proof could be generated by the storage service provider by accessing only a small portion of the whole dataset

Figure presents the flowcharts of the protocol for provable data possession [28]. The data owner, the client in the figure, executes the protocol to verify that a dataset is stored in an outsourced storage machine as a collection of n blocks. Before uploading the data into the remote storage, the data owner pre-processes the dataset and a piece of metadata is generated.



“challenge” and send it to the service provider to ensure that the storage server has stored the dataset. The data owner requests that the storage server generate a metadata based on the stored data and then send it back. Using the previously stored local metadata, the owner verifies the response.

On the behalf of the cloud service provider’s side, the server may receive multiple challenges from different users at the same time. For the sake of availability, it is highly desired to minimize not only the computational overhead of each individual calculation, but also the number of data blocks to be accessed. In addition, considering the pressure on the communication networks, minimal bandwidth consumption also implies that there are a limited amount of metadata included in the response generated by the server. In the protocol shown in Figure 8.8, the PDP scheme only randomly accesses one subdata block when the sample the stored dataset [28]. Hence, the PDP scheme probabilistically guarantees the data integrity. It is mandatory to access the whole dataset if a deterministic guarantee is required by the user.

An Enhanced Data Possession Checking Protocol. Sebe et al. [27] pointed out that the above PDP-based protocol does not satisfy Requirement #2 with 100% probability. An enhanced protocol has been proposed based on the idea of the Diffie Hellman scheme. It is claimed that this protocol satisfies all five requirements and is computationally more efficient than the PDP-based protocol [27]. The verification time has been shortened at the setup stage by taking advantage of the trade-offs between the computation times required by the prover and the storage required at the verifier. The setup stage sets the following parameters:

p and q : two primary factors chosen by the verifier;

$N = pq$: a public RSA modulus created by the verifier;

$\Phi(N) = (p - 1)(q - 1)$: the private key of the verifier, which is the secret only known by the verifier;

l : an integer that is chosen depending on the trade-offs between the computation time required at the prover and the storage required at the verifier;

t : a security parameter;

PRNG: a pseudorandom number generator, which generates t -bit integer values.

The protocol is presented as follows:

At first, the verifier generates the digest of data m :

1. Break the data m into n pieces, each is l -bit. Let m_1, m_2, \dots, m_n ($n \cdot l = |m|$) be the integer values corresponding to fragments of m .
2. For each fragment m_i , compute and store $M_i = m_i \bmod \Phi(N)$.

The challenge response verification protocol is as follows:

1. The verifier
 - generates a random seed S and a random element $\alpha \in \mathbb{Z}_N$
 - 1.1 $\alpha \in \{1, N-1\}$
 - And
 - 1.2 sends the challenge (α, S) to the prover.
2. Upon receiving the challenge, the prover:
 - 2.1 generates n pseudorandom values $c_i \in [1, 2^l]$, for $i = 1$ to n , using PRNG seeded by S ,
 - 2.2 calculates $r = \prod_{i=1}^n c_i m_i$ and $R = \alpha^r \bmod N$, and
 - 2.3 sends R to the verifier.
3. The verifier:
 - 3.1 regenerates the n pseudorandom values $c_i \in [1, 2^l]$, for $i = 1$ to n , using PRNG seeded by S ,
 - 3.2 calculates $r' = \prod_{i=1}^n c_i m_i \bmod \Phi(N)$ and $R' = \alpha^{r'} \bmod N$, and
 - 3.3 checks whether $R = R'$.

Due to the space constraints, this section only introduces the basic principles and the working flows of the protocols for data integrity checking in untrustworthy storages. The proof of the correctness, security analysis, and the performance analysis of the protocols are left for the interested readers to explore deeper in the cited research papers [25, 26-28].

3 Web-Application-Based Security

In cloud computing environments, resources are provided as a service over the Internet in a dynamic, virtualized, and scalable way [29, 30]. Through cloud computing services, users access business applications on-line from a Web browser, while the software and data are stored on the servers. Therefore, in the era of cloud computing, Web security plays a more important role than ever. The Web site server is the first gate that guards the vast cloud resources. Since the cloud may operate continuously to process millions of dollars' worth of daily on-line transactions, the impact of any Web security vulnerability will be amplified at the level of the whole cloud.

Web attack techniques are often referred as the class of attack. When any Web security vulnerability is identified, attacker will employ those techniques to take advantage of the security vulnerability. The types of attack can be categorized in Authentication, Authorization, Client-Side Attacks, Comm-and Execution, Information Disclosure, and Logical Attacks [31]. Due to the limited space, this section introduces each of them briefly. Interested read-ers are encouraged to explore for more detailed information from the materials cited.

Authentication. Authentication is the process of verifying a claim that a subject made to act on behalf of a given principal. Authentication attacks target a Web site's method of validating the identity of a user, service, or application, including Brute Force, Insufficient Authentication, and Weak Password Recovery Validation. Brute Force attack employs an automated process to guess a person's username and password by trial and error. In the Insufficient Authentication case, some sensitive content or functionality are protected by

“hiding” the specific location in obscure string but still remains accessible directly through a specific URL. The attacker could discover those URLs through a Brute Force probing of files and directories. Many Web sites provide password recovery service. This service will automatically recover the user name or password to the user if she or he can answer some questions defined as part of the user registration process. If the recovery questions are either easily guessed or can be skipped, this Web site is considered to be Weak Password Recovery Validation.

Authorization. Authorization is used to verify if an authenticated subject can perform a certain operation. Authentication must precede authorization. For example, only certain users are allowed to access specific content or functionality.

Authorization attacks use various techniques to gain access to protected areas beyond their privileges. One typical authorization attack is caused by Insufficient Authorization. When a user is authenticated to a Web site, it does not necessarily mean that she should have access to certain content that has been granted arbitrarily. Insufficient authorization occurs when a Web site does not protect sensitive content or functionality with proper access control restrictions. Other authorization attacks are involved with session. Those attacks include Credential/Session Prediction, Insufficient Session Expiration, and Session Fixation.

In many Web sites, after a user successfully authenticates with the Web site for the first time, the Web site creates a session and generate a unique “session ID” to identify this session. This session ID is attached to subsequent requests to the Web site as “Proof” of the authenticated session.

Credential/Session Prediction attack deduces or guesses the unique value of a session to hijack or impersonate a user.

Insufficient Session Expiration occurs when an attacker is allowed to reuse old session credentials or session IDs for authorization. For example, in a shared computer, after a user accesses a Web site and then leaves, with Insufficient Session Expiration, an attacker can use the browser’s back button to access Web pages previously accessed by the victim.

Session Fixation forces a user’s session ID to an arbitrary value via Cross-Site Scripting or peppering the Web site with previously

made HTTP requests. Once the victim logs in, the attacker uses the predefined session ID value to impersonate the victim's identity.

Client-Side Attacks. The Client-Side Attacks lure victims to click a link in a malicious Web page and then leverage the trust relationship expectations of the victim for the real Web site. In Content Spoofing, the malicious Web page can trick a user into typing user name and password and will then use this information to impersonate the user.

Cross-Site Scripting (XSS) launches attacker-supplied executable code in the victim's browser. The code is usually written in browser-supported scripting languages such as JavaScript, VBScript, ActiveX, Java, or Flash. Since the code will run within the security context of the hosting Web site, the code has the ability to read, modify, and transmit any sensitive data, such as cookies, accessible by the browser.

Cross-Site Request Forgery (CSRF) is a server security attack to a vulnerable site that does not take the checking of CSRF for the HTTP/HTTPS request. Assuming that the attacker knows the URLs of the vulnerable site which are not protected by CSRF checking and the victim's browser stores credentials such as cookies of the vulnerable site, after luring the victim to click a link in a malicious Web page, the attacker can forge the victim's identity and access the vulnerable Web site on victim's behalf.

Command Execution. The Command Execution attacks exploit server-side vulnerabilities to execute remote commands on the Web site. Usually, users supply inputs to the Web-site to request services. If a Web application does not properly sanitize user-supplied input before using it within application code, an attacker could alter command execution on the server. For example, if the length of input is not checked before use, buffer overflow could happen and result in denial of service. Or if the Web application uses user input to construct statements such as SQL, XPath, C/C11 Format String, OS system command, LDAP, or dynamic HTML, an attacker may inject arbitrary executable code into the server if the user input is not properly filtered.

Information Disclosure. The Information Disclosure attacks acquire sensitive information about a web site revealed by developer comments, error messages, or well-know file name conventions. For example, a Web server may return a list of files within a requested

directory if the default file is not present. This will supply an attacker with necessary information to launch further attacks against the system. Other types of Information Disclosure includes using special paths such as “.” and “..” for Path Traversal, or uncovering hidden URLs via Predictable Resource Location.

Logical Attacks. Logical Attacks involve the exploitation of a Web application’s logic flow. Usually, a user’s action is completed in a multi-step process. The procedural workflow of the process is called application logic. A common Logical Attack is Denial of Service (DoS). DoS attacks will attempt to consume all available resources in the Web server such as CPU, memory, disk space, and so on, by abusing the functionality provided by the Web site. When any one of any system resource reaches some utilization threshold, the Web site will no longer be responsive to normal users. DoS attacks are often caused by Insufficient Anti-automation where an attacker is permitted to automate a process repeatedly. An automated script could be executed thousands of times a minute, causing potential loss of performance or service.

4. Multimedia Data Security Storage

With the rapid developments of multimedia technologies, more and more multimedia contents are being stored and delivered over many kinds of devices, databases, and networks. Multimedia Data Security plays an important role in the data storage to protect multimedia data. Recently, how storage multimedia contents are delivered by both different providers and users has attracted much attention and many applications. This section briefly goes through the most critical topics in this area.

Protection from Unauthorized Replication. Contents replication is required to generate and keep multiple copies of certain multimedia contents. For example, content distribution networks (CDNs) have been used to manage content distribution to large numbers of users, by keeping the replicas of the same contents on a group of geographically distributed surrogates [32, 33]. Although the replication can improve the system performance, the unauthorized replication causes some problems such as contents copyright, waste of replication cost, and extra control overheads.

Protection from Unauthorized Replacement. As the storage capacity is limited, a replacement process must be carried out when the capacity exceeds its limit. It means the situation that a currently stored content [34] must be removed from the storage space in order to make space for the new coming content. However, how to decide which content should be removed is very important. If an unauthorized replacement happens, the content which the user doesn't want to delete will be removed resulting in an accident of the data loss. Furthermore, if the important content such as system data is removed by unauthorized replacement, the result will be more serious.

Protection from Unauthorized Pre-fetching. The Pre-fetching is widely deployed in Multimedia Storage Network Systems between server databases and end users' storage disks [35]. That is to say, If a content can be predicted to be requested by the user in future requests, this content will be fetched from the server database to the end user before this user requests it, in order to decrease user response time. Although the Pre-fetching shows its efficiency, the un-authorized pre-fetching should be avoided to make the system to fetch the necessary content.

UNIT IV

WORKFLOW MANAGEMENT SYSTEMS AND CLOUDS

The primary benefit of moving to clouds is application scalability. Unlike grids, scalability of cloud resources allows real-time provisioning of resources to meet application requirements at runtime or prior to execution. The elastic nature of clouds facilitates changing of resource quantities and characteristics to vary at runtime, thus dynamically scaling up when there is a greater need for additional resources and scaling down when the demand is low.

This enables workflow management systems to readily meet quality-of-service (QoS) requirements of applications, as opposed to the traditional approach that required advance reservation of resources in global multi-user grid environments. With most cloud computing services coming from large commercial organizations, service-level agreements (SLAs) have been an important concern to both the service providers and consumers.

Due to competitions within emerging service providers, greater care is being taken in designing SLAs that seek to offer (a) better QoS guarantees to customers and (b) clear terms for compensation in the event of violation. This allows workflow management systems to provide better end-to-end guarantees when meeting the service requirements of users by mapping them to service providers based on characteristics of SLAs. Economically motivated, commercial cloud providers strive to provide better services guarantees compared to grid service providers. Cloud providers also take advantage of economies of scale, providing

compute, storage, and bandwidth resources at substantially lower costs. Thus utilizing public cloud services could be economical and a cheaper alternative (or add-on) to the more expensive dedicated resources.

1. Architectural Overview

Figure 12.1 presents a high-level architectural view of a Workflow Management System (WfMS) utilizing cloud resources to drive the execution of a scientific

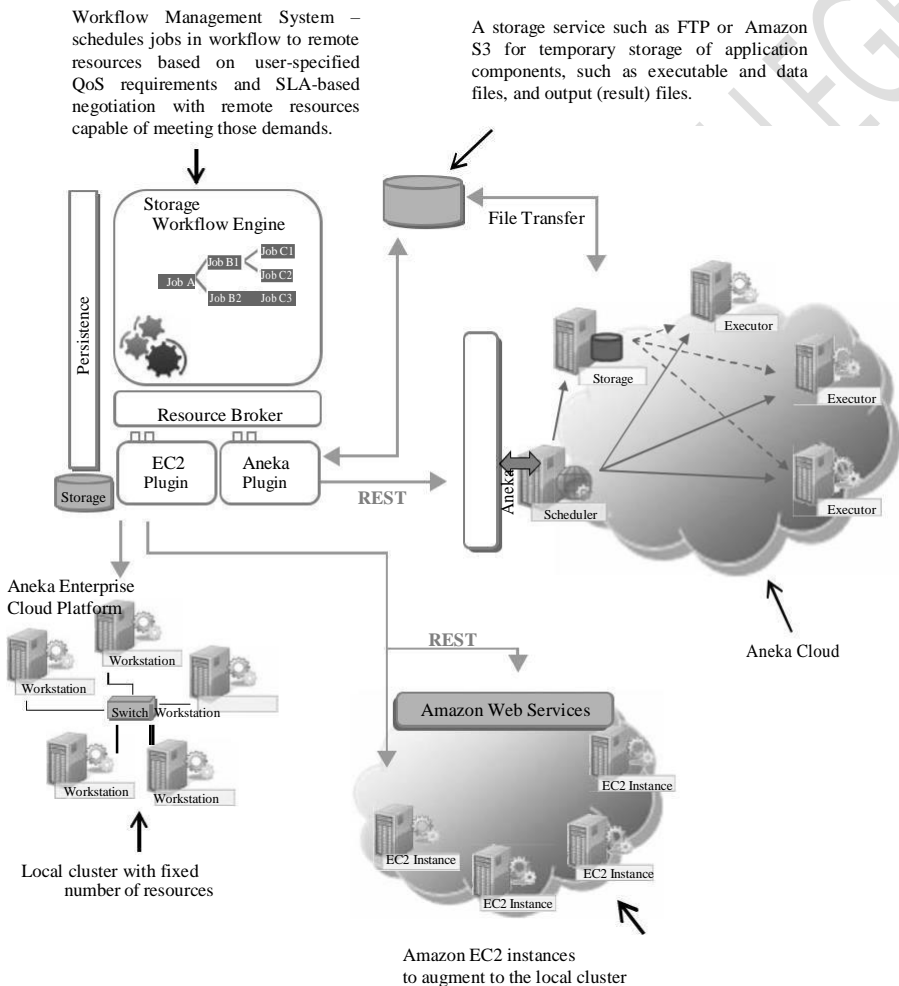


FIGURE : Workflow engine in the cloud.

workflow application. The workflow system comprises the workflow engine, a resource broker [13], and plug-ins for communicating with various technological platforms, such as Aneka [14] and Amazon EC2. A detailed architecture describing the components of a WfMS is given in Section 12.4.

User applications could only use cloud services or use cloud together with existing grid/cluster-based solutions. Figure 12.1 depicts two scenarios, one where the Aneka platform is used in its entirety to complete the workflow, and the other where Amazon EC2 is used to supplement a local cluster when there are insufficient resources to meet the QoS requirements of the application. Aneka [13], described in further detail in Section 12.5, is a PaaS cloud and can be run on a corporate network or a dedicated cluster or can be hosted entirely on an IaaS cloud. Given limited resources in local networks, Aneka is capable of transparently provisioning additional resources by acquiring new resources in third-party cloud services such as Amazon EC2 to meet application demands. This relieves the WfMS from the responsibility of managing and allocating resources directly, to simply negotiating the required resources with Aneka.

Aneka also provides a set of Web services for service negotiation, job submission, and job monitoring. The WfMS would orchestrate the workflow execution by scheduling jobs in the right sequence to the Aneka Web Services.

The typical flow of events when executing an application workflow on Aneka would begin with the WfMS staging in all required data for each job onto a remote storage resource, such as Amazon S3 or an FTP server. In this case, the data would take the form of a set of files, including the application binaries. These data can be uploaded by the user prior to execution, and they can be stored in storage facilities offered by cloud services for future use. The WfMS then forwards workflow tasks to Aneka's scheduler via the Web service interface. These tasks are subsequently examined for required files, and the storage service is instructed to stage them in from the remote storage server, so that they are accessible by the internal network of execution nodes. The execution begins by scheduling tasks to available execution nodes (also known as worker nodes). The workers download any required files for each

task they execute from the storage server, execute the application, and upload all output files as a result of the execution back to the storage server. These files are then staged out to the remote storage server so that they are accessible by other tasks in the workflow managed by the WfMS. This process continues until the workflow application is complete.

The second scenario describes a situation in which the WfMS has greater control over the compute resources and provisioning policies for executing workflow applications. Based on user-specified QoS requirements, the WfMS schedules workflow tasks to resources that are located at the local cluster and in the cloud. Typical parameters that drive the scheduling decisions in such a scenario include deadline (time) and budget (cost) [15, 16]. For instance, a policy for scheduling an application workflow at minimum execution cost would utilize local resources and then augment them with cheaper cloud resources, if needed, rather than using high-end but more expensive cloud resources. On the contrary, a policy that scheduled workflows to achieve minimum execution time would always use high-end cluster and cloud resources, irrespective of costs. The resource provisioning policy determines the extent of additional resources to be provisioned on the public clouds. In this second scenario, the WfMS interacts directly with the resources provisioned. When using Aneka, however, all interaction takes place via the Web service interface.

The following sections focuses on the integration of workflow management systems and clouds and describes in detail practical issues involved in using clouds for scientific workflow applications.

ARCHITECTURE OF WORKFLOW MANAGEMENT SYSTEMS

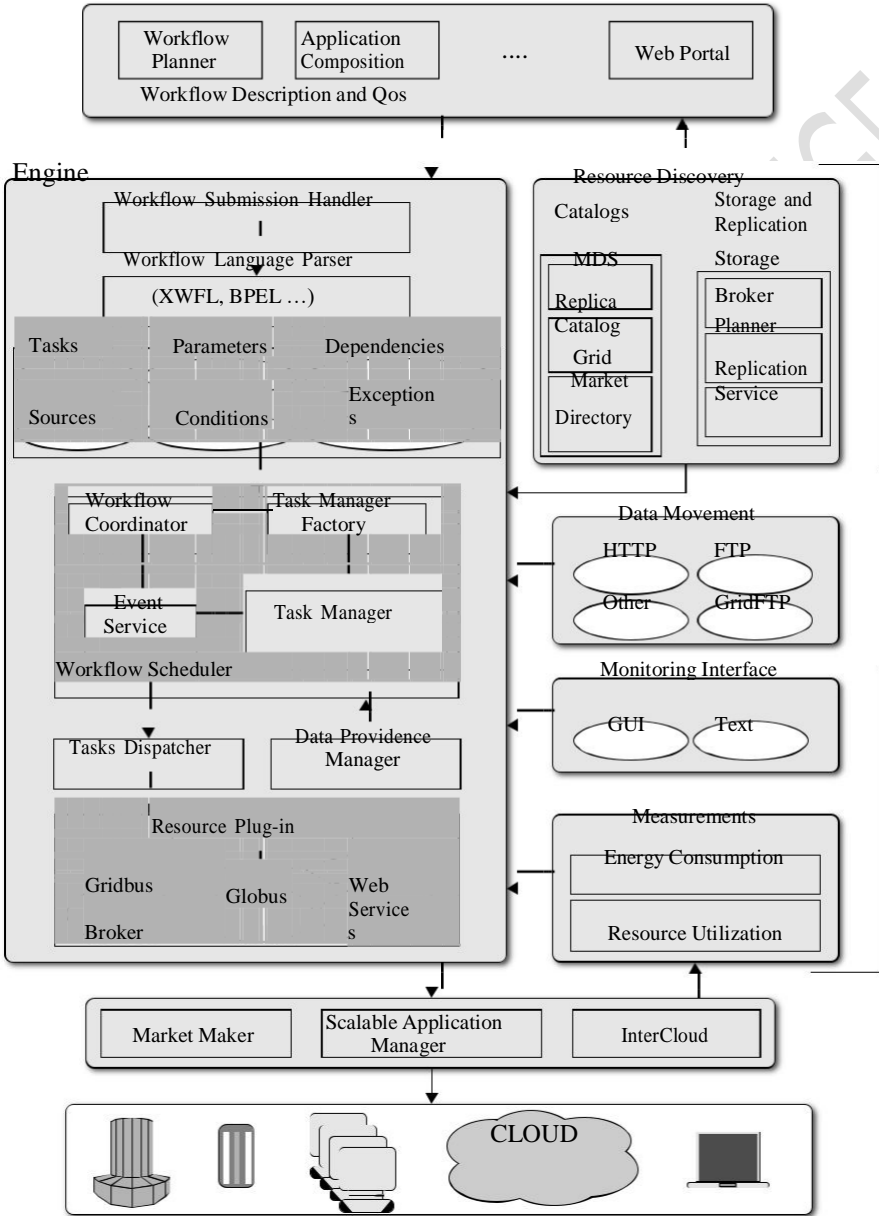
Scientific applications are typically modeled as workflows, consisting of tasks, data elements, control sequences and data dependencies. Workflow management systems are responsible for managing and executing these workflows. According to Raicu et al. [17], scientific workflow management systems are engaged and applied to the following aspects of scientific computations: (1) describing complex scientific procedures (using GUI tools, workflow specific languages), (2) automating data derivation processes (data transfer components), (3) high-performance computing (HPC) to improve throughput and performance (distributed resources and their coordination), and (4) provenance management and query (persistence components). The Cloudbus Workflow Management System [12] consists of components that are responsible for handling tasks, data and resources taking into account users' QoS requirements. Its architecture is depicted in Figure 12.2.

The architecture consists of three major parts: (a) the user interface, (b) the core, and (c) plug-ins. The user interface allows end users to work with workflow composition, workflow execution planning, submission, and monitoring. These features are delivered through a Web portal or through a stand-alone application that is installed at the user's end. Workflow composition is done using an XML-based Workflow Language (xWFL). Users define task properties and link them based on their data dependencies. Multiple tasks can be constructed using copy-paste functions present in most GUIs.

The components within the core are responsible for managing the execution of workflows. They facilitate in the translation of high-level workflow descriptions (defined at the user interface using XML) to task and data objects. These objects are then used by the execution subsystem. The scheduling component applies user-selected scheduling policies and plans to the workflows at various stages in their execution. The tasks and data dispatchers interact

with the resource interface plug-ins to continuously submit and monitor tasks in the workflow. These components form the core part of the workflow engine.

Workflow Management System



(e.g., querying metadata services, reading from trace files), transferring data to and from resources (e.g., transfer protocol implementations, and storage and replication services), monitoring the execution status of tasks and applications (e.g., real-time monitoring GUIs, logs of execution, and the scheduled retrieval of task status), and measuring energy consumption.

The resources are at the bottom layer of the architecture and include clusters, global grids, and clouds. The WfMS has plug-in components for interacting with various resource management systems present at the front end of distributed resources. Currently, the Cloudbus WfMS supports Aneka, Pbs, Globus, and fork-based middleware. The resource managers may communicate with the market maker, scalable application manager, and InterCloud services for global resource management [18].

UTILIZING CLOUDS FOR WORKFLOW EXECUTION

Taking the leap to utilizing cloud services for scientific workflow applications requires an understanding of the types of clouds services available, the required component changes in workflow systems for interacting with cloud services, the set of tools available to support development and deployment efforts, the steps involved in deploying workflow systems and services on the cloud, and an appreciation of the key benefits and challenges involved. In the sections to follow, we take a closer look at some of these issues. We begin by introducing the reader to the Aneka Enterprise Cloud service. We do this for two reasons. First, Aneka serves as a useful tool for utilizing clouds, including platform abstraction and dynamic provisioning. Second, we describe later in the chapter a case study detailing the use of Aneka to execute a scientific workflow application on clouds.

1. Aneka

Aneka is a distributed middleware for deploying platform-as-a-service (PaaS) offerings (Figure 12.3). Developed at CLOUDS Lab, University of Melbourne, Aneka is the result of years of research on cluster, grid, and cloud computing for high-performance computing

(HPC) applications. Aneka, which is both a development and runtime environment, is available for public use (for a cost),⁴ can be installed on corporate networks, or dedicated clusters, or can be hosted on infrastructure clouds like Amazon EC2. In comparison, similar PaaS services such as Google AppEngine [19] and Windows Azure [20] are in-house platforms hosted on infrastructures owned by the respective companies. Aneka was developed on Microsoft's .NET Framework 2.0 and is compatible with other implementations of the ECMA 335 standard [21], such as Mono. Aneka

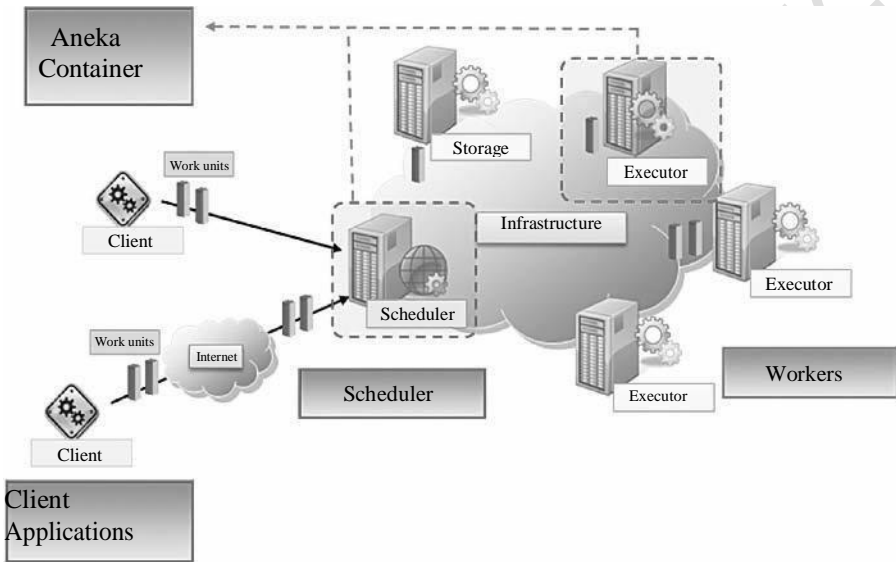


FIGURE . A deployment of Aneka Enterprise Cloud.

can run on popular platforms such as Microsoft Windows, Linux, and Mac OS X, harnessing the collective computing power of a heterogeneous network.

The runtime environment consists of a collection of Aneka containers running on physical or virtualized nodes. Each of these containers can be configured to play a specific role such as scheduling or execution. The Aneka distribution also provides a set of tools for administrating the cloud, reconfiguring nodes, managing users, and monitoring the execution of applications. The Aneka service stack provides services for infrastructure management, application execution management, accounting, licensing, and security. For more information we refer you to Vecchiola et al. [14].

Aneka's Dynamic Resource Provisioning service enables horizontal scaling depending on the overall load in the cloud. The platform is thus

elastic in nature and can provision additional resources on-demand from external physical or virtualized resource pools, in order to meet the QoS requirements of applications. I

n a typical scenario, Aneka would acquire new virtualized resources from external clouds such as Amazon EC2, in order to meet the minimum waiting time of applications submitted to Aneka. Such a scenario would arise when the current load in the cloud is high, and there is a lack of available resources to timely process all jobs.

The development environment provides a rich set of APIs for developing applications that can utilize free resources of the underlying infrastructure.

These APIs expose different programming abstractions, such as the task model, thread model, and MapReduce [22]. The task programming model is of particular importance to the current discussion. It models “independent bag of tasks” (BoT) applications that are composed of a collection of work units independent of each other, and it may be executed in any given order. One of the benefits of the task programming model is its simplicity, making it easy to run legacy applications on the cloud.

An application using the task model composes one or more task instances and forwards them as work units to the scheduler. The scheduling service currently supports the First-In-First-Out, First-In-First-Out with Backfilling, Clock-Rate Priority, and Preemption-Based Priority Queue scheduling algorithms.

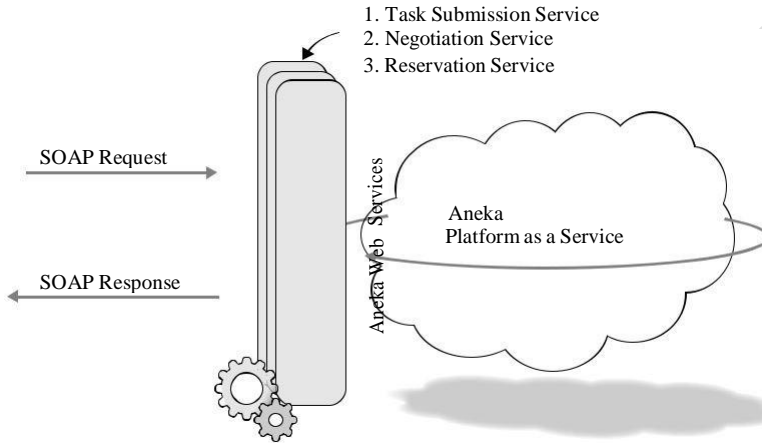
The runtime environment also provides two specialized services to support this model: the task scheduling service and the task execution service.

The storage service provides a temporary repository for application files—that is, input files that are required for task execution, and output files that are the result of execution. Prior to dispatching work units, any files required are staged-in to the storage service from the remote location. This remote location can be either the client machine, a remote FTP server, or a cloud storage service such as Amazon S3.

The work units are then dispatched to executors, which download the files before execution. Any output files produced as a result of the execution are uploaded back to the storage service. From here they are staged-out to the remote storage location.

2. Aneka Web Services

Aneka exposes three SOAP Web services for service negotiation, reservation, and task submission, as depicted in Figure 12.4. The negotiation and reservation services work in concert, and they provide interfaces for negotiating



resource use and reserving them in Aneka for predetermined timeslots. As such, these services are only useful when Aneka has limited resources to work with and no opportunities for provisioning additional resources. The task Web service provides a SOAP interface for executing jobs on Aneka. Based on the task programming model, this service allows remote clients to submit jobs, monitor their status, and abort jobs.

3.General Approach

Traditional WfMSs were designed with a centralized architecture and were thus tied to a single machine. Moving workflow engines to clouds requires (a) architectural changes and (b) integration of cloud management tools.

Architectural Changes. Most components of a WfMS can be separated from the core engine so that they can be executed on different cloud services. Each separated component could communicate with a centralized or replicated workflow engine using events. The manager is responsible for coordinating the distribution of load to its subcomponents, such as the Web server, persistence, monitoring units, and so forth.

In our WfMS, we have separated the components that form the architecture into the following: user interface, core, and plug-ins.

The user interface can now be coupled with a Web server running on a “large” instance of cloud that can handle increasing number of users.

The Web request from users accessing the WfMS via a portal is thus offloaded to a different set of resources.

Similarly, the core and plug-in components can be hosted on different types of instances separately. Depending on the size of the workload from users, these components could be migrated or replicated to other resources, or reinforced with additional resources to satisfy the increased load. Thus, employing distributed modules of the WfMS on the basis of application requirements helps scale the architecture.

Integration of Cloud Management Tools. As the WfMS is broken down into components to be hosted across multiple cloud resources, we need a mechanism to (a) access, transfer, and store data and (b) enable and monitor executions that can utilize this approach of scalable distribution of components.

The cloud service provider may provide APIs and tools for discovering the VM instances that are associated to a user's account. Because various types of instances can be dynamically created, their characteristics such as CPU capacity and amount of available memory are a part of the cloud service provider's specifications. Similarly, for data storage and access, a cloud may provide data sharing, data movement, and access rights management capabilities to user's applications. Cloud measurement tools may be in place to account for the amount of data and computing power used, so that users are charged on the pay-per-use basis. A WfMS now needs to access these tools to discover and characterize the resources available in the cloud. It also needs to interpret the access rights (e.g., access control lists provided by Amazon), use the data movement APIs, and share mechanisms between VMs to fully utilize the benefits of moving to clouds. In other words, traditional catalog services such as the Globus Monitoring and Discovery Service (MDS) [23], Replica Location Services, Storage Resource Brokers, Network Weather Service [24], and so on could be easily replaced by more user-friendly and scalable tools and APIs associated with a cloud service provider. We describe some of these tools in the following section.

4.Tools for Utilizing Clouds in WfMS

The range of tools and services offered by cloud providers play an important role in integrating WfMSs with clouds (Figure 12.5). Such services can facilitate in the deployment, scaling, execution, and monitoring of workflow systems. This section discusses some of the tools and services offered by various service providers that can complement and support WfMSs.

A WfMS manages dynamic provisioning of compute and storage resources in the cloud with the help of tools and APIs provided by service providers. The provisioning is required to dynamically scale up/down according to application requirements. For instance, data-intensive workflow applications may require

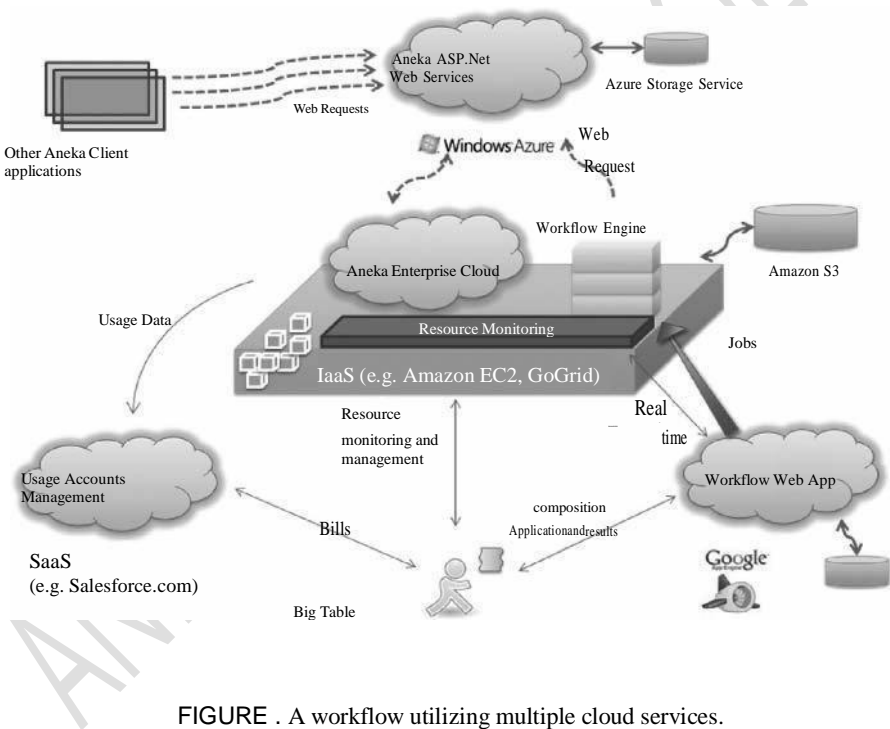


FIGURE . A workflow utilizing multiple cloud services.

large amount of disk space for storage. A WfMS could provision dynamic volumes of large capacity that could be shared across all instances of VMs (similar to snapshots and volumes provided by Amazon). Similarly, for compute-intensive tasks in an workflow, a WfMS could provision specific instances that would help accelerate the execution of these compute-intensive tasks.

A WfMS implements scheduling policies to assign tasks to resources based on applications' objectives. This task-resource mapping is dependent on several factors: compute resource capacity, application requirements, user's QoS, and so forth. Based on these objectives, a WfMS could also direct a VM provisioning system to consolidate data center loads by migrating VMs so that it could make scheduling decisions based on locality of data and compute resources.

A persistence mechanism is often important in workflow management systems and for managing metadata such as available resources, job queues, job status, and user data including large input and output files. Technologies such as Amazon S3, Google's BigTable, and the Windows Azure Storage Services can support most storage requirements for workflow systems, while also being scalable, reliable, and secure. If large quantities of user data are being dealt with, such as a large number of brain images used in functional magnetic resonance imaging (fMRI) studies [12], transferring them online can be both expensive and time-consuming. In such cases, traditional post can prove to be cheaper and faster. Amazon's AWS Import/Export⁵ is one such service that aims to speed up data movement by transferring large amounts of data in portable storage devices. The data are shipped to/from Amazon and offloaded into/from S3 buckets using Amazon's high-speed internal network. The cost savings can be significant when transferring data on the order of terabytes.

Most cloud providers also offer services and APIs for tracking resource usage and the costs incurred. This can complement workflow systems that support budget-based scheduling by utilizing real-time data on the resources used, the duration, and the expenditure.

This information can be used both for making scheduling decisions on subsequent jobs and for billing the user at the completion of the workflow application.⁶

Cloud services such as Google App Engine and Windows Azure provide platforms for building scalable interactive Web applications. This makes it relatively easy to port the graphical components of a workflow management system to such platforms while benefiting from their inherent scalability and reduced administration. For instance, such components deployed on Google App Engine can utilize the same scalable systems that drive Google applications, including technologies such as BigTable [25] and GFS .

MAJOR MAPREDUCE IMPLEMENTATIONS FOR THE CLOUD

In the following sections, we will introduce some of the major MapReduce implementations around the world as shown in Table 14.2, and we will provide a comparison of these different implementations, considering their functionality, platform, the associated storage system, programming environment, and so on, as shown in Table 14.3.

Hadoop

Hadoop [7] is a top-level Apache project, being built and used by a community of contributors from all over the world [13]. It was advocated by industry's premier Web players—Google, Yahoo!, Microsoft, and Facebook—as the engine to power the cloud [14]. The Hadoop project is stated as a collection of various subprojects for reliable, scalable distributed computing [7]. It is defined as follows [7]:

TABLE : MapReduce Cloud Implementations

Owner	Imp Name and Website	Start Time	Last Release	Distribution Model
Google	Google MapReduce http://labs.google.com/papers/mapreduce.html	2004	—	Internal use by Google
Apache	Hadoop http://hadoop.apache.org/	2004	Hadoop0.20.0 April 22, 2009	Open source
GridGain	GridGain http://www.gridgain.com/	2005	GridGain 2.1.1 February 26, 2009	Open source
Nokia	Disco http://discoproject.org/	2008	Disco 0.2.3 September 9, 2009	Open source
Geni.com	SkyNet http://skynet.rubyforge.org	2007	Skynet0.9.3 May 31, 2008	Open source
Manjrasoft	MapReduce.net (Optional service of Aneka) http://www.manjrasoft.com/products.html	2008	Aneka 1.0 March 27, 2009	Commercial

HadoopMapReduce Overview. The Hadoop common [7], formerly Hadoop core, includes file System, RPC, and serialization libraries and provides the basic services for building a cloud computing environment with commodity hardware. The two fundamental subprojects are the MapReduce framework and the Hadoop Distributed File System (HDFS).

The Hadoop Distributed File System is a distributed file system designed to run on clusters of commodity machines. It is highly fault-tolerant and is appropriate for data-intensive applications as it provides high speed access the application data.

The Hadoop MapReduce framework is highly reliant on its shared file system (i.e., it comes with plug-ins for HDFS, CloudStore [15], and Amazon Simple Storage Service S3 [16]).

The Map/Reduce framework has master/slave architecture. The master, called JobTracker, is responsible for (a) querying the NameNode for the block locations, (b) scheduling the tasks on the slave which is hosting the task's blocks, and (c) monitoring the successes and failures of the tasks. The slaves, called TaskTracker, execute the tasks as directed by the master.

Hadoop Communities. Yahoo! has been the largest contributor to the Hadoop project [13]. Yahoo! uses Hadoop extensively in its Web search and advertising businesses [13]. For example, in 2009, Yahoo! launched, according to them, the world's largest Hadoop

production application, called Yahoo! Search Webmap. The Yahoo! Search Webmap runs on a more than 10,000 core Linux cluster and produces data that are now used in every Yahoo! Web search query .

TABLE . Some Major Enterprise Solutions Based on Hadoop

Or Name	Solution and Website	Brief Description
Yahoo!	Yahoo! Distribution of Hadoop, http://developer.yahoo.com/hadoop/distribution/	The Yahoo! distribution is based entirely on code found in the Apache Hadoop project. It includes code patches that Yahoo! has added to improve the stability and performance of their clusters. In all cases, these patches have already been contributed back to Apache.
Cloudera	Cloudera Hadoop Distribution, http://www.cloudera.com/	Cloudera provides enterprise-level support to users of Apache Hadoop. The Cloudera Hadoop Distribution is an easy-to-install package of Hadoop software. It includes everything you need to configure and deploy Hadoop using standard Linux system administration tools. In addition, Cloudera provides a training program aimed at producers and users of large volumes of data.
Amazon	Amazon Elastic MapReduce, http://aws.amazon.com/elasticmapreduce/	“Web service that enables businesses, researchers, data analysts, and developers to easily and cost-effectively process vast amounts of data. It utilizes a hosted Hadoop framework running on the web-scale infrastructure of Amazon Elastic Compute Cloud (Amazon EC2)[17] and Amazon Simple Storage Service (Amazon S3).”
Sun Microsy Stems	Hadoop Live CD, http://opensolaris.org/os/project/livehadoop/	This project’s initial CD development tool aims to provide users who are new to Hadoop with a fully functional Hadoop cluster that is easy to start up and use.
IBM	Blue Cloud, http://www-03.ibm.com/press/us/en/pressrelease/22613.wss	Targets clients who want to explore the extreme scale of cloud computing infrastructures quickly and easily. “Blue Cloud will include Xen and PowerVM virtualized Linux operating system images and Hadoop parallel workload scheduling. It is supported by IBM Tivoli software that manages servers to ensure optimal performance based on demand.”

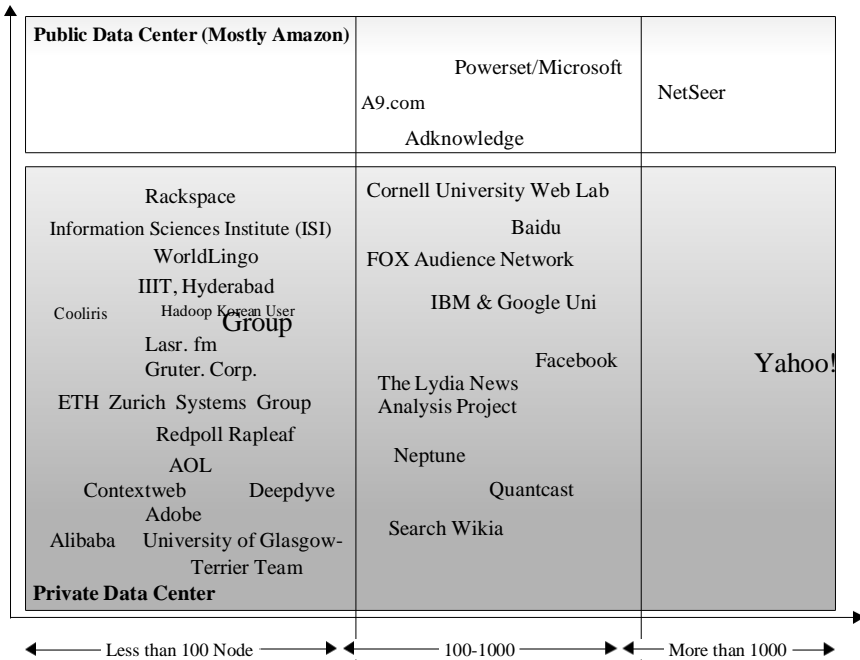


FIGURE . Organizations using Hadoop to run distributed applications, along with their cluster scale.

aforementioned vendors, many other organizations are using Hadoop solutions to run large distributed computations as shown in Figure .

Disco

Disco is an open-source MapReduce implementation developed by Nokia [21]. The Disco core is written in Erlang, while users of Disco typically write jobs in Python. Disco was started at Nokia Research Center as a lightweight frame-work for rapid scripting of distributed data processing tasks. Furthermore, Disco has been successfully used, for instance, in parsing and reformatting data, data clustering, probabilistic modeling, data mining, full-text indexing, and log analysis with hundreds of gigabytes of real-world data.

Disco is based on the master-slave architecture as shown is Figure 14.5. When the Disco master receives jobs from clients, it adds them to the job queue, and runs them in the cluster when CPUs become available. On each node there is a Worker supervisor that is responsible for spawning and monitoring all the running Python worker processes withi

n that node. The Python worker runs the assigned tasks and then sends the addresses of the resulting files to the master through their supervisor.

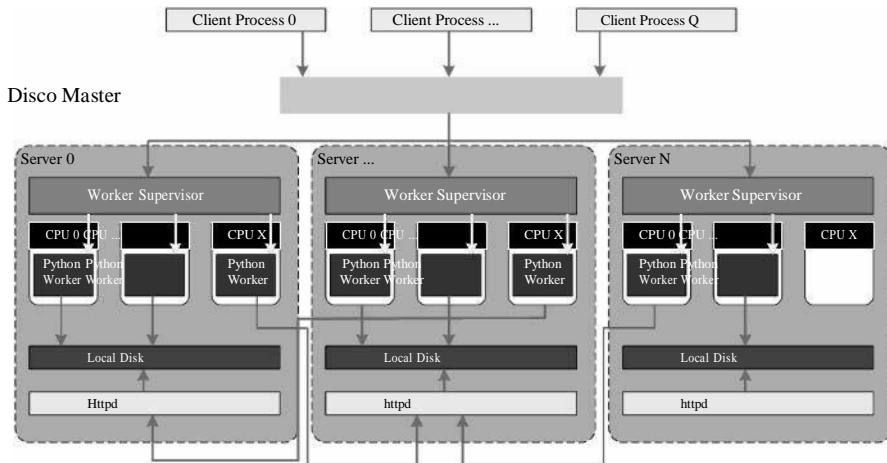


FIGURE . Architecture of Disco [21].

An “httpd” daemon (Web server) runs on each node which enables a remote Python worker to access files from the local disk of that particular node.

Mapreduce.NET

Mapreduce.NET [22] is a realization of MapReduce for the .NET platform. It aims to provide support for a wider variety of data-intensive and compute-intensive applications (e.g., MRPGA is an extension of MapReduce for GA applications based on MapReduce.NET [23]).

MapReduce.NET is designed for the Windows platform, with emphasis on reusing as many existing Windows components as possible. As shown in Figure 14.6, the MapReduce.Net runtime library is assisted by several components services from Aneka [24, 25] and runs on WinDFS.

Aneka is a .NET-based platform for enterprise and public cloud computing. It supports the development and deployment of .NET-based cloud applications in public cloud environments, such as Amazon EC2.

Besides Aneka, MapReduce.NET is using WinDFS, a distributed storage service over the .NET platform. WinDFS manages the stored data by providing an object-based interface with a flat name space. Moreover, MapReduce.NET can also work with the Common Internet File System (CIFS) or NTFS.

Skynet

Skynet [17, 26] is a Ruby implementation of MapReduce, created by Geni. Skynet is “an adaptive, self-upgrading, fault-tolerant, and fully distributed system with no single point of failure” [17]. At the heart of Skynet is plug-in based message queue architecture, with the message queuing allowing workers to

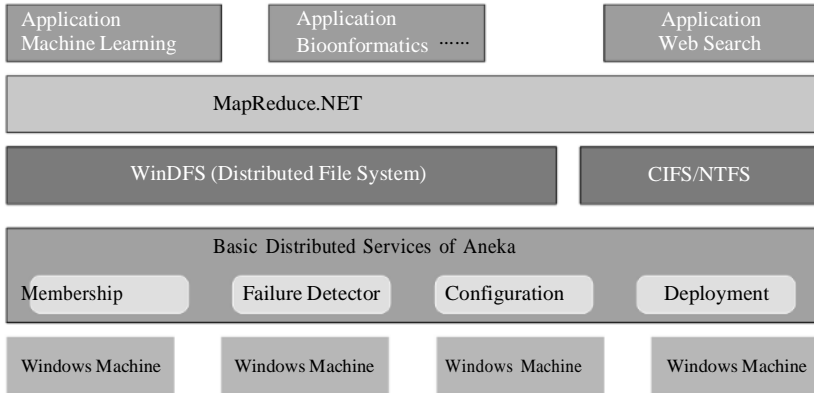


FIGURE . Architecture of Mapreduce.NET [22].

watch out for each other. If a worker fails, another worker will notice and pick up that task. Currently, there are two message queue implementations available: one built on Rinda [27] that uses Tuplespace [28] and one built on MySQL.

Skynet works by putting “tasks” on a message queue that are picked up by skynet workers. Skynet workers execute the tasks after loading the code at startup; Skynet tells the worker where all the needed code is. The workers put their results back on the message queue.

GridGain

GridGain [29] is an open cloud platform, developed in Java, for Java. GridGain enables users to develop and run applications on private or public clouds. The MapReduce paradigm is at core of what GridGain does. It defines the process of splitting an initial task into multiple subtasks, executing these subtasks in parallel and aggregating (reducing) results back to one final result. New features have been added in the GridGain MapReduce implementation such as: distributed task session, checkpoints for long running tasks, early and late load balancing, and affinity co-location with data grids.

MAPREDUCE IMPACTS AND RESEARCH DIRECTIONS

Since J. Dean and S. Ghemawat proposed the MapReduce model [4], it has received much attention from both industry and academia. Many projects are exploring ways to support MapReduce on various types of distributed architecture and for a wider range of applications as shown in Figure 14.7.

For instance, QT Concurrent [30] is a C11 library for multi-threaded application; it provides a MapReduce implementation for multi-core computers. Stanford’s Phoenix [31] is a MapReduce implementation that targets

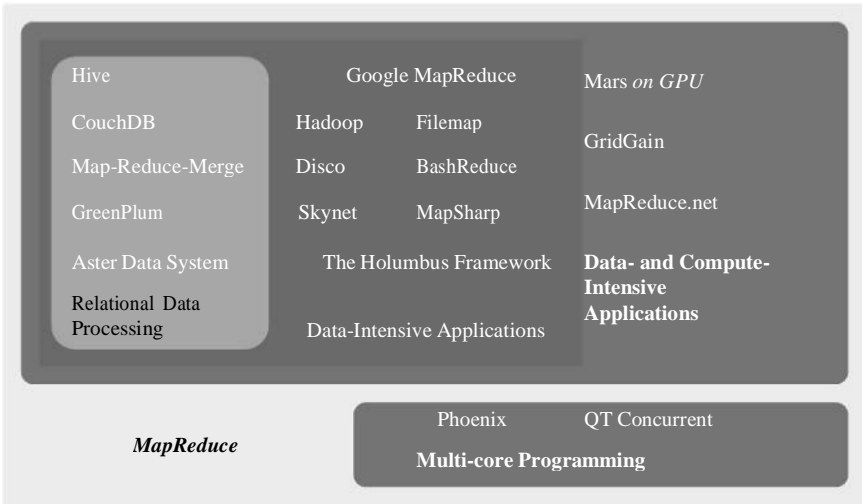


FIGURE . MapReduce different implementations.

shared memory architecture, while Kruijf and Sankaralingam implemented MapReduce for the Cell B.E. architecture [32]. Mars [33] is a MapReduce framework on graphic processors (GPUs). The Mars framework aims to provide a generic framework for developers to implement data- and computation-intensive tasks correctly, efficiently, and easily on the GPU.

Hadoop [7], Disco [21], Skynet [26], and GridGain [29] are open-source implementations of MapReduce for large-scale data processing. Map-Reduce-Merge [34] is an extension on MapReduce. It adds to MapReduce a merge phase to easily process data relationships among heterogeneous datasets. Microsoft Dryad [35] is a distributed execution engine for coarse-grain data parallel applications. In Dryad, computation tasks are expressed as directed acyclic graph (DAG).

Other efforts [36, 37] focus on enabling MapReduce to support a wider range of applications. S. Chen and S. W. Schlosser from Intel are working on making MapReduce suitable for performing earthquake simulation, image processing and general machine learning computations [36]. MRPSO [38] utilizes Hadoop to parallelize a compute-intensive application, called Particle Swarm Optimization. Research groups from Cornell, Carnegie Mellon, University of Maryland, and PARC are also starting to use Hadoop for both Web data and non-data-mining applications, like seismic simulation and natural language processing [39].

At present, many research institutions are working to optimize the performance of MapReduce for the cloud. We can classify these works in two directions:

The first one is driven by the simplicity of the MapReduce scheduler. In Zaharia et al. [40] the authors introduced a new scheduling algorithm called the

Longest Approximate Time to End (LATE) to improve the performance of Hadoop in a heterogeneous environment by running “speculative” tasks—that is, looking for tasks that are running slowly and might possibly fail—and replicating them on another node just in case they don’t perform. In LATE, the slow tasks are prioritized based on how much they hurt job response time, and the number of speculative tasks is capped to prevent thrashing.

The second one is driven by the increasing maturity of virtualization technology—for example, the successful adoption and use of virtual machines (VMs) in various distributed systems such as grid [41] and HPC applications [42, 43]. To this end, some efforts have been proposed to efficiently run MapReduce on VM-based cluster, as in Cloudlet [44] and Tashi [45].

A MODEL FOR FEDERATED CLOUD COMPUTING

In our model for federated cloud computing we identify two major types of actors: Service Providers (SPs) are the entities that need computational resources to offer some service. However, SPs do not own these resources; instead, they lease them from Infrastructure Providers (IPs), which provide them with a seemingly infinite pool of computational, network, and storage resources.

A Service Application is a set of software components that work collectively to achieve a common goal. Each component of such service applications executes in a dedicated VEE. SPs deploy service applications in the the cloud by providing to a IP, known as the primary site, with a Service Manifest—that is, a document that defines the structure of the application as well as the contract and SLA between the SP and the IP.

To create the illusion of an infinite pool of resources, IPs shared their unused capacity with each other to create a federation cloud. A Framework Agreement

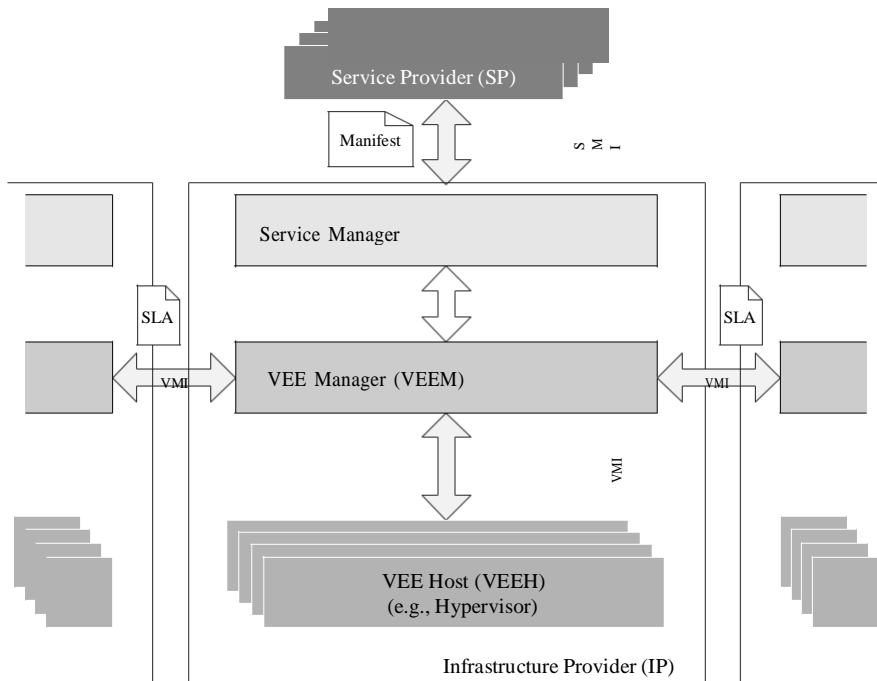


FIGURE . The RESERVOIR architecture: major components and interfaces.

is document that defines the contract between two IPs—that is, it states the terms and conditions under which one IP can use resources from another IP.

Within each IP, optimal resource utilization is achieved by partitioning physical resources, through a virtualization layer, into Virtual Execution Environments (VEEs)—fully isolated runtime environments that abstract away the physical characteristics of the resource and enable sharing. We refer to the virtualized computational resources, alongside the virtualization layer and all the management enablement components, as the Virtual Execution Environment Host (VEEH).

With these concepts in mind, we can proceed to define a reference architecture for federated cloud computing. The design and implementation of such architecture are the main goals of the RESERVOIR European research project. The RESERVOIR architecture Which operations are required may be related to the amount of information that is exposed by the remote sites; access to more information may also increase the possibility and need to manipulate the deployed VEEs.

Federation Scenarios

In this section, a number of federation scenarios are presented, ranging from a baseline case to a full-featured federation. These scenarios have various requirements on the underlying architecture, and we use the features presented in previous section as the basis for differentiating among them.

The baseline federation scenario provides only the very basic required for supporting opportunistic placement of VEEs at a remote site. Migration is not supported, nor does it resize the VEEs once placed at the remote site. Advanced features such as virtual networks across site boundaries are also not supported. The baseline federation should be possible to build on top of most public cloud offerings, which is important for interoperability. The basic federation scenario includes a number of features that the baseline federation does not, such as framework agreements, cold migration, and retention of public IP addresses. Notably missing is (a) support for hot migration and (b) cross-site virtual network functionality. This scenario offers a useful cloud computing federation with support for site collaboration in terms of framework agreements without particularly high technological requirements on the underlying architecture in terms of networking support. The advanced federation scenario offers advanced functionality such as cross-site virtual network support. The feature most notably missing is hot migration, and the monitoring system also does not disclose VEE substate metadata information. The full-featured federation scenario offers the most complete set of features, including hot migration of VEEs.

Layers Enhancement for Federation

Taking into account the different types of federation, a summary of the features needed in the different layers of the RESERVOIR architecture to achieve federation is presented.

Service Manager. The baseline federation is the most basic federation scenario, but even here the SM must be allowed to specify placement restrictions when a service is deployed. Deployment restrictions are associated to an specific VEE (although the restriction expression could involve other VEEs, as can be seen in the affinity restrictions above) and passed down to the VEEM along with any other specific VEE metadata when the VEE is issued for creation through VMI. They specify a set of constraints that must be held when the VEE is created, so they can be seen as some kind of “contour conditions” that determine the domain that can be used by the placement algorithm run at VEEM layer. Two kinds of

deployment restrictions are envisioned: First, there are affinity restrictions, related to the relations between VEEs; and second, there can be site restrictions, related to sites.

In the basic federation scenario, federation uses framework agreement (FA) between organizations to set the terms and conditions for federation. Framework agreements are negotiated and defined by individuals, but they are encoded at the end in the service manager (SM)—in particular, within the business information data base (BIDB). The pricing information included in the FA is used by the SM to calculate the cost of resources running in remote systems (based on the aggregated usage information that it received from the local VEEM) and correlate this information with the charges issued by those remote sites. The SM should be able to include as part of the VEE metadata a “price hint vector” consisting on a sequence of numbers, each one representing an estimation of the relative cost of deploying the VEE on each federated site. The SM calculate this vector based on the FA established with the other sites.

Given that the advanced federation scenario supports migration, the placement restrictions have to be checked not only at service deployment time but also for migration. In addition, the SM could update the deployment restrictions during the service lifespan, thereby changing the “contour conditions” used by the placement algorithm. When the VEE is migrated across sites, its deployment restrictions are included along with any other metadata associated with the VEE. On the other hand, no additional functionality is needed from the service manager to implement the full-featured federation.

Virtual Execution Environment Manager. Very little is needed in the baseline federation scenario of the VEEM. The only requirement will be the ability to deploy a VEE in the remote site, so it will need a plug-in that can communicate with the remote cloud by invoking the public API. This will satisfy the opportunistic placement requirement. For the different features offered by the basic federation scenario, the VEEM will need framework agreement, since it is necessary that the VEEM implement a way to tell whether it can take care of the VEE or not, attending to the SLAs defined in the framework agreement. The best module in the VEEM for the SLA evaluation to take place is the admission control of the policy engine. Also, cold migration is needed; therefore the VEEM needs the ability to signal the hypervisor to save the VEE state (this is part of the VEEM life-cycle module) and also the ability to transfer the state files to the remote site. Additionally, the VEEM need to be able to signal the hypervisor to restore the VEE state and resume its execution (also part of the VEEM life-cycle module). Regarding advance resource reservation support, the policy engine must be capable of reserving capacity in the physical infrastructure given a timeframe for certain VEEs.

In the advanced federation scenario, the ability to create cross-site virtual networks for the VEEs has to be achieved using the functionality offered by the virtual application network (VAN) as part of the virtual host interface API. Therefore, the VEEM needs to correctly interface with the VAN and be able to express the virtual network characteristics in a VEEM-to-VEEM connection.

In the full-featured federation scenario the live migration feature offered by this scenario will need to be supported also in the VHI API. The VEEM will just need to invoke the functionality of live migration to the hypervisor part of the VHI API to achieve live migration across administrative domains.

Virtual Execution Environment Host. The ability to monitor a federation is needed. The RESERVOIR monitoring service supports the asynchronous monitoring of a cloud data centers⁰ VEEHs, their VEEs, and the applications running inside the VEEs. To support federation, the originating data center must be able to monitor VEEs and their applications running at a remote site. When an event occurs related to a VEE running on a remote site, it is published and a remote proxy forwards the request to the subscribing local proxy, which in turn publishes the event to the waiting local subscribers. The monitoring framework is agnostic to type and source of data being monitored and supports the dynamic creation of new topics.

No further functionality is required for the basic federation in the VEEH apart from the features described for the baseline scenario. On the other hand, for the advanced federation one, several features are needed. First, it must have the ability to implement federated network service with virtual application network (VANs), a novel overlay network that enables virtual network services across subnets and across administrative boundaries [8,9]. VANs enables the establishment of large-scale virtual networks, free of any location dependency, that in turn allows completely “migratable” virtual networks. (1) The offered virtual network service is fully isolated, (2) it enables sharing of hosts, network devices, and physical connections, and (3) hides network related physical characteristics such as link throughputs, location of hosts, and so forth. Also, the ability to do federated migration with non-shared storage service is required. RESERVOIR enhances the standard VM migration capability typically available in every modern hypervisor with support for environments in which the source and the destination hosts do not share storage; typically the disk(s) of the migrated VM resided in the shared storage.

Regarding the full-featured federation scenario, hot migration is the functionality that affects the most what is demanded from VEEH in this scenario. RESERVOIR's separation principle requires that each RESERVOIR site be an autonomous entity. Site configuration, topology, and so on, are not shared between sites. So one site is not aware of the host addresses on another site. However, currently VM migration between hosts require that the source and destination hypervisors know each other's addresses and transfer a VM directly from the source host to the destination host. In order to overcome this apparent contradiction, RESERVOIR introduces a novel federated migration channel to transfer a VEE from one host to another host without directly addressing the destination host. Instead of transferring the VEE directly to the destination host, it passes through proxies at the source site and destination site, solving the unknown hypervisor location problem.

TRADITIONAL APPROACHES TO SLO MANAGEMENT

Traditionally, load balancing techniques and admission control mechanisms have been used to provide guaranteed quality of service (QoS) for hosted web applications. These mechanisms can be viewed as the first attempt towards managing the SLOs. In the following subsections we discuss the existing approaches for load balancing and admission control for ensuring QoS.

Load Balancing

The objective of a load balancing is to distribute the incoming requests onto a set of physical machines, each hosting a replica of an application, so that the

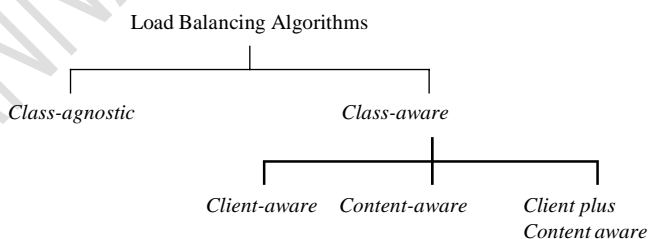


FIGURE General taxonomy of load-balancing algorithms.

load on the machines is equally distributed [4]. The load balancing algorithm executes on a physical machine that interfaces with the clients. This physical machine, also called the front-end node, receives the incoming requests and distributes these requests to different physical machines for further execution. This set of physical machines is responsible for serving the incoming requests and are known as the back-end nodes. Typically, the algorithm executing on the front-end node is agnostic to the nature of the request. This means that the front-end node is neither aware of the type of client from which the request originates nor aware of the category (e.g., browsing, selling, payment, etc.) to which the request belongs to. This category of load balancing algorithms is known as class-agnostic. There is a second category of load balancing algorithms that is known as class-aware. With class-aware load balancing and requests distribution, the front-end node must additionally inspect the type of client making the request and/or the type of service requested before deciding which back-end node should service the request. Inspecting a request to find out the class or category of a request is difficult because the client must first establish a connection with a node (front-end node) that is not responsible for servicing the request. Figure 16.5 shows the general taxonomy of different load-balancing algorithms.

Admission Control

Admission control algorithms play an important role in deciding the set of requests that should be admitted into the application server when the server experiences “very” heavy loads [5, 6]. During overload situations, since the response time for all the requests would invariably degrade if all the arriving requests are admitted into the server, it would be preferable to be selective in identifying a subset of requests that should be admitted into the system so that the overall pay-off is high. The objective of admission control mechanisms, therefore, is to police the incoming requests and identify a subset of incoming requests that can be admitted into the system when the system faces overload situations. Figure 16.6 shows the general taxonomy of the admission control mechanisms. The algorithms proposed in the literature are broadly categorized

TYPES OF SLA

Service-level agreement provides a framework within which both seller and buyer of a service can pursue a profitable service business relationship. It outlines the broad understanding between the service provider and the service consumer for conducting business and forms the basis for maintaining a mutually beneficial relationship. From a legal perspective, the necessary terms and conditions that bind the service provider to provide services continually to the service consumer are formally defined in SLA.

SLA can be modeled using web service-level agreement (WSLA) language specification [7]. Although WSLA is intended for web-service-based applications, it is equally applicable for hosting of applications. Service-level parameter, metric, function, measurement directive, service-level objective, and penalty are some of the important components of WSLA and are described in Table 16.1.

TABLE 16.1 Key Components of a Service-Level Agreement

Service-Level Parameter Metrics	Describes an observable property of a service whose value is measurable. These are definitions of values of service properties that are measured from a service-providing system or computed from other metrics and constants. Metrics are the key instrument to describe exactly what SLA parameters mean by specifying how to measure or compute the parameter values.
Function	A function specifies how to compute a metric's value from the values of other metrics and constants. Functions are central to describing exactly how SLA parameters are computed from resource metrics.
Measurement Directives	These specify how to measure a metric.

There are two types of SLAs from the perspective of application hosting. These are described in detail here.

Infrastructure SLA. The infrastructure provider manages and offers guarantees on availability of the infrastructure, namely, server machine, power, network connectivity, and so on. Enterprises manage themselves, their applications that are deployed on these server machines. The machines are leased to the customers and are isolated from machines of other customers. In such dedicated hosting environments, a practical example of service-level guarantees offered by infrastructure providers is shown in Table 16.2.

Application SLA. In the application co-location hosting model, the server capacity is available to the applications based solely on their resource demands. Hence, the service providers are flexible in allocating and de-allocating computing resources among the co-located applications. Therefore, the service

providers are also responsible for ensuring to meet their customer's application SLOs. For example, an enterprise can have the following application SLA with a service provider for one of its application, as shown in Table 16.3.

It is also possible for a customer and the service provider to mutually agree upon a set of SLAs with different performance and cost structure rather than a single SLA. The customer has the flexibility to choose any of the agreed SLAs from the available offerings. At runtime, the customer can switch between the different SLAs.

However, from the SLA perspective there are multiple challenges for provisioning the infrastructure on demand. These challenges are as follows:

- a. The application is a black box to the MSP and the MSP has virtually no knowledge about the application runtime characteristics. Therefore, the MSP needs to determine the right amount of computing resources required for different components of an application at various workloads.
- b. The MSP needs to understand the performance bottlenecks and the scalability of the application.
- c. The MSP analyzes the application before it goes on-live. However, subsequent operations/enhancements by the customer's to their applications or auto updates beside others can impact the performance of the applications, thereby making the application SLA at risk.
- d. The risk of capacity planning is with the service provider instead of the customer. If every customer decides to select the highest grade of SLA simultaneously, there may not be a sufficient number of servers for provisioning and meeting the SLA obligations of all the customers.

LIFE CYCLE OF SLA

Each SLA goes through a sequence of steps starting from identification of terms and conditions, activation and monitoring of the stated terms and conditions, and eventual termination of contract once the hosting relationship ceases to exist. Such a sequence of steps is called SLA life cycle and consists of the following five phases:

1. Contract definition
2. Publishing and discovery
3. Negotiation
4. Operationalization
5. De-commissioning

Here, we explain in detail each of these phases of SLA life cycle.

Contract Definition. Generally, service providers define a set of service offerings and corresponding SLAs using standard templates. These service offerings form a catalog. Individual SLAs for enterprises can be derived by customizing these base SLA templates.

Publication and Discovery. Service provider advertises these base service offerings through standard publication media, and the customers should be able to locate the service provider by searching the catalog. The customers can search different competitive offerings and shortlist a few that fulfill their requirements for further negotiation.

Negotiation. Once the customer has discovered a service provider who can meet their application hosting need, the SLA terms and conditions need to be mutually agreed upon before signing the agreement for hosting the application. For a standard packaged application which is offered as service, this phase could be automated. For customized applications that are hosted on cloud platforms, this phase is manual. The service provider needs to analyze the application's behavior with respect to scalability and performance before agreeing on the specification of SLA. At the end of this phase, the SLA is mutually agreed by both customer and provider and is eventually signed off. SLA negotiation can utilize the WS-negotiation specification [8].

Operationalization. SLA operation consists of SLA monitoring, SLA accounting, and SLA enforcement. SLA monitoring involves measuring parameter values and calculating the metrics defined as a part of SLA and determining the deviations. On identifying the deviations, the concerned parties are notified. SLA accounting involves capturing and archiving the SLA adherence for compliance. As part of accounting, the application's actual performance and the performance

guaranteed as a part of SLA is reported. Apart from the frequency and the duration of the SLA breach, it should also provide the penalties paid for each SLA violation. SLA enforcement involves taking appropriate action when the runtime monitoring detects a SLA violation. Such actions could be notifying the concerned

parties, charging the penalties besides other things. The different policies can be expressed using a subset of the Common Information Model (CIM) [9]. The CIM model is an open standard that allows expressing managed elements of data center via relationships and common objects.

De-commissioning. SLA decommissioning involves termination of all activities performed under a particular SLA when the hosting relationship between the service provider and the service consumer has ended. SLA specifies the terms and conditions of contract termination and specifies situations under which the relationship between a service provider and a service consumer can be considered to be legally ended.

SLA MANAGEMENT IN CLOUD

SLA management of applications hosted on cloud platforms involves five phases.

1. Feasibility
2. On-boarding
3. Pre-production
4. Production
5. Termination

Different activities performed under each of these phases are shown in Figure 16.7. These activities are explained in detail in the following subsections.

Feasibility Analysis

MSP conducts the feasibility study of hosting an application on their cloud platforms. This study involves three kinds of feasibility: (1) technical feasibility, (2) infrastructure feasibility, and (3) financial feasibility. The technical feasibility of an application implies determining the following:

1. Ability of an application to scale out.
2. Compatibility of the application with the cloud platform being used within the MSP's data center.
3. The need and availability of a specific hardware and software required for hosting and running of the application.
4. Preliminary information about the application performance and whether they can be met by the MSP.

Performing the infrastructure feasibility involves determining the availability of infrastructural resources in sufficient quantity so that the projected demands of the application can be met. The financial feasibility study involves determining the approximate cost to be incurred by the MSP and the price the MSP charges the customer so that the hosting activity is profitable to both of them. A feasibility report consists of the results of the above three feasibility studies. The report forms the basis for further communication with the customer. Once the provider and customer agree upon the findings of the report, the outsourcing of the application hosting activity proceeds to the next phase, called “on-boarding” of application. Only the basic feasibility of hosting an application has been carried in this phase. However, the detailed runtime characteristics of the application are studied as part of the on-boarding activity.

On-Boarding of Application

Once the customer and the MSP agree in principle to host the application based on the findings of the feasibility study, the application is moved from the customer servers to the hosting platform. Moving an application to the MSP’s hosting platform is called on-boarding [10]. As part of the on-boarding activity, the MSP understands the application runtime characteristics using runtime profilers. This helps the MSP to identify the possible SLAs that can be offered to the customer for that application. This also helps in creation of the necessary policies (also called rule sets) required to guarantee the SLOs mentioned in the application SLA. The application is accessible to its end users only after the on-boarding activity is completed.

On-boarding activity consists of the following steps:

- a. Packing of the application for deploying on physical or virtual environments. Application packaging is the process of creating deployable components on the hosting platform (could be physical or virtual). Open Virtualization Format (OVF) standard is used for packaging the application for cloud platform [11].
- b. The packaged application is executed directly on the physical servers to capture and analyze the application performance characteristics. It allows the functional validation of customer’s application. Besides, it provides a baseline performance value for the application in nonvirtual environment. This can be used as one of the data points for customer’s performance expectation and for application SLA. Additionally, it helps to identify the nature of application—that is, whether it is CPU-intensive or I/O-intensive or network-intensive and the potential performance bottlenecks.
- c. The application is executed on a virtualized platform and the application performance characteristics are noted again. Important performance characteristics like the application’s ability to scale (out and up) and performance bounds (minimum and maximum performance) are noted.

- d. Based on the measured performance characteristics, different possible SLAs are identified. The resources required and the costs involved for each SLA are also computed.
- e. Once the customer agrees to the set of SLOs and the cost, the MSP starts creating different policies required by the data center for automated management of the application. This implies that the management system should automatically infer the amount of system resources that should be allocated/de-allocated to/from appropriate components of the application when the load on the system increases/decreases. These policies are of three types: (1) business, (2) operational, and (3) provisioning. Business policies help prioritize access to the resources in case of contentions. Business policies are in the form of weights for different customers or group of customers. Operational policies are the actions to be taken when different thresholds/conditions are reached. Also, the actions when thresholds/conditions/triggers on service-level parameters are breached or about to be breached are defined. The corrective action could be different types of provisioning such as scale-up, scale-down, scale-out, scale-in, and so on, of a particular tier of an application. Additionally, notification and logging action (notify the enterprise application's administrator, etc.) are also defined. Operational policies (OP) are represented in the following format:

OP 5 collection of hCondition, Actioni

Here the action could be workflow defining the sequence of actions to be undertaken. For example, one OP is

OP 5 haverage latency of web server . 0.8 sec, scale-out the web-server tieri

It means, if average latency of the web server is more than 0.8 sec then automatically scale out the web-server tier. On reaching this threshold, MSP should increase the number of instances of the web server.

Provisioning policies help in defining a sequence of actions corresponding to external inputs or user requests. Scale-out, scale-in, start, stop, suspend, resume are some of the examples of provisioning actions. A provisioning policy (PP) is represented as

PP 5 collection of hRequest, Actioni

For example, a provisioning policy to start a web site consists of the following sequence: start database server, start web-server instance 1, followed by start the web-server instance 2, and so on. On defining these policies, the packaged applications are deployed on the cloud platform and the application is tested to validate whether the policies are able to meet the SLA requirements. This step is iterative and is repeated until all the infrastructure conditions necessary to satisfy the application SLA are identified.

Once the different infrastructure policies needed to guarantee the SLOs mentioned in the SLA are completely captured, the on-boarding activity is said to be completed.

Preproduction

Once the determination of policies is completed as discussed in previous phase, the application is hosted in a simulated production environment. It facilitates the customer to verify and validate the MSP's findings on application's runtime characteristics and agree on the defined SLA. Once both parties agree on the cost and the terms and conditions of the SLA, the customer sign-off is obtained. On successful completion of this phase the MSP allows the application to go on-live.

Production

In this phase, the application is made accessible to its end users under the agreed SLA. However, there could be situations when the managed application tends to behave differently in a production environment compared to the preproduction environment. This in turn may cause sustained breach of the terms and conditions mentioned in the SLA. Additionally, customer may request the MSP for inclusion of new terms and conditions in the SLA. If the application SLA is breached frequently or if the customer requests for a new non-agreed SLA, the on-boarding process is performed again. In the case of the former, on-boarding activity is repeated to analyze the application and its policies with respect to SLA fulfillment. In case of the latter, a new set of policies are formulated to meet the fresh terms and conditions of the SLA.

Termination

When the customer wishes to withdraw the hosted application and does not wish to continue to avail the services of the MSP for managing the hosting of its application, the termination activity is initiated. On initiation of termination, all data related to the application are transferred to the customer and only the essential information is retained for legal compliance. This ends the hosting relationship between the two parties for that application, and the customer sign-off is obtained.

AUTOMATED POLICY-BASED MANAGEMENT

This section explains in detail the operationalization of the “Operational” and “Provisioning” policies defined as part of the on-boarding activity. The policies specify the sequence of actions to be performed under different circumstances.

Operational policies specify the functional relationship between the system-level infrastructural attributes and the business level SLA goals. Knowledge of such a relationship helps in identifying the quantum of system resources to be allocated to the various components of the application for different system

attributes at various workloads, workload compositions, and operating conditions, so that the SLA goals are met. Figure 16.8 explains the importance of such a relationship. For example, consider a three-tier web application consisting of web server, application server, and database server. Each of the servers is encapsulated using a virtual machine and is hosted on virtualized servers. Furthermore, assume that the web tier and the database tier of the application have been provisioned with sufficient resources at a particular workload. The effect of varying the system resources (such as CPU) on the SLO, which in this case is the average response time for customer requests, is shown in Figure 16.8.

To understand the system resource requirements for each of the tiers of an application at different workloads necessitates the deployment of the application on a test system. The test system is used to collect the low-level system metrics such as usage of memory and CPU at different workloads, as well as to observe the corresponding high-level service level objectives such as average response time. The metrics thus collected are used to derive the functional relationship between the SLOs and low-level system attributes. These functional relations are called policies. For example, a classification technique is used to derive policies [12, 13].

The triggering of operational and provisional policies results in a set of actions to be executed by the service provider platform. It is possible that some of these actions contend for the same resources. In such a case, execution of certain actions needs to be prioritized over the execution of others. The rules that govern this prioritization of request execution in case of resource contention are specified

as a part of business policy. Some of the parameters often used to prioritize action and perform resource contention resolution are:

- The SLA class (Platinum, Gold, Silver, etc.) to which the application belongs to.
- The amount of penalty associated with SLA breach.
- Whether the application is at the threshold of breaching the SLA.
- Whether the application has already breached the SLA.
- The number of applications belonging to the same customer that has breached SLA.
- The number of applications belonging to the same customer about to breach SLA.
- The type of action to be performed to rectify the situation.

Priority ranking algorithms use these parameters to derive scores. These scores are used to rank each of the actions that contend for the same resources. Actions having high scores get higher priority and hence, receive access to the contended resources.

Furthermore, automatic operationalization of these policies consists of a set of components as shown in Figure 16.9. The basic functionality of these components is described below:

1. **Prioritization Engine.** Requests from different customers' web applications contending for the same resource are identified, and accordingly their execution is prioritized. Business policies defined by the MSP helps in identifying the requests whose execution should be prioritized in case of resource contentions so that the MSP can realize higher benefits.
2. **Provisioning Engine.** Every user request of an application will be enacted by the system. The set of steps necessary to enact the user requests are defined in the provisioning policy, and they are used to fulfill the application request like starting an application, stopping an application, and so on. These set of steps can be visualized as a workflow. Hence, the execution of provisioning policy requires a workflow engine [14].
3. **Rules Engine.** The operation policy defines a sequence of actions to be enacted under different conditions/trigger points. The rules engine evaluates the data captured by the monitoring system [15], evaluates against the predefined operation rules, and triggers the associated action if required. Rules engine and the operational policy is the key to guaranteeing SLA under a self healing system.
4. **Monitoring System.** Monitoring system collects the defined metrics in SLA. These metrics are used for monitoring resource failures, evaluating operational policies, and auditing and billing purpose.
5. **Auditing.** The adherence to the predefined SLA needs to be monitored and recorded. It is essential to monitor the compliance of SLA because

any noncompliance leads to strict penalties. The audit report forms the basis for strategizing and long-term planning for the MSP.

6. Accounting/Billing System. Based on the payment model, chargebacks could be made based on the resource utilized by the process during the operation. The fixed cost and recurring costs are computed and billed accordingly.

The interactions among these components are shown in Figure 16.9 and described below.

The policies and packaged application are deployed on the platform after completing the on-boarding activity. The customer is provided with options to start the application in any of the agreed SLAs. The application request is sent via the access layer to the system. Using the provisioning policy, the provisioning engine determines how and in what sequence the different components/tiers of an application should be started and configured. If the start operation requires a resource that is also contended by a different application request, then provisioning engine interacts with the prioritization engine to determine the request that should have access to the contended resource in case of conflict. This conflict resolution is guided by the business policy defined in the prioritization engine.

Once an application begins execution, it is continuously monitored by the monitoring system. Monitoring involves collecting statistics about the key metrics and evaluating them against the rules defined in the operational policy for validating the SLA adherence.

SLA violation triggers rules that initiate appropriate corrective action automatically. For example, whenever the performance of the application degrades and chances of violating the agreed SLO limits are high, the rules that help scale out the bottleneck tier of the application is triggered. This ensures that the performance does not degenerate to a level of violating the SLA. Periodically, the amount of resource utilized by the application is calculated. On calculating the resource utilization, the cost is computed correspondingly and the bill is generated. The bill along with the report on the performance of the application is sent to the customer.

Alternatively, the monitoring system can interact with the rules engine through an optimization engine, as shown in Figure 16.10. The role of the optimization system is to decide the migration strategy that helps optimize certain objective functions for virtual machine migration. The objective could be to minimize the number of virtual machines migrated or minimize the number of physical machines affected by the migration process. The following example highlights the importance of the optimization engine within a policy based management system [16].

Assume an initial assignment of seven virtual machines (VM) to the three physical machines (PM) at time t_1 as shown in Figure 16.11. Also, each of the three PMs has memory and CPU capacity of 100. At time t_1 , the CPU usage by VM₁, VM₂, and VM₃ on PM_A are 40, 40, and 20, respectively, and the memory

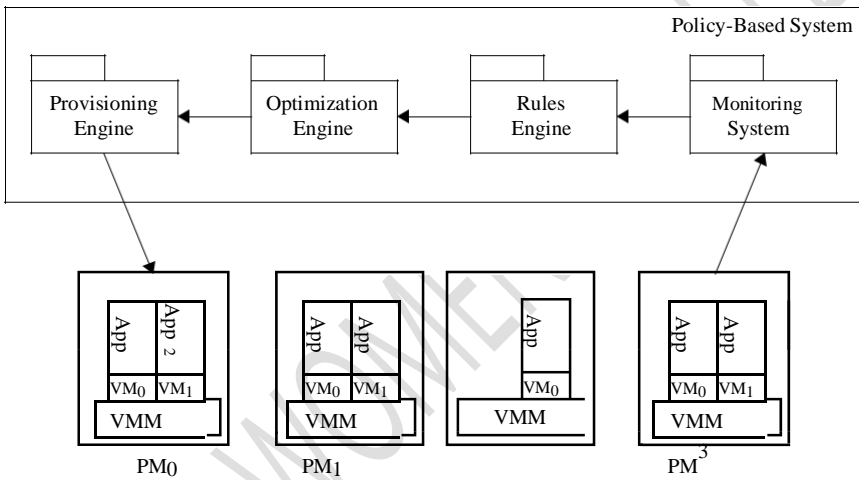


FIGURE . Importance of optimization in the policy-based management system.

consumption is 20, 10, and 40 respectively. Similarly, at time t_1 the CPU and memory requirements of VM₄, VM₅, and VM₆ on PM_B are 20, 10, 40 and 20, 40, 20, respectively. VM₇ only consumes 20% of CPU and 20% of memory on PM_C. Thus, PM_B and PM_C are underloaded but PM_A is overloaded. Assume VM₁ is the cause of the overload situation in PM_A.

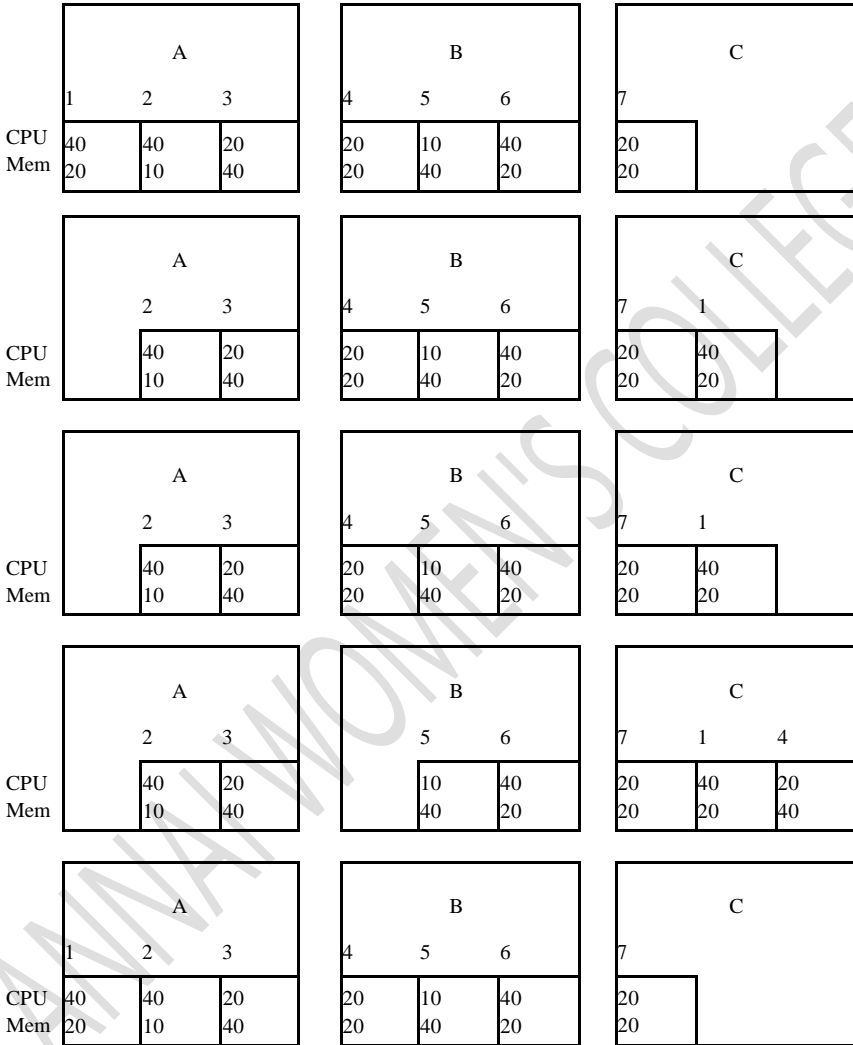


FIGURE 16.11. (a) Initial configuration of the VMs and the PMs at time t_1 . (b) Configuration resulting from event-based migration of VM₁ at time t_1 . (c) Resource requirement situation at time $t_2 . t_1$. (d) Configuration resulting from “event-based” migration of VM₄ at time $t_2 . t_1$. (e) Alternate configuration resulting from optimization-based migration at time $t_2 . t_1$.

In the above scenario, event-based migration will result in migration of VM₁ out of PM_A to PM_C. Furthermore, consider that at time t_2 ($t_2 > t_1$), PM_B is overloaded as the memory requirement of VM₄ increases to 40. Consequently, an event-based scheme results in migration of VM₄ to PM_C. At time t_3 ($t_3 > t_2$), a new VM, VM₈, with CPU and memory requirements of 70 each, needs to be allocated to one of the PMs; then a new PM, PM_D, needs to be switched on for hosting it. In such a scenario, VM₈ cannot be hosted on any of the three existing PMs: PM_A, PM_B, and PM_C. However, assume that the duration of the time window $t_2 - t_1$ is such that the QoS and SLA violations due to the continued hosting of VM₁ on PM_A are well within the permissible limits. In such a case, the migration of both VMs—VM₁ to PM_B and VM₄ to PM_A— at time t_2 ensures lesser number of PM are switched on. This results in a global resource assignment that may be better than local resource management.

UNIT-V

GRID AND CLOUD

“Grid vs Cloud” is the title of an incredible number of recent Web blogs and articles in on-line forums and magazines, where many HPC users express their own opinion on the relationship between the two paradigms. Cloud is simply presented, by its supporters, as an evolution of the grid. Some consider grids and clouds as alternative options to do the same thing in a different way. However, there are very few clouds on which one can build, test, or run compute-intensive applications. In fact it still necessary to deal with some open issues. One is when, in term of performance, a cloud is better than a grid to run a specific application. Another problem to be addressed concerns the effort to port a grid application to a cloud. In the following it will be discussed how these and other arguments suggest that we investigate the integration of grids and clouds to improve the exploitation of computing resources in HPC.

Grid and Cloud as Alternatives

Both grid and cloud are technologies that have been conceived to provide users with handy computing resources according to their specific requirements.

Grid was designed with a bottom-up approach. Its goal is to share a hardware or a software among different organizations by means of common protocols and policies. The idea is to deploy interoperable services in order to allow the access to physical resources (CPU, memory, mass storage, etc.) and to available software utilities. Users get access to a real machine. Grid resources are administrated by their owners. Authorized users can invoke grid services on remote machines without paying and without service level guaran-tees. A grid middleware provides a set of API (actually services) to program a heterogeneous, geographically distributed system.

On the other hand, cloud technology was designed using a top-down approach. It aims at providing its users with a specific high-level functionality: a storage, a computing platform, a specialized service. They get virtual resources from the cloud. The underlying hardware/software infrastructure is not exposed. The only information the user needs to know is the quality of service (QoS) of the services he

is paying for. Bandwidth, computing power, and storage represent parameters that are used for specifying the QoS and for billing. Cloud users ask for a high-level functionality (service, platform, infrastructure), pay for it, and become owners of a virtual machine. From a technological point of view, virtualization is exploited to build an insulated environment, which is configured to meet users' requirements and is exploited for easy reconfiguration and backup. A single enterprise is the owner of the cloud platform (software and underlying hardware), whereas customers become owners of the virtual resources they pay for.

Cloud supporters claim that the cloud is easy to be used, is scalable, and always gives users exactly what they want. On the other hand, grid is difficult to be used, does not give performance guarantees, is used by narrow communities of scientists to solve specific problems, and does not actually support interoperability.

Grid fans answer that grid users do not need a credit card, that around the world there are many examples of successful projects, and that a great number of computing nodes connected across the net execute large-scale scientific applications, addressing problems that could not be solved otherwise. Grid users can use a reduced set of functionalities and can develop simple applications, or they can get, theoretically, an infinite amount of resources.

As always, truth is in the middle. Some users prefer to pay since they need a specific service with strict requirements and require a guaranteed QoS. Cloud can provide this. Many users of the scientific community look for some sort of supercomputing architecture to solve intensive computations that process a huge amount of data, and they do not care about getting a guaranteed performance level. The grid can provide it. But, even on this last point, there are divergent opinions.

Grid and Cloud Integration

To understand why grids and clouds should be integrated, we have to start by considering what the users want and what these two technologies can provide. Then we can try to understand how cloud and grid can complement each other and why their integration is the goal of intensive research activities. We know that a supercomputer runs faster than a virtualized resource. For example, a LU benchmark on EC2 (the cloud platform provided by Amazon) runs slower, and some overhead is added to start VMs. On the other hand, the probability to execute an application in fixed time on a grid resource depends on many parameters

and cannot be guaranteed. As experimented in Foster, if 400 msec is the time that an EC2 requires to execute an LU benchmark, then the probability of obtaining a grid resource in less than 400 msec is very low (34%), even if the same benchmark can take less than 100 msec to complete.

If you want to get your results as soon as possible, you are adopting the cloud end-user perspective. If you want to look for the optimum resources that solve the problem, overcoming the boundaries of a single enterprise, you are using the grid perspective that aims at optimizing resources sharing and system utilization.

The integration of cloud and grid, or at least their integrated utilization, has been proposed since there is a trade-off between application turnaround and system utilization, and sometimes it is useful to choose the right compromise between them.

- Some issues to be investigated have been pointed out:
- Integration of virtualization into existing e-infrastructures
- Deployment of grid services on top of virtual infrastructures
- Integration of cloud-base services in e-infrastructures
- Promotion of open-source components to build clouds
- Grid technology for cloud federation

In light of the above, the integration of the two environments is a debated issue. At the state of the art, two main approaches have been proposed:

Grid on Cloud. A cloud IaaS (Infrastructure as a Service) approach is adopted to build up and to manage a flexible grid system. Doing so, the grid middleware runs on a virtual machine. Hence the main drawback of this approach is performance. Virtualization inevitably entails performance losses as compared to the direct use of physical resources.

Cloud on Grid: The stable grid infrastructure is exploited to build up a cloud environment. This solution is usually preferred because the cloud approach mitigates the inherent complexity of the grid. In this case, a set of grid services is offered to manage (create, migrate, etc.) virtual machines. The use of Globus workspaces ,

along with a set of grid services for the Globus Toolkit 4, is the prominent solution, as in the Nimbus project

The integration could simplify the task of the HPC user to select, to configure, and to manage resources according to the application requirements. It adds flexibility to exploit available resources, but both of the above-presented approaches have serious problems for overall system management, due to the complexity of the resulting architectures. Performance prediction, application tuning, and benchmarking are some of the relevant activities that become critical and that cannot be performed in the absence of performance evaluation of clouds.

HPC IN THE CLOUD: PERFORMANCE-RELATED ISSUES

This section will discuss the issues linked to the adoption of the cloud paradigm in the HPC context. In particular, we will focus on three different issues:

1. The difference between typical HPC paradigms and those of current cloud environments, especially in terms of performance evaluation.
2. A comparison of the two approaches in order to point out their advantages and drawbacks, as far as performance is concerned.
3. New performance evaluation techniques and tools to support HPC in cloud systems.

As outlined in the previous sections, the adoption of the cloud paradigm for HPC is a flexible way to deploy (virtual) clusters dedicated to execute HPC applications. The switch from a physical to a virtual cluster is completely transparent for the majority of HPC users, who have just terminal access to the cluster and limit themselves to “launch” their tasks.

The first and well-known difference between HPC and cloud environments is the different economic approach: (a) buy-and-maintain for HPC and (b) pay-per-use in cloud systems. In the latter, every time that a task is started, the user will be charged for the used resources. But it is very hard to know in advance which will be the resource usage and hence the cost. On the other hand, even if the global expense for a physical cluster is higher, once the system has been acquired, all the costs are fixed and predictable (in fact, they are so until the system is not faulty). It would

be great to predict, albeit approximately, the resource usage of a target application in a cloud, in order to estimate the cost of its execution.

These two issues above are strictly related, and a performance problem becomes an economic problem. Let us assume that a given application is well-optimized for a physical cluster. If it behaves on a virtual cluster as on the physical one, it will use the cloud resources in an efficient way, and its execution will be relatively cheap. This is not so trivial as it may seem, as the pay-per-use paradigm commonly used in commercial clouds (see Table 17.1) charges the user for virtual cluster up-time, not for CPU usage. Almost surprisingly, this means that processor idle time has a cost for cloud users.

For clarity's sake, it is worth presenting a simple but interesting example regarding performance and cost. Let us consider two different virtual clusters with two and four nodes, respectively. Let us assume that the application is well-optimized and that, at least for a small number of processors, it gets linear speed-up. The target application will be executed in two hours in the first cluster and in one hour in the second .

In conclusion: In clouds, performance counts two times. Low performance means not only long waiting times, but also high costs. The use of alternative cost factors (e.g., the RAM memory allocated, as for GoGrid in below Table) leads to completely different considerations and requires different application optimizations to reduce the final cost of execution.

In light of the above, it is clear that the typical HPC user would like to know how long his application will run on the target cluster and which configuration has the highest performance/cost ratio. The advanced user, on the other hand, would also know if there is a way to optimize its application so as to reduce the cost of its run without sacrificing performance. The high-end user, who cares more for performance than for the cost to be sustained, would like instead to know how to choose the best configuration to maximize the performance of his application. In other words, in the cloud world the hardware configuration is not fixed, and it is not the starting point for optimization decisions. Configurations can be easily changed in order to fit the user needs. All the three classes of users should resort to performance analysis and prediction tools. But, unfortunately, prediction tools for virtual

environments are not available, and the literature presents only partial results on the performance analysis of such systems.

An additional consequence of the different way that HPC users exploit a virtual cluster is that the cloud concept makes very different the system dimensioning—that is, the choice of the system configuration fit for the user purposes (cost, maximum response time, etc.). An HPC machine is chosen and acquired, aiming to be at the top of available technology (under inevitable money constraints) and to be able to sustain the highest system usage that may eventually be required. This can be measured in terms of GFLOPS, in terms of number of runnable jobs, or by other indexes depending on the HPC applications that will be actually executed. In other words, the dimensioning is made by considering the peak system usage. It takes place at system acquisition time, by examining the machine specifications or by assembling it using hardware components of known performance. In this phase, simple and global performance indexes are used (e.g., bandwidth and latency for the interconnect, peak FLOPS for the computing nodes, etc.)

In clouds, instead, the system must be dimensioned by finding out an optimal trade-off between application performance and used resources. As mentioned above, the optimality is a concept that is fairly different, depending on the class of users. Someone would like to obtain high performance at any cost, whereas others would privilege economic factors. In any case, as the choice of the system is not done once and for all, the dimensioning of the virtual clusters takes place every time the HPC applications have to be executed on new datasets. In clouds, the system dimensioning is a task under the control of the user, not of the system administrator. This completely changes the scenario and makes the dimensioning a complex activity, eager for performance data and indexes that can be measured fairly easily in the HPC world on physical systems, but that are not generally available for complex and rapidly changing systems as virtual clusters. Above Table summarizes the differences between HPC classical environments and HPC in clouds. To summarize the above discussion, in systems (the clouds) where the availability of performance data is crucial to know how fast your applications will run and how much you will pay, there is great uncertainty about what to measure and how to measure, and there are great difficulties when attempting to interpret the meaning of measured data.

HPC Systems and HPC on Clouds: A Performance Comparison

The second step of our analysis is a performance comparison between classical HPC systems and the new cloud paradigm. This will make it possible to point out the advantages and disadvantages of the two approaches and will enable us to understand if and when clouds can be useful for HPC.

The performance characterization of HPC systems is usually carried out by executing benchmarks. However, the only ones that make measurements of virtual clusters. The LINPACK benchmark, a so-called kernel benchmark, which aims at measuring the peak performance (in FLOPSs) of the target environment.

The NAS Parallel Benchmarks (NPB), a set of eight programs designed to help to evaluate the performance of parallel supercomputers, derived from computational fluid dynamics (CFD) applications and consisting of five kernels and three pseudo-applications. As performance index, together with FLOPS, it measures response time, network bandwidth usage, and latency. *mpptest*, a microbenchmark that measures the performance of some of the basic MPI message passing routines in a variety of different conditions. It measures (average) response time, network bandwidth usage and latency.

When these benchmarks are executed on physical machines (whether clusters or other types of parallel hardware), they give a coarse-level indication of the system potentialities. In the HPC world, these benchmarks are of common use and widely diffused, but their utility is limited. Users usually have an in-depth knowledge of the target hardware used for executing their applications, and a comparison between two different (physical) clusters makes sense only for Top500 classification or when they are acquired. HPC users usually outline the potentiality and the main features of their system through (a) a brief description of the hardware and (b) a few performance indexes obtained using some of the above-presented benchmarks. In any case, these descriptions are considered useless for application performance optimization, because they only aim at providing a rough classification of the hardware.

Recently, the benchmarking technique has been adopted in a similar way, tackling also the problem of the utility of the cloud paradigm for scientific applications. In particular, the papers focusing on the

development of applications executed in virtual clusters propose the use of a few benchmarks to outline the hardware potentialities. These results are of little interest for our comparison. On the other hand, papers that present comparisons between virtual and physical clusters use benchmarks to find out the limits of cloud environments, as discussed below. In the following, we will focus on these results.

We can start our analysis from benchmark-based comparison of virtual clusters and physical HPC systems. In the literature there are results on all three types of benchmarks mentioned above, even if the only cloud provider considered is Amazon EC2 (there are also results on private clusters, but in those cases the analysis focuses on virtual engine level and neglects the effects of the cloud environment, and so it is outside the scope of this chapter).

Napper and Bientinesi and Ostermann et al. adopted the LINPACK benchmark, measuring the GFLOPS provided by virtual clusters composed of Amazon EC2 virtual machines. Both studies point out that the values obtained in the VCs are an order of magnitude lower than equivalent solutions on physical clusters. The best result found in the literature is about 176 GFLOPS, to be compared to 37.64 TFLOPS of the last (worst) machine in Top500 list. Even if it is reasonable that VCs peak performances are far from the supercomputer ones, it is worth noting that the GFLOPS tends to decrease (being fixed the memory load) when the number of nodes increases. In other words, virtual clusters are not so efficient as physical clusters, at least for this benchmark. As shown later, the main cause of this behavior is the inadequate internal interconnect.

An analysis by real-world codes, using the NPB (NAS parallel benchmark) benchmark suite, was proposed in Walker, Ostermann et al. NPBs are a collection of MPI-based HPC applications. The suite is organized so as to stress different aspects of an HPC systems for example, computation, communication, or I/O.

Walker compared a virtual EC2 cluster to a physical cluster composed of TeraGrid machines with similar hardware configuration (i.e., the hardware under the virtual cluster was the same adopted by the physical cluster). This comparison pointed out that the overheads introduced by the virtualization layer and the cloud environment level were fairly high. It should be noted that Walker adopted for his analysis two virtual

clusters made up of a very limited number of nodes (two and four). But, even for such small systems, the applications did not scale well with the number of nodes.

The last kind of benchmark widely adopted in the literature is the MPI kernel benchmark, which measures response time, bandwidth, and latency for MPI communication primitives. These tests, proposed by almost all the authors who tried to run scientific applications on cloud-based virtual clusters, are coherent with the results presented above. In all the cases in the literature, bandwidth and, above all, latency have unacceptable values for HPC applications.

In the literature, at the best of the authors' knowledge, there are currently no other examples of virtual cluster benchmarking, even if the ongoing diffusion of the paradigm will lead probably to a fast growth of this kind of results in the next years. As mentioned above, the benchmarking technique is able to put in evidence the main drawback linked to the adoption of cloud systems for HPC: the unsatisfactory performance of the network connection between virtual clusters. In any case, the performance offered by virtual clusters is not comparable to the one offered by physical clusters.

Even if the results briefly reported above are of great interest and can be of help to get insight on the problem, they do not take into account the differences between HPC machines and HPC in the cloud, which we have summarized at the start of this section. Stated another way, the mentioned analyses simply measure global performance indexes. But the scenario can drastically change if different performance indexes are measured.

Just to start, the application response time is perhaps the performance index of great importance in a cloud context. In fact, it is a measurement of interest for the final user and, above all, has a direct impact on the cost of the application execution. An interesting consideration linked to response time was proposed by Ian Foster in his blog. The overall application response time (RT) is given by the formula $RT = 5 \text{ h job submission time} + 1 \text{ hexecution time}$.

In common HPC environments (HPC system with batch queue, grids, etc.) the job submission time may be fairly long (even minutes or hours, due to necessity to get all the required computing resources together). On the other hand, in a cloud used to run HPC workload (a virtual

cluster dedicated to the HPC user), queues (and waiting time) simply disappear. The result is that, even if the virtual cluster may offer a much lower computational power, the final response time may be comparable to that of (physical) HPC systems.

In order to take into account this important difference between physical and virtual environments, Foster suggests to evaluate the response time in terms of probability of completion, which is a stochastic function of time, and represents the probability that the job will be completed before that time. Note that the stochastic behavior mainly depends on the job submission time, whereas execution time is usually a deterministic value. So in a VC the probability of completion is a threshold function (it is zero before the value corresponding to execution time of actual task, and one after). In a typical HPC environment, which involves batch and queuing systems, the job submission time is stochastic and fairly long, thus leading to a global completion time higher than the one measured on the VC.

This phenomenon opens the way to a large adoption of the cloud approach, at least for middle- or small-dimension HPC applications, where the computation power loss due to the use of the cloud is more tolerable. In Jha et al. and in the on-line discussion it is well shown that the cloud approach could be very interesting for substituting the ecosystem of HPC clusters that are usually adopted for solving middle-dimension problems. This is a context in which the grid paradigm was never largely adopted because of the high startup overhead.

Supporting HPC in the Cloud

The above-presented analysis shows how the cloud approach has good chances to be widely adopted for HPC, even if there are limits one should be aware of, before trying to switch to virtualized systems. Moreover, the differences between “physical computing” and “virtual computing,” along with their impact on performance evaluation, clearly show that common performance indexes, techniques, and tools for performance analysis and prediction should be suitably adapted to comply with the new computing paradigm.

To support HPC applications, a fundamental requirement from a cloud provider is that an adequate service-level agreement (SLA) is granted. For HPC applications, the SLA should be different from the ones currently offered for the most common uses of cloud systems, oriented at transactional Web applications. The SLA should offer guarantees useful for the HPC user to predict his application

performance behavior and hence to give formal (or semi-formal) statements about the parameters involved. At the state of the art, cloud providers offer their SLAs in the form of a contract (hence in natural language, with no formal specification). Two interesting examples are Amazon EC2 (<http://aws.amazon.com/ec2-sla/>) and GoGrid (<http://www.gogrid.com/legal/sla.php>).

The first one (Amazon) stresses fault tolerance parameters (such as service uptime), offering guarantees about system availability. There are instead no guarantees about network behavior (for both internal and external network), except that it will “work” 95% of the time. Moreover, Amazon guarantees that the virtual machine instances will run using a dedicated memory (i.e., there will be no other VM allocated to on the physical machine using the same memory). This statement is particularly relevant for HPC users, because it is of great help for the performance predictability of applications.

On the other hand, GoGrid, in addition to the availability parameters, offers a clear set of guarantees on network parameters, as shown in the below Table. This kind of information is of great interest, even if the guaranteed network latency (order of milliseconds) is clearly unacceptable for HPC applications. GoGrid does not offer guarantees about the sharing of physical computing resources with other virtual machines.

In conclusion, even if the adoption of SLA could be (part of) a solution for HPC performance tuning, giving a clear reference for the offered virtual cluster performances, current solutions offer too generic SLA contracts or too poor values for the controlled parameters.

As regards performance measurement techniques and tools, along with their adaption for virtualized environments, it should be noted that very few performance-oriented services are offered by cloud providers or by third parties. Usually these services simply consist of more or less detailed performance monitoring tools, such as CloudWatch offered by Amazon, or CloudStatus, offered by Hyperic (and integrated in Amazon). These tools essentially measure the performance of the cloud internal or external network and should help the cloud user to tune his applications. In exactly the same way as SLAs, they can be useful only for the transactional applications that are the primary objective of cloud systems, since, at the state of the art, they do not offer any features to predict the behavior of long-running applications, such as HPC codes.

An interesting approach, although still experimental, is the one offered by solutions as C-meter and PerfCloud, which offer frameworks that dynamically benchmark the target VMs or VCs offered by the cloud. The idea is to provide a benchmark-on-demand service to take into account the extreme variability of the cloud load and to evaluate frequently its actual state. The first framework supports the GrenchMark benchmark (which generates syn-thetic workloads) and is oriented to Web applications. The second one, instead, supports many different benchmarks typical of the HPC environment (the above-mentioned NPB and MPP tests, the SkaMPI benchmark, etc.). More detailed, the PerfCloud project aims at providing performance evaluation and prediction services in grid-based clouds. Besides providing services for on-demand benchmarking of virtual clusters, the PerfCloud framework uses the benchmarking results to tune a simulator used for predict the performance of HPC applications.

DATA SECURITY IN THE CLOUD

AN INTRODUCTION TO THE IDEA OF DATA SECURITY

Taking information and making it secure, so that only yourself or certain others can see it, is obviously not a new concept. However, it is one that we have struggled with in both the real world and the digital world. In the real world, even information under lock and key, is subject to theft and is certainly open to accidental or malicious misuse. In the digital world, this analogy of lock-and-key protection of information has persisted, most often in the form of container-based encryption. But even our digital attempt at protecting information has proved less than robust, because of the limitations inherent in protecting a container rather than in the content of that container. This limitation has become more evident as we move into the era of cloud computing: Information in a cloud environment has much more dynamism and fluidity than information that is static on a desktop or in a network folder, so we now need to start to think of a new way to protect information.

Before we embark on how to move our data protection methodologies into the era of The cloud, perhaps we should stop, think, and consider the true applicability of information security and its value and scope. Perhaps we should be viewing the application of data security as less of a walled and impassable fortress and more of a sliding series of options that are more appropriately termed “risk mitigation.”

The reason that I broach this subject so early on is that I want the reader to start to view data security as a lexicon of choices, as opposed to an

on/off technology. In a typical organization, the need for data security has a very wide scope, varying from information that is set as public domain, through to information that needs some protection (perhaps access control), through to data that are highly sensitive, which, if leaked, could cause catastrophic damage, but nevertheless need to be accessed and used by selected users.

One other aspect of data security that I want to draw into this debate is the human variable within the equation. Computer technology is the most modern form of the toolkit that we have developed since human prehistory to help us improve our lifestyle. From a human need perspective, arguably, computing is no better or worse than a simple stone tool, and similarly, it must be built to fit the hand of its user. Technology built without considering the human impact is bound to fail. This is particularly true for security technology, which is renowned for failing at the point of human error.

If we can start off our view of data security as more of a risk mitigation exercise and build systems that will work with humans (i.e., human-centric), then perhaps the software we design for securing data in the cloud will be successful.

THE CURRENT STATE OF DATA SECURITY IN THE CLOUD

At the time of writing, cloud computing is at a tipping point: It has many arguing for its use because of the improved interoperability and cost savings it offers. On the other side of the argument are those who are saying that cloud computing cannot be used in any type of pervasive manner until we resolve the security issues inherent when we allow a third party to control our information. These security issues began life by focusing on the securing of access to the datacenters that cloud-based information resides in. However, it is quickly becoming apparent in the industry that this does not cover the vast majority of instances of data that are outside of the confines of the data center, bringing us full circle to the problems of having a container-based view of securing data. This is not to say that data-center security is obsolete. Security, after all, must be viewed as a series of concentric circles emanating from a resource and touching the various places that the data go to and reside. However, the very nature of cloud computing dictates that data are fluid objects, accessible from a multitude of nodes and geographic locations and, as such, must have a data security methodology that takes this into account while ensuring that this fluidity is not compromised. This apparent dichotomy—data security with open movement of data—is not as

juxtaposed as it first seems. Going back to my previous statement that security is better described as “risk mitigation,” we can then begin to look at securing data as a continuum of choice in terms of levels of accessibility and content restrictions: This continuum allows us to choose to apply the right level of protection, ensuring that the flexibility bestowed by cloud computing onto the whole area of data communication is retained.

As I write, the IT industry is beginning to wake up to the idea of content-centric or information-centric protection, being an inherent part of a data object. This new view of data security has not developed out of cloud computing, but instead is a development out of the idea of the “de-perimeterization” of the enterprise. This idea was put forward by a group of Chief Information Officers (CIOs) who formed an organization called the Jericho Forum. The Jericho Forum was founded in 2004 because of the increasing need for data exchange between companies and external parties—for example: employees using remote computers; partner companies; customers; and so on. The old way of securing information behind an organization’s perimeter wall prevented this type of data exchange in a secure manner. However, the ideas forwarded by the Jericho Forum are also applicable to cloud computing. The idea of creating, essentially, de-centralized perimeters, where the perimeters are created by the data object itself, allows the security to move with the data, as opposed to retaining the data within a secured and static wall. This simple but revolutionary change in mindset of how to secure data is the ground stone of securing information within a cloud and will be the basis of this discussion on secur-ing data in the cloud.

HOMO SAPIENS AND DIGITAL INFORMATION

Cloud computing offers individuals and organizations a much more fluid and open way of communicating information. This is a very positive move forward in communication technology, because it provides a more accurate mimic of the natural way that information is communicated between individuals and groups of human beings. Human discourse, including the written word, is, by nature, an open transaction: I have this snippet of information and I will tell you, verbally or in written form, what that information is. If the information is sensitive, it may be whispered, or, if written on paper, passed only to those allowed to read it. The result is that human-to-human information communication will result in a very fluid discourse. Cloud computing is a platform for creating the digital equivalent of this fluid, human-to-

human information flow, which is something that internal computing networks have never quite achieved. In this respect, cloud computing should be seen as a revolutionary move forward in the use of technology to enhance human communications.

Although outside of the remit of this chapter, it is worthwhile for any person looking into developing systems for digital communications to attempt to understand the underlying social evolutionary and anthropological reasons behind the way that human beings communicate. This can give some insight into digital versions of communication models, because most fit with the natural way that humans communicate information. Security system design, in particular, can benefit from this underlying knowledge, because this type of system is built both to thwart deceptive attempts to intercept communication and to enhance and enable safe and trusted communications: Bear in mind that both deception and trust are intrinsic evolutionary traits, which human beings have developed to help them to successfully communicate.

CLOUD COMPUTING AND DATA SECURITY RISK

The cloud computing model opens up old and new data security risks. By its very definition, Cloud computing is a development that is meant to allow more open accessibility and easier and improved data sharing. Data are uploaded into a cloud and stored in a data center, for access by users from that data center; or in a more fully cloud-based model, the data themselves are created in the cloud and stored and accessed from the cloud (again via a data center). The most obvious risk in this scenario is that associated with the storage of that data. A user uploading or creating cloud-based data include those data that are stored and maintained by a third-party cloud provider such as Google, Amazon, Microsoft, and so on. This action has several risks associated with it: Firstly, it is necessary to protect the data during upload into the data center to ensure that the data do not get hijacked on the way into the database. Secondly, it is necessary to the stores the data in the data center to ensure that they are encrypted at all times. Thirdly, and perhaps less obvious, the access to those data need to be controlled; this control should also be applied to the hosting company, including the administrators of the data center. In addition, an area often forgotten in the application of security to a data resource is the protection of that resource during its use—that is, during a collaboration step as part of a document workflow process. Other issues that complicate the area of

hosted data include ensuring that the various data security acts and rules are adhered to; this becomes particularly complicated when you consider the cross border implications of cloud computing and the hosting of data in a country other than that originating the data.

Data security risks are compounded by the open nature of cloud computing. Access control becomes a much more fundamental issue in cloud-based systems because of the accessibility of the data therein. If you use a system that provides improved accessibility and opens up the platform to multi-node access, then you need to take into account the risks associated with this improvement. One way this can be done is by adding an element of control, in the form of access control, to afford a degree of risk mitigation. Information-centric access control (as opposed to access control lists) can help to balance improved accessibility with risk, by associating access rules with different data objects within an open and accessible platform, without losing the inherent usability of that platform.

A further area of risk associated not only with cloud computing, but also with traditional network computing, is the use of content after access. The risk is potentially higher in a cloud network, for the simple reason that the information is outside of your corporate walls; for example, a user printing off a sensitive document within an office of a company is more likely to think twice about doing so if her colleagues can see her actions than if she prints out that document in the privacy of her own home or within the anonymity of an Internet cafe.

Recent research by Gartner, on the top 10 “disruptive technologies,” outlined these as being key transformation technologies for the industry. The technologies included Cloud and Web ecosystems as well as virtualization and social software. Gartner predict that by 2010, Mashups, used to create composite applications to share and combine internal and external data sources, will be used as the dominant mode of creation for enterprise composite, applications. In addition to this, corporate blogs are being heavily touted as a means of disseminating and collaborating on information: Technorati research for the 2008 State of the Blogosphere report puts corporate blogging at 12% of the total blogs and a Universal McCann study shows that consumers think more positively about companies that have blogs; Statistics suggest that this media will become more heavily used within a corporate context.

A recent survey by Citrix which polled UK IT directors and managers showed that two-thirds of UK companies were computing in the cloud. Of those polled, one-third said they thought there were security risks and 22% said they had concerns over the control of their data in the cloud. However, coupled with these improvements in computing capabilities come new technical challenges and hurdles, in particular in the area of security because of the highly complex manner in which security applications need to operate and inter-operate. The Internet and mobile devices have effectively opened up new points at which data can leak; and as new methods of communicating emerge, they will open up even more potential for information loss.

The development of Web 2.0 technologies has created a new and more dynamic method of communicating information; blogs, social networking sites, Web conferencing, wikis, podcasts and ultimately cloud computing itself offer new and novel methods of getting information from a to b; unfortunately, this can also often be via x, y, and z.

Since cloud computing has come to the fore, there has been a general consensus that data within this domain are more at risk. While on the one hand these new technologies are being met with a degree of enthusiasm, there is also an equal degree of fear in terms of securing data and risk management. Compliance with data security directives and acts still needs to be met, no matter what platform for communication is being used. The lack of security and privacy within a cloud computing environment is hotly debated over whether this problem is perceived or real. However, reports by IT industry analysts suggest that this is a real problem and must be overcome to allow full utilization of cloud computing. A recent report by IDC which surveyed 244 respondents identified security as the main challenge for cloud computing, with 74.6% of the vote stating this as a stumbling block to the uptake of the technology. Reports by Gartner and Gigacom, specifically on cloud security, also confirms this.

With new technologies come new exploits; and cloud computing, being by definition a more open way of performing information technology operations, will bring security challenges that will leave Internet-based data vulnerable. As previously mentioned, mashups have been identified as being a security concern. Data-centric mashups—that is, those that are used to perform business processes around data creation and dissemination—by their very nature, can be used to hijack data, leaking

sensitive information and/or affecting integrity of that data. An InfoWorld article summed up this fear: “. . . megabytes of valuable customer or financial data could be compromised in just a few seconds if a rogue data-centric mashup is created”.

Cloud computing, more than any other form of digital communication technology, has created a need to ensure that protection is applied at the inception of the information, in a content centric manner, ensuring that a security policy becomes an integral part of that data throughout its life cycle.

Encryption is a vital component of the protection policy, but further controls over the access of that data and on the use of the data must be met. In the case of mashups the controlling of access to data resources, can help alleviate the security concerns by ensuring that mashup access is authenticated. Linking security policies, as applied to the use of content, to the access control method offer a way of continuing protection of data, post access and throughout the life cycle; this type of data security philosophy must be incorporated into the use of cloud computing to alleviate security risks.

We can thus conclude that the risk profile of an organization, or individual, using the cloud to store, manage, distribute, and share its information has several layers. Each layer can be seen as a separate, but tied, level of risk that can be viewed independently, but these risks should be approached as a whole, to make sure that areas constituting a “weakest link” do not end up built into the system.

CLOUD COMPUTING AND IDENTITY

Digital identity holds the key to flexible data security within a cloud environment. This is a bold statement, but nonetheless appears to be the method of choice by a number of industry leaders. However, as well as being a perceived panacea for the ills of data security, it is also one of the most difficult technological methods to get right. Identity, of all the components of information technology, is perhaps the most closest to the heart of the individual. After all, our identity is our most personal possession and a digital identity represents who we are and how we interact with others on-line. The current state of the art in digital identity, in particular with reference to cloud identities, is a work in progress, which by the time you are reading this should hopefully be entering more maturity. However, going back to my opening statement, digital identity can be used to form the basis of data security, not only in the cloud but also at the local network level too. To expand on this

somewhat, we need to look at the link between access, identity, and risk. These three variables can become inherently connected when applied to the security of data, because access and risk are directly proportional: As access increases, so then risk to the security of the data increases. Access controlled by identifying the actor attempting the access is the most logical manner of performing this operation. Ultimately, digital identity holds the key to securing data, if that digital identity can be programmatically linked to security policies controlling the post-access usage of data.

The developments seen in the area of a cloud-based digital identity layer have been focused on creating a “user-centric” identity mechanism. User-centric identity, as opposed to enterprise-centric identity, is a laudable design goal for something that is ultimately owned by the user. However, the Internet tenet of “I am who I say I am” cannot support the security requirements of a data protection methodology based on digital identity, therefore digital identity, in the context of a security system backbone, must be a verified identity by some trusted third party: It is worth noting that even if your identity is verified by a trusted host, it can still be under an individual’s management and control.

With this proposed use of identity, on the type of scale and openness as expected in a cloud computing context, we must also consider the privacy implications of that individual’s identity. A digital identity can carry with it many identifiers about an individual that make identity theft a problem, but identity should also be kept private for the simple reason of respect. However, privacy is a very personal choice and, as such, the ability to remain private within a cloud, should be, at the very least, an option.

Identity, Reputation, and Trust

One of the other less considered areas of digital identity is the link between the identity and the reputation of the individual identity owner. Reputation is a real-world commodity that is a basic requirement of human-to-human relationships: Our basic societal communication structure is built upon the idea of reputation and trust. Reputation and its counter value, trust, is easily transferable to a digital realm: eBay, for example, having partly built a successful business model on the strength of a ratings system, builds up the reputation of its buyers and sellers through successful (or unsuccessful) transactions. These types of reputation systems can be extremely useful when used with a digital

identity. They can be used to associate varying levels of trust with that identity, which in turn can be used to define the level (granular variations) of security policy applied to data resources that the individual wishes to access.

Identity for Identity's Sake

An aspect of identity that again is part of our real world and needs to be mimicked in the digital world is that of “multiple identities,” because in the cloud you may find that you need a different “identity” or set of identifiers to access resources or perform different tasks.

If we are to go down the path of using digital identity as the backbone of a cloud-based data security system, then we must make sure that the identity layer of cloud computing is able to handle the very flexible requirements of data security. These include the need for free flow of information, dynamic policies, data-centric security, and privacy. User-centric identity systems, based on dynamic claims (individual identifying artifacts), do seem to have the pre-requisites for this, and the next part of this chapter will look more closely at the currently available cloud-based identities including those based on claims.

Cloud Identity: User-Centric and Open-Identity Systems

As the use of the Internet and cloud computing increases, the risks associated with identifying yourself, via this medium, have also increased. Identity fraud and theft are a real threat to the uptake and acceptance of cloud computing; and as already stated, a robust digital identity can be the backbone of data security in the cloud.

Internet identities such as information cards were originally designed to overcome the problem of “password fatigue,” which is an increasing problem for users needing to remember multiple log-on credentials for Web site access. Similarly, OpenID was developed for the purpose of an easier logon into multiple Web sites, negating the need to remember username/logon credentials. Information cards differ from OpenID in a fundamental manner in that information cards have an architecture built on the principle of “claims,” claims being pieces of information that can be used to identify the card holder. At this juncture it is worth pointing out that, although OpenID can use claims, the architecture behind OpenID makes this use of claims less flexible—and, more importantly, less dynamic in nature—than those offered by information cards.

One of the most powerful aspects of these Internet identities is the push toward a common framework of operation. This type of framework can make managing such identities simpler and provide more extensible

cross-platform and cross-application support, improving scalability and ultimately security. The IT industry is making great strides in this area by coming together in a cooperative way to work toward such a common framework. The work toward this has come about as a result of the large number of prior identity management systems built for purpose, but not for interoperability.

The Philosophy of User-Centric Identity

Digital identities are a still evolving mechanism for identifying an individual, particularly within a cloud environment; and, as such, the philosophy behind the idea is also still being formed. However, one area that is being recognized as a basic component of an identity is that of identity ownership being placed upon the individual (user-centric). Placing ownership with an individual then sets in place a protocol around the use of the identity. The industry is slanting heavily toward allowing users to consent and control how their identity (and the individual identifiers making up the identity, the claims) is used. This reversal of ownership away from centrally managed identity platforms (enterprise-centric) has many advantages. This includes the potential to improve the privacy aspects of a digital identity, by giving an individual the ability to apply permission policies based on their identity and to control which aspects of that identity are divulged. To this end, the term “user-centric” has come to mean that an identity may be controllable by the end user, to the extent that the user can then decide what information is given to the party relying on the identity.

User-Centric but Manageable

One area that often gets confused by the use of the term “user-centric” is the management of users’ identities. Although the term “user-centric” implies that the identity is under the control and management of the end user (or that the identity “flows” from the user to the relying application), this is true only within the context of the use of the identity. For example, in the case of many user-centric identities, the user can entirely create and manage them within their own desktop or cloud environment. However, within the context of data security, a personally managed identity may not carry enough assurance or weight of nonrepudiation to be used sensibly. In situations that require a degree of nonrepudiation and verification, where a user is who they say they are—that is, situations that require a digital identity to provide access control and security—user-centric identities can still be under user control and thus user-centric (the user choosing which identity and

which identity claims to send across a transaction path) but must be issued and managed by a trusted host able to verify the user (for example, the user's bank). This may seem like a security paradox, but it is actually a balanced way of using a digital identity to assign security policies and control while retaining a high measure of privacy and user choice.

What Is an Information Card?

Information cards permit a user to present to a Web site or other service (relying party) one or more claims, in the form of a software token, which may be used to uniquely identify that user. They can be used in place of user name/ passwords, digital certificates, and other identification systems, when user identity needs to be established to control access to a Web site or other resource, or to permit digital signing.

Information cards are part of an identity meta-system consisting of:

1. Identity providers (IdP), who provision and manage information cards, with specific claims, to users.
2. Users who own and utilize the cards to gain access to Web sites and other resources that support information cards.
3. An identity selector/service, which is a piece of software on the user's desktop or in the cloud that allows a user to select and manage their cards.
4. Relying parties. These are the applications, services, and so on, that can use an information card to authenticate a person and to then authorize an action such as logging onto a Web site, accessing a document, signing content, and so on.

Each information card is associated with a set of claims which can be used to identify the user. These claims include identifiers such as name, email address, post code, and so on. Almost any information may be used as a claim, if supported by the identity provider/relying party; for example, a security clearance level could be used as a claim, as well as a method of assigning a security policy. Only the claim types are stored in cards issued by an identity provider; the claim values are stored by the provider, creating a more secure and privacy-rich system. One of the strengths of these claims is that they are dynamic and thus can be changed in real time: If linked to a security policy, they can provide a method of dynamic security policy application. As part of the security

process inherent in the use of the information card, the cards are backed by an authentication mechanism that the user must satisfy in order to use the card. This could be a password, possession of an X509 certificate, OpenID account, a Kerberos ticket, an out-of-band method, or possession of another information card, and so on.

One of the most positive aspects of an information card is the user-centric nature of the card. An information card IdP can be set up so that the end users themselves can self-issue a card, based on the required claims that they themselves input—the claims being validated if needed. Alternatively, the claims can be programmatically input by the IdP via a Web service or similar, allowing the end user to simply enter the information card site and download the card.

Using Information Cards to Protect Data

Information cards are built around a set of open standards devised by a consortium that includes Microsoft, IBM, Novell, and so on.

The original remit of the cards was to create a type of single sign on system for the Internet, to help users to move away from the need to remember multiple passwords. However, the information card system can be used in many more ways. Because an information card is a type of digital identity, it can be used in the same way that other digital identities can be used. For example, an information card can be used to digitally sign data and content and to control access to data and content. One of the more sophisticated uses of an information card is the advantage given to the cards by way of the claims system. Claims are the building blocks of the card and are dynamic in that they can be changed either manually or programmatically, and this change occurs in real time: As soon as the change is made, it can be reflected when the card is used, for example, by a subsequent change in the access or content usage policy of the resource requiring the information card. This feature can be used by applications that rely on the claims within an information card to perform a task (such as control access to a cloud-based data resource such as a document). A security policy could be applied to a data resource that will be enacted when a specific information card claim is presented to it: If this claim changes, the policy can subsequently change.

For example, a policy could be applied to a Google Apps document specifying that access is allowed for user A when they present their information card with claim “security clearance level 5 3” and that post access, this user will be able to view this document for 5 days and be

allowed to edit it. The same policy could also reflect a different security setting if the claim changes, say to a security clearance level 5 1; in this instance the user could be disallowed access or allowed access with very limited usage rights.

Weakness and Strengths of Information Cards

The dynamic nature of information cards is the strength of the system, but the weakness of information cards lies in the authentication. The current information card identity provisioning services on offer include Microsoft Geneva, Parity, Azigo, Higgins Project, Bandit, and Avoco Secure. Each offers varying levels of card authentication and are chosen from Username and password, Kerberos token, x509 digital certificate, and personal card. Each of these methods has drawbacks. For example, username and password is less secure and also not transparent. X509 digital certificates can be difficult for less technical users to install and use. However, new developments in information card authentication are on the industry roadmap, including Live ID, OpenID, and out-of-band (also referred to as “out-of-wallet”). This latter option offers much higher levels of authentication and thus security, but does have drawbacks in terms of transparency. However, a full gamut of authentication offerings can only improve the security of the information card system. Going forward, it is hoped that GPS location authentication can also be added to the list of authentication choices to control access to resources. Based on geo-graphic location of the person attempting access, this could become a particularly important feature for cloud-based data, which can potentially be accessed anywhere in the world but may be constrained by compliance with industry legal requirements.

Cross-Border Aspects of Information Cards

Cloud computing brings with it certain problems that are specific to a widely distributed computing system. These problems stem from the cross-border nature of cloud computing and the types of compliance issues arising out of such a situation. An identity meta-system based on interoperable standards of issuance and authentication, such as an information card, is an absolute requirement for digital identity to be successfully used across borders. Information cards can potentially provide such a framework, because they are based on the idea of an identity metasystem, the goal of which is to connect individual identity systems resulting in cards issued by a given host being compatible across the entire system. The Oasis Foundation, which is nonprofit

organization that is striving to establish open standards for IT, has formed a working committee to “enable the use of information cards to universally manage personal digital identities.”

In addition, the Information Card Foundation, headed up by some of the largest IT companies in the world, has a mission statement that includes: to “provide guidance and support for projects advancing information card infrastructure on the widest possible range of platforms, including freely available open source implementations”.

The idea of using information cards as a cross-border, interoperable system was presented in March 2009 as an idea at the The European e-ID interoperability Conference: Current Perspective and Initiatives from around Europe in Government and Business.

The use of information cards as a method of digitally identifying an individual within the cloud (as well as on the desktop) will gain ground, as its usage model extends with increased support for information cards, from relying parties and as usability through the use of cloud-based selectors becomes more mainstream.

THE CLOUD, DIGITAL IDENTITY, AND DATA SECURITY

When we look at protecting data, irrespective of whether that protection is achieved on a desktop, on a network drive, on a remote laptop, or in a cloud, we need to remember certain things about data and human beings. Data are most often information that needs to be used; it may be unfinished and require to be passed through several hands for collaboration for completion, or it could be a finished document needing to be sent onto many organizations and then passed through multiple users to inform. It may also be part of an elaborate workflow, across multiple document management systems, working on platforms that cross the desktop and cloud domain. Ultimately, that information may end up in storage in a data center on a third-party server within the cloud, but even then it is likely to be re-used from time to time. This means that the idea of “static” data is not entirely true and it is much better (certainly in terms of securing that data) to think of it as highly fluid, but intermittently static.

What are the implications of this? If we think of data as being an “entity” that is not restricted by network barriers and that is opened by multiple users in a distributed manner, then we should start to envision that a successful protection model will be based on that protection policy being an intrinsic part of that entity. If the protection becomes

inherent in the data object, in much the same way that perhaps a font type is inherent in a document (although in the case of security in a much more persistent manner), then it is much less important where that data resides. However, how this is achieved programmatically is a little trickier, particularly in terms of interoperability across hybrid cloud systems.

One of the other aspects of data security we need to assess before embarking on creating a security model for data in the cloud is the levels of need; that is, how secure do you want that data to be? The levels of security of any data object should be thought of as concentric layers of increasingly pervasive security, which I have broken down here into their component parts to show the increasing granularity of this pervasiveness:

- Level 1: Transmission of the file using encryption protocols
- Level 2: Access control to the file itself, but without encryption of the content
- Level 3: Access control (including encryption of the content of a data object)
- Level 4: Access control (including encryption of the content of a data object) also including rights management options (for example, no copying content, no printing content, date restrictions, etc.)

Other options that can be included in securing data could also include watermarking or red-acting of content, but these would come under level 4 above as additional options.

You can see from the increasing granularity laid out here that security, especially within highly distributed environments like cloud computing, is not an on/off scenario. This way of thinking about security is crucial to the successful creation of cloud security models. Content level application of data security gives you the opportunity to ensure that all four levels can be met by a single architecture, instead of multiple models of operation which can cause interoperability issues and, as previously mentioned, can add additional elements of human error, leading to loss of security.

The current state of cloud computing provides us with a number of cloud deployment models, namely, public (cloud infrastructure that is open for public use, for example, Google App engine is deployed in a public cloud), private (privately available clouds on a private network used by an individual company; for example, IBM provides private clouds to

customers, particularly concerned by the security issues surrounding public cloud deployments), managed (clouds offered by a third-party hosting company who look after the implementation and operational aspects of cloud computing for an organization), and hybrid (a mix of both public and private cloud implementations). It is highly likely, especially in the early years of cloud computing, that organizations will use a mixture of several, if not all, of these different models. With this in mind, to allow an organization to deal with securing data within any of these types of systems means that the issues of interoperability, cross-cloud support, minimisation of human error, and persistence of security are crucial. The fluid movement of data through and between these clouds is an integral part of the cloud philosophy, and any data security added into this mix must not adversely encumber this movement. This requires that you look at that data as a separate entity with respect to the underlying system that it moves through and resides within. If you do not view the data as a free-moving object, you will build a data security model that is not built to suit the data, but instead is built for the specific system surrounding that data. In a cloud-type system, the end result is likely to be only suitable for static data (something that we have already described as not truly existing) which will not be able to transcend that original system without potentially having to be re-engineered to do so, or at the very least having additional features and functions tagged onto the original specification. This type of software engineering results in interoperability issues and an increased chance of bugs occurring, because of feature adjuncts being added as an after thought, as opposed to being built into the original working architecture of the software. In addition, what can occur with security software development, which uses a non-extensible approach to software design, is that security holes end up being inadvertently built into the software, which may be very difficult to test for as the software feature bloat increases. With this in mind, the way forward in creating data security software models for a cloud computing environment must be done from scratch. We must leave the previous world of encrypted containers behind us and open up a new paradigm of fluidic protection mechanisms based on content-centric ideologies. Only through this approach will we hope to achieve transcendence of security across the varying types of cloud architectures.

CONTENT LEVEL SECURITY—PROS AND CONS

Much of the substance of this chapter has described a new way of thinking about securing data, so that data within a cloud can remain fluid, accessible on multiple nodes and yet remain protected throughout its life cycle. The basis of this new security model has been described as “content or information-centric.” What this means in reality is that the content that makes up any given data object (for example, a Word document) is protected, as opposed to the file—that is, the carrier of that information being protected. This subtle difference in approach gives us a major advantage in terms of granularity and choice of protection level, as well as persistence of protection. We will take a Word document as our example here to outline the main pros and cons of this type of security approach.

Imagine that I have just prepared a merger and acquisition (M&A) draft document using an on-line document authoring application, such as Google Apps. I need to share this document with persons within my own company, across several departments, as well as with an external lawyer and with the third-party company to be acquired. In addition, I want to make sure that certain sections are only visible to certain of these parties and that they cannot change any item or copy the content (I don’t want some of the sensitive clauses to be placed on an ex-employee’s blog page, or leaked to the press to affect share prices). I also want to audit the access and use of the document and to limit the time that these people can read the draft of this document, because I want to close this acquisition within 2 weeks. Thereafter I need to publish the finished M&A document with new access rights and restrictions to reflect its new status. I am also acutely aware that the data center that is holding this sensitive document is being hosted by a cloud vendor, and I definitely do not want the administrator of that data center to see this transaction.

How can I achieve this? I could create a shared on-line document portal that controls access to the document using a password login and set up user accounts for those persons I wish to share the document with. The main problem with this type of container-based security is that it relies on the user not sharing their password. In addition, once access is gained, the user can use the document without restriction; for example, copy the document content to their blog page, email the document to others, or download the document to a local computer and share it with anyone they wish to, across their network. In addition, the document is

potentially accessible by the cloud vendor themselves. To prevent any of these unauthorized actions, I will need to control the document content itself and improve on the access control measures, because password access is far too insecure. This is where a content-centric approach delivers persistent and pervasive security. Content-centric security, which is also digital identity led (i.e., the identity used to access the content), also dictates the security policy applied to that content and will allow me to control who accesses my M&A draft, because at the time of protecting the draft I will decide who can access it and how access is controlled. This brings us back to the section on information cards. I could protect the draft document by assigning access to persons who hold a managed information card, which contains certain claims—for example, specific email addresses, a security clearance level (set by a specified identity provider), or a specified company number, and so on.

Only those persons could then access the document; and because the claims are managed by an identity provider (perhaps my own company), the claims can also be dynamically changed and, as such, if I need to revoke access to the document, I can arrange for the claims to change in line with this, revoke the information card of that user, or alternatively change the security applied to the document. Once access is gained, security policies that control what part of the document can be seen, by which person and what they can do with the content, will be applied; because the access is based on an individual identity, individual content controls can be applied and so some users can be given stronger rights restrictions than others. Importantly, even though the document is held on third-party servers, in the cloud it can't be accessed by even the system administrator of that server, because the access is controlled at the content level and is not dependent on the access to the database holding the data.

You can easily see the advantages that are conferred on data protected at the content level: greater control, more focused access control, increased granular protection over content, and assurance within a cloud-hosted system. But what, if any, disadvantages come with this type of methodology?

Container security is a much simpler way of securing data. Within a cloud computing environment you have the storage and transfer of data, both of which can be easily accommodated in terms of security by using encryption protocols already built for the purpose. It is fairly simple to apply database encryption, because it is applied natively to the data and

decrypted, on-the-fly, when there is a query on that data. Similarly, transfer of the data between application and database, or human-to-human transfer, can protect the data as an encrypted package, decrypted when access is granted. Content-centric security measures need to be compatible with both database security and secure transfer of data within a cloud environment. Protecting the content of our Word document needs to be done in such a manner that it does not impact the storage of that data. This may be problematic, especially across different storage types and in use with query engines, which is particularly pertinent with the use of dynamic data updating, as required by modern data storage operations. One of the other aspects of cloud computing data storage that can complicate the area of data security is the use of redundant storage in more than one location. However, at this juncture it is worth noting that this same issue causes more problems for a container approach than for a content-centric approach, in terms of synchronicity between databases. The current state of research, with respect to the protection of data within a cloud computing environment, is focused on the protection of data within the data centers hosting the cloud: The problems therein are compounded by the highly distributed nature of the cloud and the use of multi-center storage and replication of data. Content-centric security needs to overcome these same problems and also needs to retain protection of data within the structure of the database itself; this, however, is a programmatic problem.

Data Privacy & Security in Cloud Computing

Cloud technology has given opportunities to many businesses to showcase their potential in the business world. SMEs are not only getting an opportunity to grow, they are also taking their business operations to the next level. *Cloud technology* has opened a door for small & medium scale companies to acquire market share by entering the yard of bigger players. As the business requirements have become on-demand and need-based, it gave many companies a significant edge and allow them to complete in a much larger business space.

Cloud technology provides various advantages. Starting from data management, data storage, 0% downtime, CRM management, resource optimization to entire business automation. It also reduces a high amount of investment and saves a lot of time.

At the same time, cloud computing has raised multiple eyebrows with IT management, especially when it comes to *data security in the cloud computing*. *Data security and privacy protection are two major factors. These two factors are becoming more important for the future development of cloud computing technology in business, industry, and government.* While addressing this fear, Google claimed that data stored in the cloud are much safer.

If someone asks me what cloud computing is, I try not to get bogged down with definitions. I tell them that, simply put, cloud computing is a better way to run your business.

What are the Challenges?

Data Replication

Every business faces this challenge. Snapshots and data backups are taken on a daily basis. They automatically stored in the cloud. Are you aware where they have been stored and who can see and access them? Can you identify and control unauthorised copying of your data?

Data Loss

Data loss can be a disaster for any business. Virtual data can be easily lost or exposed as it moves between VMs or in the cloud. Are you sure that authorised users are accessing your data within predefined policies? Do you have the authority to block any user who is violating data use policies?

New Class of Users

Cloud computing need cooperation between security, storage, application, and security admins. They all manage your sensitive business data. With more number of users, the risk also increases. If one admin went wrong, entire data in the system will be at risk.

Insecure APIs

Application Programming Interfaces (API) allow users to customize their cloud computing practices. APIs can be a threat to *cloud security* because of their nature. APIs give developers the tools to build solutions to integrate their applications with other software. **The**

vulnerability of an API depends on the communication that takes place between applications. While this can help developers and businesses, they also issue serious security concerns.

Internal Threat

Never keep this point out of your mind. You may be thinking data is safe inside. But this is one of the biggest challenge company's face. Employees can use their access to an organisation's cloud-based services to misuse or access information related to finance, customer details etc.

How to Protect your Data?

You can protect your business data in the cloud from unauthorised access. All you need is a sharp eye and an extra effort. Here are few practical tips to keep your cloud data safe and secure.



Always keep backup locally

When it comes to business data, you have to be extra conscious. Always have a backup for your data. It is always good to create hard copies of your business data and keep it with yourself so that you can have access

them even if you lost the original one. You can use any cloud storage solutions to store your data. You can set up a cloud account & can keep the backup copies. You have another option of keeping the backup data in an external storage device also like a hard disk or a thumb drive. This will allow you to access the information even if without the internet.

Don't store sensitive data

Technology is changing. Businesses are also changing as per the technology. Data is playing an important role in businesses today. So, data privacy is one of the primary aspects of any business. But if something is there on the internet, it is hard to trust it is safe. So, one should avoid storing the most sensitive files or information in the cloud. Identity theft is on rising and you can't take any risk. You should keep those files in cloud platform which you access frequently and should avoid putting information related to financial details, competitor details, client details, contact details like phone number/address etc. If you are keeping these files, make sure you encrypt them before uploading.

Data encryption

One of the best ways to protect your data while using cloud storage is to do data encryption. This is the best form of security because you need decryption before accessing the data. This will protect data against service providers and users also. To make it more protected, you can also ensure cloud encryption during uploading and downloading phases. But, this will make data sharing and sync in the cloud platform little slow.

Encrypted cloud service

There are few *cloud services* which provide local encryption and decryption of your files and information inside that other than storage and backup. This means the service takes care of both encrypting your files and storing them safely in the cloud. This will ensure that no one including the service provider or the administrators can have the access to your data files. There are many free versions and also trial versions available in the market. You can use them to learn how it works and later can upgrade to enjoy more space.

Using password

The first thing which can be done is to put strong password which can stand a hacking. You can take the help of internet to learn how to create a strong password. It is very important to change your password frequently and never use the same password for all the accounts or folders. You can opt for 2-step verification for login if your cloud service offers that option. Google drive use 2 phase log in option, consist of password & code sent to the registered number. This added security will make your data much safer.

Keep an eye on what you do online

The security of your cloud data largely depends on your online behaviour. While using a public computer, never save your password, and always ensure that you logged out properly. Another biggest concern is accessing cloud data in unsecured or open Wi-Fi hotspots. Such connections are unencrypted, hackers can target your data easily. Never save your password in any of the public forum or social media. Change Wi-Fi passwords frequently.

Anti-virus is a must

Sometimes the weakest link happens to be the computer or device you use for cloud data access. You need to put proper protection in your system/device. It will help in securing your business data. If you expose yourself to bugs and viruses, hackers can access your system easily. You need to choose a very effective and robust anti-virus system for your system, which will protect all the files and information inside that. If your system isn't well protected, and if the system is not encrypted and secured from bugs, hackers can get hold of your information.

Read your user agreement

If you are new to the world of cloud computing and not sure what cloud storage to choose or how it really work, you have to read the user agreement of the service you are going to sign up for. Initially, it will be difficult to understand and at times it will test your patience, but you need to face this. User agreements always carry essential information which can help you understand things in detail.

Access limitation

Give access to those users who really need. **Internal users and third party vendors should only get access to those files which will help them to do their jobs.** Use encryption keys if required. Make sure to evaluate the users and vendors regularly and add/remove users as per the requirement.

Platform, control & service monitoring

Platform, control & services monitoring is usually performed as a dashboard interface and makes it possible to identify the operational status of the platform being monitored at any time. Each operational element which is monitored provides an operational status indicator. This helps in determining which elements are performing as per the established standards. By identifying such problems, you can take defensive actions to prevent loss of data or service.

Continuous system updating

Cloud data security is enhanced with regular patching and upgrading of systems and application software in the cloud platform. New patches, updates, and service packs for the operating system are required to maintain high-end security levels and support new versions of installed products. You have to be committed enough to identify the market trends and new software versions and communicate gaps in security that can appear in installed systems and applications.

Legal & regulatory challenges

Cloud services can give you the best solutions for your business related problems when you are assured that your & your customers data are private and secure. This should be the primary focus for cloud service providers. There are many legal & regulatory challenges which needs to be addressed when data moves from one country to another.

Multinational Framework on privacy and security :

Uncertainty about the legal and regulatory obligations related to data will increase with the increase of the data in the cloud platform. To ensure every business and country get full advantage of cloud computing, different countries have to cooperate to develop a multinational framework on data privacy and security in the cloud. As

cloud computing evolves, and data flows from one country to another. For example, data has been created in India using a software hosted in UK & stored in US with users based in Australia. Cloud provider needs to coordinate this entire process to make sure the data flow is going smooth & safe.

Rules on Cross-border data transfers :

To enhance the efficiency and security of *cloud solutions* and deliver quick results, cloud service providers must be able to operate datacentres in multiple locations and transfer data freely between them. Smooth data flow allows cloud providers to optimize their service and deliver the best business solutions. However, restrictions on cross-border data transfers can create uncertainty if the rules or the legal framework are not followed.

Conflicting legal obligations :

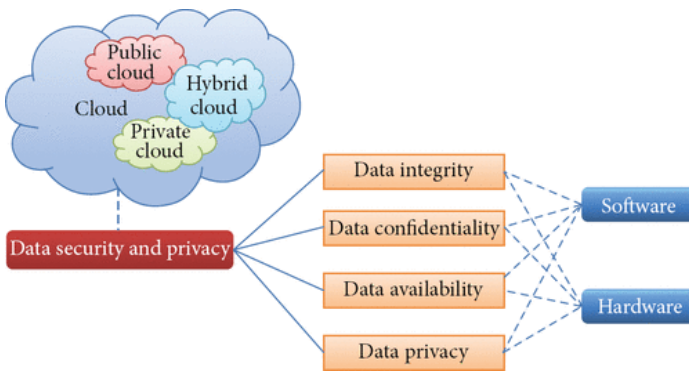
Different governments have different policies when it comes to data flow in their country. Cloud providers will be in legal trouble if they won't follow the predefined cyber laws. Divergent rules on privacy, data retention, law enforcement access and other issues can lead to ambiguity. **For example, one country might have certain rules when it comes to cloud data storage, which might be in direct conflict with another country or a particular service provider.**

In order to protect data in the cloud platform, you need to keep all these above things in mind.

Cloud computing is one of the most promising technology for the next generation of IT applications. The primary concern toward the accelerated growth of cloud services is data security and privacy issues. The main goal for any company is to reduce data storage and cost associated with it. As we all know data is playing a bigger role in taking business decisions, no company will deploy all their business data into the cloud unless they trust it completely. **There are many techniques which have been introduced by IT researchers for data protection and to achieve the highest level of data security.** However, there are still certain gaps to be filled by making these methods more effective. More awareness is required in the area of cloud computing to make it acceptable.

Cloud computing purely targets on cost-effective solutions and is a significant promoter of the modern digital economy by enabling leading companies to innovate, operate and conduct business more promptly and efficiently. However, the cloud is more than just delivering cost-effective solutions.

If you expect your business to grow, you are short on capital, or you don't have technology expertise, **cloud computing** could be the solution. It can add real value and can take your small business to the enterprise level.



CLOUD CONTRACTS: 5 MUST-HAVE ELEMENTS

1. Clearly state your data egress terms and conditions

When you leave a cloud provider, you want your data back. And, you want it in a readable format. During the signing of a new contract, nobody wants to discuss breaking the agreement, so this is often overlooked or swept under the carpet. The time to negotiate your potential early exit is before you ink the contract. Many cloud providers charge a small fortune for returning your data. Depending on the size, this could be a significant dollar amount. These data egress terms should also include commitments from the vendor to assist in the extraction and preformatting of your data into a useable state. Expect to pay a fee for this but negotiate the amount in advance to avoid sticker shock. Document all these services and charges up front.

2. Early termination fees

All contracts have these. It protects the cloud provider and is a necessary part of the contract. That said, you can and should ensure the penalties are reasonable, or eliminated if there is a lapse in service level.

agreements. No cloud provider wants to lose a paying customer. Most boilerplate cloud contracts have hefty penalty fees to discourage you from leaving. Times change, and technology changes, so plan ahead in the event you need to move to another provider. In the event of a service level agreement default, I would push for the elimination of any fees, and the possibility of penalties paid by the provider.

3. Security and audits

This needs to be clearly stated in the contract that you have the right to perform periodic audits of the cloud provider and their operations. Your contract should state detailed audits of the datacenters, what tests will be performed, what tools you plan to use, and other items important to your business. On top of all that, certifications such as SAS70, PCI, and others relevant to the service should be valid and current. It is highly advisable to include your CISO in this item.

4. Shop around

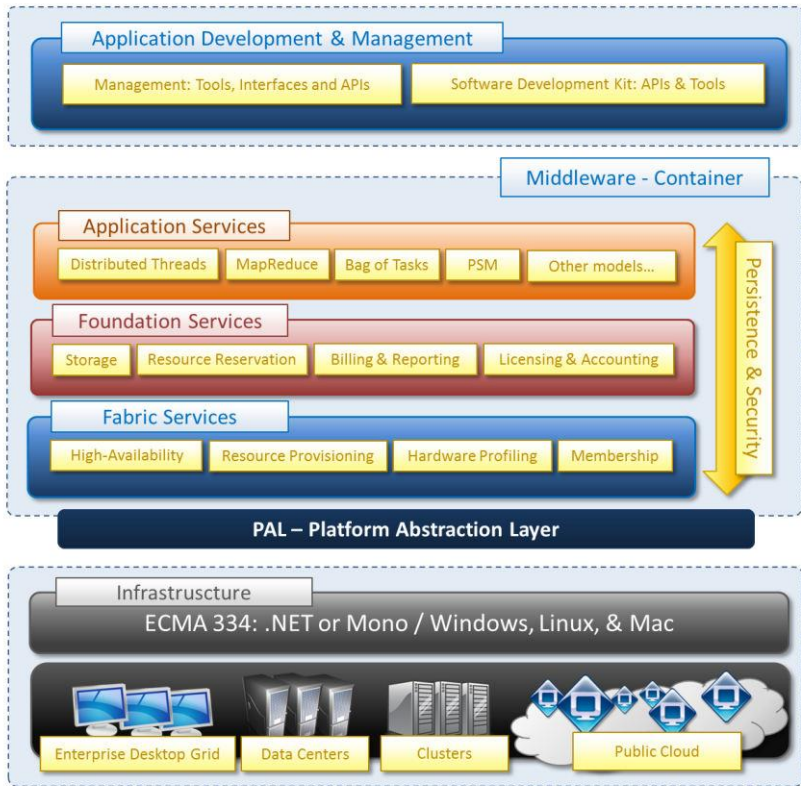
Nothing will get a vendor to the table with a sharpened pencil faster than knowing you have three other offers in your pocket. It is unethical to reveal competitors' prices; however, nothing prevents you from admitting you have other options to try and obtain the best price possible. This falls under Negotiating 101 but is sometimes skipped when discussing with a top-tier provider with a reputation to match.

5. Negotiate banded pricing

Many vendors will offer very attractive pricing to secure you as a client based on your current needs. That's great, but when you need to increase user counts or resources, you may be hit with sticker shock regarding the price. Best to negotiate a "banded" pricing program up front with the cloud provider that stipulates the cost for adding new employees and resources over and above initial commitment levels. This price protection can amount to tremendous savings over the long run.

CASE STUDIES

Aneka Architecture



Aneka is a platform and a framework for developing distributed applications on the Cloud. It harnesses the spare CPU cycles of a heterogeneous network of desktop PCs and servers or datacenters on demand. Aneka provides developers with a rich set of APIs for transparently exploiting such resources and expressing the business logic of applications by using the preferred programming abstractions. System administrators can leverage on a collection of tools to monitor and control the deployed infrastructure. This can be a public cloud available to anyone through the Internet, or a private cloud constituted by a set of nodes with restricted access.

The Aneka based computing cloud is a collection of physical and virtualized resources connected through a network, which are either the Internet or a private intranet. Each of these resources hosts an instance of the Aneka Container representing the runtime environment where the distributed applications are executed. The container provides the basic management features of the single node and leverages all the other operations on the services that it is hosting. The services are broken up into fabric, foundation, and execution services. Fabric services directly interact with the node through the Platform Abstraction Layer (PAL) and perform hardware profiling and dynamic resource provisioning. Foundation services identify the core system of the Aneka middleware, providing a set of basic features to enable Aneka containers to perform specialized and specific sets of tasks. Execution services directly deal with the scheduling and execution of applications in the Cloud.

One of the key features of Aneka is the ability of providing different ways for expressing distributed applications by offering different programming models; execution services are mostly concerned with providing the middleware with an implementation for these models. Additional services such as persistence and security are transversal to the entire stack of services that are hosted by the Container. At the application level, a set of different components and tools are provided to: 1) simplify the development of applications (SDK); 2) porting existing applications to the Cloud; and 3) monitoring and managing the Aneka Cloud.

A common deployment of Aneka is presented at the side. An Aneka based Cloud is constituted by a set of interconnected resources that are dynamically modified according to the user needs by using resource virtualization or by harnessing the spare CPU cycles of desktop machines. If the deployment identifies a private Cloud all the resources are in house, for example within the enterprise. This deployment is extended by adding publicly available resources on demand or by interacting with other Aneka public clouds providing computing resources connected over the Internet.

COMETCLOUD

(a) The CometCloud architecture comprises three key layers: infrastructure/federation, autonomic management, and programming/interface. (b) The CometCloud coordination model is used to orchestrate different aspects of the federation.

