The RESERVOIR Model and Architecture for Open Federated Cloud Computing *

B. Rochwerger D. Breitgand E. Levy A. Galis K. Nagin
I. Llorente R. Montero Y. Wolfsthal E. Elmroth J. Caceres
M. Ben-Yehuda W. Emmerich F. Galán

Abstract

The emerging cloud computing paradigm is rapidly gaining momentum as an alternative to traditional IT. However, contemporary cloud computing offerings are mostly geared towards Web 2.0 style applications and only recently have begun to address the needs of enterprise-grade solutions, such as support for infrastructure-level SLAs.

To address the challenges and deficiencies in the current state of the art, we propose a modular, extensible cloud architecture with intrinsic support for Business Service Management (BSM) and federation of clouds. Our goal is to facilitate an open, service-based, on-line economy, where resources and services are transparently provisioned and managed across clouds on an on-demand basis at competitive costs with high quality of service.

We present the vision driving the RESERVOIR project - an architecture that allows providers of cloud infrastructure to dynamically partner with each other to create a seemingly infinite pool of IT resources while fully preserving the autonomy of technological and business management decisions. To this end, RESERVOIR leverages and extends the advantages of virtualization and embeds autonomous management into the infrastructure. At the same time, the RESERVOIR approach aims to achieve a very ambitious goal - creating a foundation for next-generation enterprise-grade cloud computing.

1 Introduction

In the Web 2.0 era, companies grow from inception to a massive scale at incredible rates. For example, MySpace acquired 20 million users in two years; YouTube reached the same number of users in just 16 months [1]. However, to leverage this potential rate of growth, companies must properly address critical business decisions related to their service delivery infrastructure.

The emerging Cloud Computing paradigm [2], as exemplified by the Amazon Elastic Compute Cloud (EC2), represents a promising conceptual foundation for hosting and deployment of web-based services while theoretically relieving service providers from the responsibility of provisioning the computational resources needed to support these services. Cloud computing offers multiple advantages: it allows individuals or companies with market domain expertise to build and run their Software as a Service (SaaS) company with minimal effort in software development and without managing any hardware operations. This helps reduce software complexity and costs, expedite time-to-market, and enhance accessibility of consumers.

With cloud computing, companies can lease infrastructure resources on-demand from a virtually unlimited pool. The "pay as you go" billing model applies charges for the actually used

*The research leading to these results is partially supported by the European Community's Seventh Framework Programme (FP7/2001-2013) under grant agreement n° 215605.

resources per unit time. This way, a business can optimize its IT investment and improve availability and scalability.

While cloud computing holds huge promise for the future of service computing, a number of inherent deficiencies in current offerings can be pointed out:

Inherently limited scalability of single-provider clouds: Although most infrastructure cloud providers today claim infinite scalability, in reality it is reasonable to assume that even the largest players may start facing scalability problems as Cloud Computing usage rate increases. In the long term, scalability problems may be expected to aggravate as cloud providers serve an increasing number of on-line services, each accessed by massive amounts of global users at all times.

Lack of interoperability among cloud providers: Contemporary cloud technologies have not been designed with interoperability in mind. This results in an inability to scale through business partnerships across clouds providers. In addition, it prevents small and medium cloud infrastructure providers from entering the cloud provisioning market. Overall, this stifles competition and locks consumers to a single vendor.

No built-in Business Service Management support: Business Service Management (BSM) is a management strategy that allows businesses to align their IT management with their high level business goals. The key aspect of BSM is Service Level Agreement (SLA) management. Current cloud computing solutions are not designed to support the BSM practices that are well established in the daily management of the enterprise IT departments. As a result, enterprises looking at transforming their IT operations to cloud-based technologies face a non-incremental and potentially disruptive step.

We argue that none of these problems, as well as other major problems such as security and availability, are not remediable by retrofitting existing architectures. On the contrary, these issues should be addressed by proper design of a cloud computing architecture from basic principles. In this paper, we propose a reference model and architecture that systematically address some of those deficiencies and serve as a potential foundation for delivering IT services as utilities over the Internet.

1.1 The RESERVOIR Approach

The RESERVOIR vision is to enable on-demand delivery of IT services at competitive costs, without requiring a large capital investment in infrastructure. Our model is inspired by a strong desire to liken the delivery of IT services to the delivery of common utilities. For example, a common scenario in the electric grid is for one facility to dynamically acquire electricity from a neighboring facility to meet a spike in demand. We deem that similarly to other industries, where no single provider serves all customers at all times, next-generation cloud computing infrastructure should support a model where multiple independent providers can cooperate seamlessly to maximize their benefit.

More specifically, we believe that to truly fulfill the promise of cloud computing, there should be technological capabilities to *federate* disparate data centers, including those owned by separate organizations. Only through federation and interoperability can infrastructure providers take advantage of their aggregated capabilities to provide a seemingly infinite service computing utility. Informally, we refer to the infrastructure that supports this paradigm as a *federated cloud*.

An additional important advantage offered by the federated cloud approach is that it democratizes the supply side of cloud computing and allows small and medium-sized businesses and new entrants to become cloud providers. This encourages competition and innovation.

As discussed above, one of the limiting factors in current cloud computing offerings is the lack of support for BSM, specifically for business-aligned SLA management. While specific cloud computing solutions can be enhanced with some aspects of BSM, the provisioning of complex services across a federated network of possibly disparate data centers is a difficult and yet unsolved problem. A service may be a composition of numerous distributed resources, including computing, storage, and network elements. Provisioning such a service consumes physical resources, but should not cause an SLA violation of any other running application with a probability larger than some predefined threshold. As SLAs serve as risk mitigation mechanisms, this threshold represents the risk that a cloud provider and the cloud customer are willing to accept.

With BSM, applications are properly dimensioned, and their non-functional characteristics (e.g., performance, availability, security, etc.), governed by SLAs, are ensured with optimal cost through continuous IT optimization. We argue that due to the immense scale envisioned by cloud computing, support for BSM should be maximally automated and embedded into the cloud infrastructure.

In the RESERVOIR model, each infrastructure provider is an autonomous business with its own business goals. A provider federates with other providers (i.e., other RESERVOIR sites) based on its own local preferences. The IT management at a specific RESERVOIR site is fully autonomous and governed by policies that are aligned with the site's business goals. To optimize this alignment, once initially provisioned, resources composing a service may be moved to other RESERVOIR sites based on economical, performance, or availability considerations. Our research addresses those issues and seeks to minimize the barriers to delivering services as utilities with guaranteed levels of service and proper risk mitigation.

Cloud computing is the latest incarnation of a general-purpose public computing utility. In recent years we have seen the many efforts towards computing as a utility - from grid computing [3], which made significant progress in the area of high performance scientific computing, to attempts at building enterprise-level utilities [4]. However, none of these attempts have materialized into a general purpose compute utility accessible by anyone, at any time, from anywhere.

What makes cloud computing different is that industry trends such as ubiquity of broadband networks, fast penetration of virtualization technology for x86-based servers [5], and the adoption of Software as a Service [6] are finally creating an opportunity and a need for a global computing utility. The reluctance to use on-line services as a replacement for traditional software is lessening – the success of companies such as salesforce.com proves that with the right set of security warranties and a competitive price, companies are willing to trust even their most valuable data – customer relations – to an on-line service provider. At the same time, virtualization has made it possible to decouple the functionality of a system as it is captured by the software stack (OS, middleware, application, configuration, and data) from the physical computational resources on which it executes [7]. This, in turn, enables a new model of on-line computing – instead of specially crafted on-line software, we can now think in terms of general purpose on-line virtual machines that can do anything. Finally, as virtualization enters the mainstream, the era of a general-purpose compute utility is now within reach.

Our specific contributions in this paper are as follows:

- Delineation of motivation and realistic use cases for enterprise-grade federated cloud computing
- Definition of model and open architecture for federation and interoperability of autonomous clouds to form a global fabric of resources that can be provided on demand with guaranteed service levels

- Definition of an open, loosely coupled Cloud Computing stack, where each level operates autonomously at a different level of abstraction and interacts with the layers above and below via the standardized interfaces

The rest of this paper is organized as follows. In Section 3, we present the RESERVOIR federation model and architecture, and provide definitions of the concepts used. In Section 2, we discuss specific use cases and derive requirements for RESERVOIR. In Section 4, we describe the RESERVOIR architecture in greater detail and provide rationale for the design choices we propose. Section 5 offers insight into the state-of-the-art for virtualization, grid computing, and BSM. We conclude with a summary in Section 6.

2 Use Cases and Requirements Analysis

In this section, we first review use cases that involve SAP systems. Due to their complexity, these systems serve as a good conceptual benchmark for validating the RESERVOIR model and deriving requirements. Next, we present a subset of primary requirements and key design principles. These requirements reflect the distinction of RESERVOIR compared to current cloud and virtualization offerings.

2.1 SAP Systems

SAP systems are used for a variety of business applications that differ by version and functionality (such as Customer Relationship Management (CRM) and Enterprise Resource Planning (ERP)). For a given application type, the SAP system components consist of generic parts customized by configuration and parts custom-coded for a specific installation. Certain SAP applications are composed of several loosely-coupled systems. Such systems have independent databases and communicate asynchronously by message with each other.

An SAP system is a typical three-tier system (see Figure 1) as follows:

- Requests are handled by the SAP Web dispatcher.
- In the middle tier, there are two types of components: multiple stateful Dialog Instances (DIs) and a single Central Instance (CI) that performs central services such as application-level locking, messaging, and registration of DIs. The number of DIs can be changed while the system is running to adapt to load.
- A single Database Management System (DBMS) serves the SAP system.

The components can be arranged in a variety of configurations, from a minimal configuration where all components run on a single machine, to larger ones where there are several DIs, each running on a separate machine, and a separate machine with the CI and the DBMS (see Figure 2).

2.2 The Virtualized Data Center Use Case

Consider a data center that consolidates the operation of different types of SAP applications and all their respective environments (e.g., test, production) using virtualization technology. The applications are offered as a service to external customers, or alternatively, the data center is operated by the IT department of an enterprise for internal users (i.e., enterprise employees).

A special variation that deserves mentioning is when the data center serves an on-demand, Software as a Service (SaaS) setup, where customers are external, and where each customer

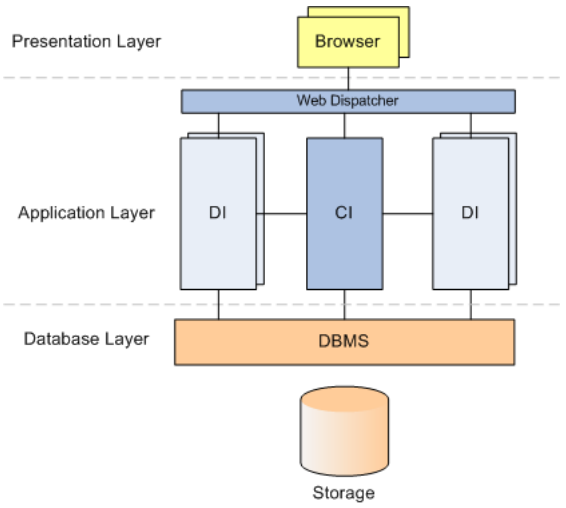

Figure 1: Abstraction of an SAP System

(tenant) gets the same base version of the application. However, each tenant configures and customizes the application to suit his specific needs. It is reasonable to assume that a tenant in this case is a small or medium business (SMB) tenant.

We briefly mention here a few aspects that are typical of virtualized data centers:

- The infrastructure provider must manage the life-cycle of the application for hundreds or thousands of tenants while keeping a very low Total Cost of Ownership (TCO). This includes setting up new tenants, backing-up the databases, managing the customizations and configurations of tenants, and getting patches and newer versions of the software from SAP (the service provider).

- Setting-up a new tenant in the SaaS for SMBs case is completely automated by a web-based wizard. The new tenant runs through a series of configuration questions and uploads master data items (e.g., product catalog and customer lists). Following these steps, the tenant is up and running, typically using a trial version. The provisioning of the resources (storage, database, and application server) is part of this automated setup.

- The customers are billed a fixed monthly subscription fee or a variable fee based on their usage of the application.

- There are several well-known approaches to multi-tenancy of the same database schema [8]. Regardless of the approach taken, multi-tenancy calls for flexible virtualization schemes where, for example, the DBMS component and the storage system are shared between multiple tenants. The main reason for this sharing is to keep the TCO per tenant at a minimum.

In summary, the key challenges in all these use cases from the point of view of the infrastructure provider are:

- Managing thousands of different service components that comprise a variety of service applications executed by thousands of virtual execution environments, on top of a complex infrastructure that also includes network and storage systems.

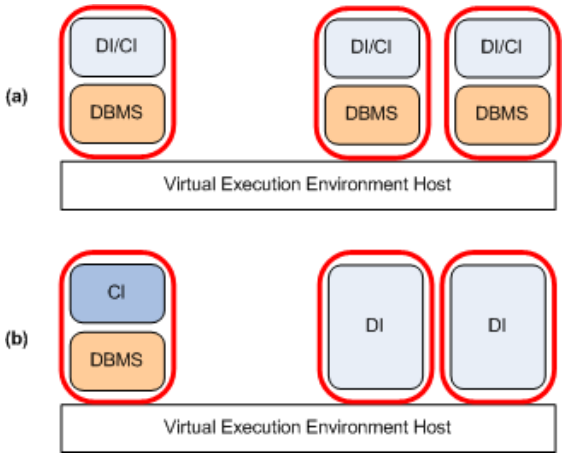

Figure 2: Sample SAP system deployments: (a) all components run in the same virtual execution environment (represented as rounded rectangles); (b) the large components (CI and DBMS) run each on a dedicated virtual execution environment. The Virtual Execution Environment Host refers to the set of components managing the virtual environments.

- Consolidating many applications on the same infrastructure, thereby increasing HW utilization and optimizing power consumption, while keeping the operational cost at minimum.
- Guaranteeing the individual SLAs of the many customers of the data center who face different and fluctuating workloads.

2.3 Primary Requirements

From the use case discussed in the previous section, we derived the following main requirements for the cloud infrastructure:

Automated and fast deployment: The RESERVOIR cloud should support automated provisioning of complex service applications based on a formal contract specifying the infrastructure SLAs. The same contract should be reused to provision multiple instances of the same application for different tenants with different customizations.

Dynamic elasticity: The RESERVOIR cloud should dynamically adjust resource allocation parameters (memory, CPU, network bandwidth, storage) of individual virtual execution environments seamlessly. Moreover, the number of virtual execution environments must be dynamically and seamlessly adjusted to adapt to the changing load.

Automated continuous optimization: The RESERVOIR cloud should continuously optimize alignment of infrastructure resources management with the high-level business goals.

Virtualization technology independence: The RESERVOIR cloud should support different virtualization technologies transparently.

3 The RESERVOIR Model for Federated Cloud Computing

In the RESERVOIR model, there is a clear separation between the functional roles of service providers and infrastructure providers. *Service providers* are the entities that understand the

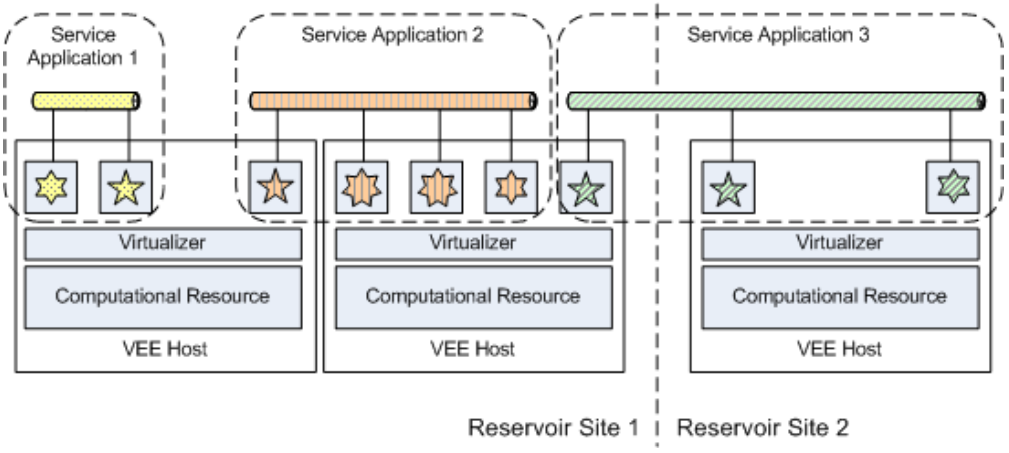

Figure 3: Service applications are executed by a set of VEEs (represented by squares) distributed across the VEE Hosts in a RESERVOIR cloud. VEEs for a particular service application may all be collocated in the same VEEH (as in service application 1); may spread across VEEHs within the same site (as in service application 2); or may even spread across sites (as in service application 3).

needs of a particular business and offer service applications to address those needs. Service providers do not own the computational resources needed by these service applications; instead, they lease resources from *infrastructure providers*, which provide them with a seemingly infinite pool of computational, network, and storage resources.

Infrastructure providers operate *RESERVOIR sites* that own and manage the physical infrastructure on which service applications execute. The federation of collaborating sites forms a *RESERVOIR Cloud*. To optimize resource utilization, the computational resources within a site are partitioned by a virtualization layer into *virtual execution environments (VEEs)* – fully isolated runtime environments that abstract away the physical characteristics of the resource and enable sharing. The virtualized computational resources, alongside the virtualization layer and all the management enablement components, are referred to as the *VEE Host*.

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. These VEEs are placed on the same or different VEE Hosts within the site, or even on different sites (see Figure 3).

A service application is deployed on the RESERVOIR Cloud using a *service manifest* that formally defines the contract and SLA between the service provider and the infrastructure provider. The service manifest is described in detail in Section 3.1.

Within each RESERVOIR site, the resource utilization is monitored and the placement of VEEs is constantly updated to achieve optimal utilization with minimal cost. Similarly, the execution of the service applications is monitored and the capacity is constantly adjusted to meet the requirements and SLA specified in the manifest.

RESERVOIR supports two modes of capacity provisioning vis-à-vis service providers:

- *Explicit* capacity requirements for *sized* service applications: In this mode, the service provider conducts sizing and capacity planning studies of the service application prior to deployment. At deployment time, the service provider precisely specifies the capacity needs of the application under specific workload conditions. In particular, the service provider specifies capacity requirements for the minimal service configuration and the

elasticity rules, i.e., the rules that govern the automated on-demand allocation and de-allocation of additional capacity under varying workload conditions. In this mode, the infrastructure provider is not committed to the high-level Service Level Objectives (SLOs) for the service (e.g., response time, throughput, etc.). Instead, the infrastructure provider commits itself to an *infrastructure SLA* that governs the terms and conditions of capacity provisioning according to the explicit requirements of the service provider. The service provider supplies explicit capacity requirements and is charged for actual capacity usage in line with the "pay as you go" model.

- *Implicit* capacity requirements for *unsized* service applications. In this mode, the service provider may have only initial sizing estimations for its service or may not have them at all. Therefore, the service is sized within the RESERVOIR service infrastructure prior to deployment. The infrastructure provider commits itself to an SLA that is formulated in terms of high-level SLOs. As in explicit capacity for sized services mode, the service provider pays for the actual usage of capacity. However, while the service provider may ask for usage reports at various level of detail, it does not have control over the sizing of its service. By default, the infrastructure provider will strive to minimize over-provisioning. In addition, the service provider may specify policies such as minimal resource utilization policy, maximal cost policy, etc.

It is important to note that the ongoing optimizations of the resource allocation are done without human intervention by the RESERVOIR software stack installed on each site.

3.1 Service Manifest

The service manifest is one of the key elements of the RESERVOIR model. The manifest specifies the structure of the service application in terms of component types that are to be deployed as VEEs.

For each of these component types, the manifest specifies a reference to a *master image*, i.e., a self-contained software stack (OS, middleware, applications, data, and configuration) that fully captures the functionality of the component type. In addition, the manifest contains the information and rules necessary to automatically create, from a single parameterized master image, unique VEE instances that can run simultaneously without conflicts [7]. The manifest also specifies the grouping of components into virtual networks and/or tiers that form the service applications. Given that the emerging Open Virtual Format (OVF) industry standard [9] includes most of this information, the service manifest will extend OVF.

The manifest also specifies the capacity requirements for an explicitly sized service application, as agreed upon between the infrastructure provider and the service provider. The minimum and maximum resource requirements of a single instance, e.g., the number of virtual CPUs, memory size, storage pool size, and the number of network interfaces and their bandwidth, are specified for each component. The capacity specification also includes the minimum and maximum number of VEEs of a particular component type. The dynamic and adaptive part of the capacity requirement is specified using a set of *elasticity rules*. These rules formally correlate monitored Key Performance Indicators (KPIs) and load parameters (e.g., response time, throughput, and number of active sessions) with resource allocations (e.g., memory, CPU, bandwidth, and number of VEEs of the same component type). These rules express how the resources allocated to the application (i.e., the resources allocated for each VEE as well as the number of VEEs of a particular component type) can be dynamically adapted (increased or reduced) to satisfy the variable demand for the service application.

Component	Web Dispatcher, CI	DI	DBMS
Master Image	`ci.img`	`di.img`	`db2.img`
# of VCPUs (min/max)	2/4	4/8	8/8
Memory Size (min/max)	4G/8G	16G/32G	32G/64G
# of NICS	2	2	1
Additional disk size	None	100G	1000G
Minimum # of instances	1	4	1
Maximum # of instances	1	20	1

Table 1: A simplified example of a service manifest for an SAP System. Although in this example, simple labels are used as master image identifiers, in a real manifest fully qualified references (such as URLs) are used.

Finally, the manifest specifies KPIs that should be monitored by RESERVOIR to verify SLA compliance and trigger the elasticity rules. This specification may include self-contained probes that periodically provide these KPIs.

To illustrate, a simplified service manifest for a SAP system (see Section 2.1) is shown in Table 1. This manifest corresponds to the configuration where a DI and the DBMS each have a separate VEE, and the CI and Web Dispatcher are encapsulated on another VEE. Notice how the manifest fixes the CI and the DBMS as single instances, and declares the DI as the elastic entity by providing it with a range of instances. To optimize cost-effectiveness, the service manifest specifies resource requirements under normal load.

To enable dynamic matching of the application capacity to the variances in workload, the manifest defines KPIs that are monitored as indications for the load that the SAP system serves. The overall response time of a certain business transaction or the number of concurrent active user sessions can be used for this purpose. An elasticity rule that triggers the addition of a new DI when this KPI exceeds a threshold value would adapt the resources allocated for the system as the workload increases. For example, if measured response time crossed a pre-specified threshold, then a new DI instance would be added.

4 The RESERVOIR Components

The RESERVOIR architecture depicted in Figure 4 is designed to provide a clean separation of concerns among the layers operating at different levels of abstraction. The rationale behind this particular layering is to keep a clear separation of concerns and responsibilities and to hide low-level infrastructure details and decisions from high-level management and service providers.

4.1 The Service Manager

The Service Manager is the highest level of abstraction, interacting with the service providers to receive their Service Manifests, negotiate pricing, and handle billing. Its two most complex tasks

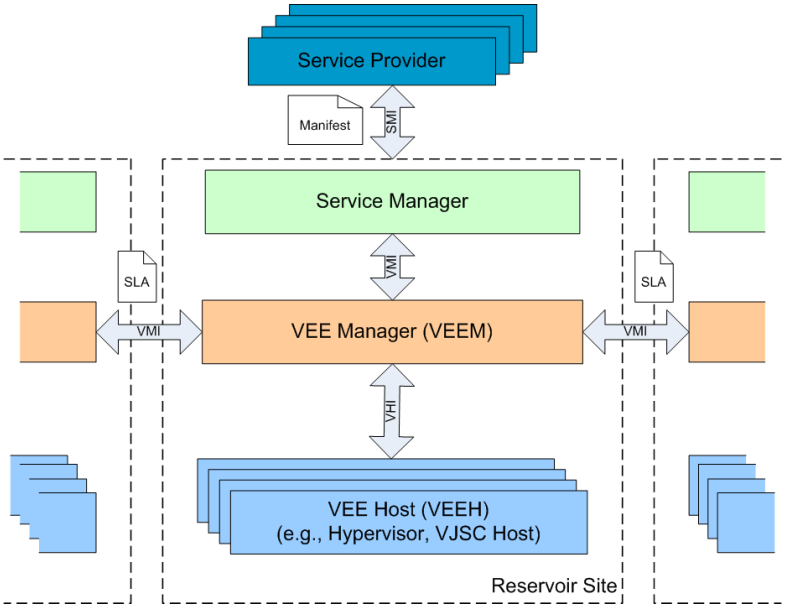

Figure 4: The RESERVOIR Architecture: major components and interfaces

are (1) deploying and provisioning VEEs based on the Service Manifest, and (2) monitoring and enforcing SLA compliance by throttling a service application's capacity.

The Service Manager receives service manifests from the service providers. Based on information in the manifest, it deploys and provisions the service application by interacting with the VEE Manager to allocate VEEs and their associated resources. From the service requirements in the manifest (i.e. SLOs, elasticity rules, etc.), the Service Manager derives a list of required resources and their configuration, as well as placement constraints based on cost, licensing, confidentiality, etc. For unsized service applications 3, the Service Manager is responsible for generating explicit rules based on site policy. Deployment and provisioning decisions are based on performance and SLA compliance and adjusted according to business considerations (e.g. costs, security, offers, etc.).

The Service Manager is also responsible for monitoring the deployed services and adjusting their capacity, i.e., the number of VEE instances as well as their resource allocation (memory, CPU, etc.), to ensure SLA compliance and alignment with high-level business goals (e.g., cost-effectiveness).

Finally, the Service Manager is responsible for accounting and billing. While existing cloud computing infrastructures tend to be quite inflexible, usually employing fixed-cost post-paid subscription models, we consider both post-paid and pre-paid billing models based on resource usage. Both models are based on the resource utilization information provided by the Service Manager accounting system, and processed according to the rules in the business information model to perform cost calculation.

4.2 The Virtual Execution Environment Manager (VEEM)

The Virtual Execution Environment Manager (VEEM) is the next level of abstraction, interacting with the Service Manager above, VEE Hosts below, and VEE Managers at other sites to enable federation.

The Virtual Execution Environment Manager (VEEM) is responsible for the optimal placement of VEEs into VEE hosts subject to constraints determined by the Service Manager. The

continuous optimization process is driven by a site-specific programmable utility function. The VEEM is free to place and move VEEs anywhere, even on the remote sites (subject to overall cross-site agreements), as long as the placement satisfies the constraints. Thus, in addition to serving local requests (from the local Service Manager), VEEM is responsible for the federation of remote sites.

At the VEEM level a service is realized as a set of inter-related VEEs (a VEE Group), and hence it should be managed as a whole. For example, the service manifest may define a specific deployment order, placement constraints (i.e., affinity rules), or rollback policies. The VEEM also provides the functionality needed to handle the dynamic nature of the service workload, such as the ability to add and remove VEEs from an existing VEE Group, or to change the capacity of a single VEE.

Operating in federated environments puts additional requirements on the VEEM for submission, management, and monitoring of VEEs on remote sites. The VEEM at the primary site performs this by taking the role of a Service Manager toward the remote VEEM in all cross-site interactions. From this distinct role definition follows a clear delegation of responsibility and separation of concerns among the Service Manager, the VEEM at the primary site, and the remote VEEM. For placement decisions, the primary VEEM takes into account agreements with other sites and detailed information about local resources before deciding on local or remote VEE placement. Notably, the primary VEEM does not get involved in the internal placement decisions on the remote site, as this is a concern of the remote VEEM. The interfaces used by the primary VEEM to interact with a remote VEEM are the same as a Service Manager uses for interactions with the (primary) VEEM.

4.3 The Virtual Execution Environment Host (VEEH)

The Virtual Execution Environment Host (VEEH) is the lowest level of abstraction, interacting with the VEE Manager to realize its IT management decisions onto a diverse set of virtualization platforms.

The VEEH is responsible for the basic control and monitoring of VEEs and their resources (e.g., creating a VEE, allocating additional resources to a VEE, monitoring a VEE, migrating a VEE, creating a virtual network and storage pool, etc.). Each VEEH type encapsulates a particular type of virtualization technology, and all VEEH types expose a common interface such that VEEM can issue generic commands to manage the life-cycle of VEEs. The receiving VEEH is responsible for translating these commands into commands specific to the virtualization platform abstracted by it.

Given that VEEs belonging to the same application may be placed on multiple VEEHs and even extend beyond the boundaries of a site, VEEHs must support isolated virtual networks that span VEEHs and sites. Moreover, VEEHs must support transparent VEE migration to any compatible VEEH in a RESERVOIR cloud, regardless of site location or network and storage configurations.

4.4 Layers of Interoperability

The layered design stresses the use of standard, open, and generic protocols and interfaces to support vertical and horizontal interoperability between layers. Different implementations of each layer will be able to interact with each other. The Service Management Interface (SMI) with its service manifest exposes a standardized interface into the RESERVOIR Cloud for service providers. The service provider may then choose among RESERVOIR cloud providers knowing that they share a common language to express their business requirements. The VEE

Management Interface (VMI) simplifies the introduction of different and independent IT optimization strategies without disrupting other layers or peer VEEMs. Further, VMI's support of VEEM-to-VEEM communication simplifies cloud federation by limiting the horizontal interoperability to one layer of the stack. The VEE Host Interface (VHI) will support plugging-in of new virtualization platforms (e.g., hypervisors), without requiring VEEM recompilation or restart. RESERVOIR's loosely coupled stack reference architecture should promote a variety of innovative approaches to support cloud computing.

5 Related Work

In this section, we briefly review the state-of-the-art in areas related to the RESERVOIR model and architecture.

Virtualization Technologies

Virtualization has re-emerged in recent years as a compelling approach to increasing resource utilization and reducing IT services costs. The common theme of all virtualization technologies is hiding the underlying infrastructure by introducing a logical layer between the physical infrastructure and the computational processes. Virtualization takes many forms. System virtualization [10], also commonly referred to as server virtualization, is the ability to run multiple heterogeneous operating systems on the same physical server [5]. With server virtualization, a control program (commonly known as "hypervisor" or "virtual machine monitor") is run on a given hardware platform, simulating one or more other computer environments (virtual machines). Each of these virtual machines, in turn, runs its respective "guest" software, typically an operating system, which runs just as if it were installed on the stand-alone hardware platform. Additional forms of virtualization include storage virtualization and network virtualization, logical representations of the physical storage and network resources.

Distributed Management of Virtualization

Given the growing popularity of virtualization, many commercial products and research projects, such as such as OpenNEbula, Platform VM Orchestrator, IBM Virtualization Manager, and VMware DRS are being developed to dynamically overlay virtual machines (VMs) over physical resources. In general, these efforts try to extend the benefits of virtualization from a single resource to a pool of resources, decoupling the VM not only from physical infrastructure but also from physical location.

There are also some commercial and scientific infrastructure cloud computing initiatives, such as Globus VWS, Eucalyptus, and Amazon, which provide remote interfaces for control and monitoring of virtual resources. Although these solutions simplify the management of VMs on a distributed pool of resources, none of these initiatives for distributed VM management evaluate its extension to a grid-like environment where different infrastructure sites could collaborate to satisfy peak or fluctuating demands.

In the context of RESERVOIR, grid interfaces and protocols [11] may enable the required interoperability between the clouds or infrastructure providers. However, RESERVOIR will strive to overcome interoperability challenges often posed in traditional grids by a very strict separation of concern and minimalistic interfaces, reducing cross-site concerns to a minimum. RESERVOIR also needs to expand substantially on the current state-of-the-art for grid-wide accounting [12] and increase flexibility to support different billing schemes and accounting for

services with indefinite lifetimes as opposed to finite jobs, with support to account for utilization metrics relevant for virtual machines.

Business Service Management

Business Service Management (BSM) is a management strategy that links IT infrastructure management objectives to the goals of the business. In contrast to a technologically-oriented management approach, in BSM IT shares the same overall targets with the business such as growing revenue, reducing costs, lowering business risk, increasing customer satisfaction, guaranteeing a given return on investment, complying with regulations, etc.

A key aspect of BSM is SLA management. In RESERVOIR, new SLA management challenges arise due to the dynamic federation of infrastructure providers. Some emerging approaches to SLA management that can be leveraged in RESERVOIR include data-driven methods (e.g., dynamic setting of service-level objectives based on statistical analysis [13]), model-driven methods (e.g., developing performance models that govern the design of ICT infrastructure [14]), and language-based approaches (i.e., using formal languages to specify SLAs to assure their consistency and unambiguity [15]). SLA monitorability and formal definition for application-service provisioning was studied in [16]. Some earlier work considered SLA management in federated environments; however, that line of research [17] did not address the special considerations of supporting migration and virtualization.

6 Summary

This paper presents a new Open Federated Cloud Computing model, Architecture, and functionality as being developed in the RESERVOIR project. The RESERVOIR model explicitly addresses the limited scalability of a single-provider cloud, the lack of interoperability among cloud providers, and the lack of built-in Business Service Management support in current cloud offerings.

Cloud computing has the potential of becoming a revolutionary technology that changes the way service computing is performed. We believe that the principles introduced in this work are essential for the cloud computing vision to materialize to its full extent.

The RESERVOIR project is in its early stages. The implementation and evaluation of the concepts outlined in this paper are underway as we go to press. Discussing these results is deferred for future work.

Acknowledgments

We would like to thank Eliot Salant from IBM and Juan Hierro from TID – without their work, this project would not have been possible. In addition, we would like to thank Shimon Agassi, Katherine Barabash, David Hadas, Irit Loy, and Edi Shmueli from IBM; Manuel Díaz, David Perales and Emilio Torres from Telefónica; Daniel Henriksson, Francisco Hernandez, and Johan Tordsson from Umeå; Mark Wusthoff from SAP; Fritz Ferstl from Sun; and Javier Fontan, Luis Gonzalez, Eduardo Huedo, Rafael Moreno, and Tino Vazquez from UCM for their help materializing the ideas expressed in this document.

References

[1] J. Meattle, "YouTube vs MySpace growth," http://blog.compete.com/2006/10/18/youtube-vs-myspace-growth-google-charts-metrics/, October 2006.

[2] N. Carr, *The Big Switch - Rewiring the World from Edisson to Google*. W. W. Norton, January 2008.

[3] I. Foster, C. Kesselman, and S. Tuecke, "The anatomy of the grid: Enabling scalable virtual organizations," *International Journal of High Performance Computing Applications*, vol. 15, no. 3, pp. 200–222, 2001.

[4] K. Appleby, S. Fakhouri, L. Fong, G. Goldszmidt, M. Kalantar, S. Krishnakumar, D. Pazel, J. Pershing, and B. Rochwerger, "Océano-SLA based management of a computing utility," *Proceedings of 6th IEEE/IFIP International Symposium on Integrated Network Management*, pp. 855–868, 2001.

[5] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," in *SOSP '03: Proceedings of the 19th ACM symposium on Operating systems principles*. ACM, 2003, pp. 164–177.

[6] M. Turner, D. Budgen, and P. Brereton, "Turning software into a service," *Computer*, vol. 36, no. 10, pp. 38–44, October 2003.

[7] O. Goldshmidt, B. Rochwerger, A. Glikson, I. Shapira, and T. Domany, "Encompass: Managing functionality," in *Proceedings of 21th International Parallel and Distributed Processing Symposium (IPDPS 2007)*, March 2007, pp. 1–5.

[8] S. Aulbach, T. Gurst, D. Jacobs, A. Kemper, and J. Rittinger, "Multi-tenant databases for software as a service," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, June 2008, pp. 1195–1206.

[9] DMTF, "The Open Virtualization Format Specification, Version 1.0.0a," http://www.dmtf.org/standards/published_documents/DSP0243_1.0.0.pdf.

[10] G. J. Popek and R. P. Goldberg, "Formal requirements for virtualizable third generation architectures," *Communications of ACM*, vol. 17, no. 7, pp. 412–421, 1974.

[11] I. Foster and C. Kesselman, "Globus: A metacomputing infrastructure toolkit," *International Journal of Software Architecture*, vol. 11, no. 2, pp. 115–128, 1997.

[12] P. Gardfjäll, E. Elmroth, L. Johnsson, O. Mulmo, and T. Sandholm, "Scalable grid-wide capacity allocation with the SweGrid Accounting System (SGAS)," *Concurrency and Computation: Practice and Experience*, vol. 20, no. 18, pp. 2089–2122, December 2008.

[13] D. Breitgand, E. Henis, O. Shehory, and J. Lake, "Derivation of response time service level objectives for business services," *Proceedings of the 2nd IEEE/IFIP International Workshop on Business-Driven IT Management BDIM*, pp. 29–38, May 2007.

[14] J. Sauve, F. Marques, A. Moura, M. Sampaio, J. Jornada, and E. Radziuk, "SLA design from a business perspective," in *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2005, pp. 72–83.

[15] D. Lamanna, J. Skene, and W. Emmerich, "SLAng: A language for service level agreements," in *Proc. of the 9th IEEE Workshop on Future Trends in Distributed Computing Systems*. IEEE Computer Society Press, 2003, pp. 100–106.

[16] J. Skene, A. Skene, J. Crampton, and W. Emmerich, "The monitorability of service-level agreements for application-service provision," in *WOSP '07: Proceedings of the 6th international workshop on Software and performance*. ACM, 2007, pp. 3–14.

[17] P. Bhoj, S. Singhal, and S. Chutani, "SLA management in federated environments," *Proceedings of the 6th IFIP/IEEE International Symposium on Integrated Network Management*, pp. 293–308, 1999.

Biographies

Benny Rochwerger IBM Haifa Research Lab., University Campus, Haifa 31905, Israel (rochwer@il.ibm.com)

Mr. Rochwerger has an M.S. in computer science from the University of Massachusetts at Amherst and a B.Sc. in computer engineering from the Technion – Israel Institute of Technology. Since joining IBM in 1995, he has worked in virtualization management, autonomic computing, event processing, grid computing, distributed graphics, and networking. Currently, he is the lead architect for the RESERVOIR project. Prior to IBM, Benny worked at Avaya, Data General, Legent Corporation, and Fidelity Medical.

David Breitgand IBM Haifa Research Lab., University Campus, Haifa 31905, Israel (davidbr@il.ibm.com)

Dr. Breitgand received his B.Sc., M. Sc, and Ph.D. in computer science from the Hebrew University of Jerusalem in 1993, 1998, and 2003 respectively. David joined the IBM Haifa Research Lab in 2003 where he is a member of the Storage and Performance Management group, leading the group's research efforts on end-to-end performance analysis and management of networked storage and systems. In RESERVOIR, David focuses on algorithms for cost-effective capacity provisioning for RESERVOIR services.

Eliezer Levy SAP Labs Israel, 15 Hatidhar Str., Raanana 43665, Israel (levy@sap.com)

Dr. Levy has a Ph.D. in computer sciences from the University of Texas at Austin and a B.Sc. in computer science from the Technion – Israel Institute of Technology. Eliezer is currently a researcher in SAP Research. Previously, he was the Chief Architect of the Small Businesses business unit in SAP. Eliezer has 18 years of experience in various R&D roles in Intel, Tandem Computers and an Israeli start-up.

Alex Galis University College London, Torrington Place, London WC1E 7 JE, United Kingdom (a.galis@ee.ucl.ac.uk) Dr. Galis is a visiting professor at the Department of Electronic and Electrical Engineering at UCL He has co-authored and published more than 125 refereed journal/conference papers and more than 100 reports and six books on network and service management, intelligent services, programmable networks and services, virtualization of resources, and grid systems.

Kenneth Nagin IBM Haifa Research Lab., University Campus, Haifa 31905, Israel(nagin@il.ibm.com)

Mr. Nagin received a B.A from the University of Madison and a B.S. from the University of Pittsburgh. Since joining IBM in 1985, he has worked on copy services, disaster recovery solutions and storage management tools for direct access storage devices, model-based software test generation and test consultation, and Blade Center Management tools. He holds over 20 patents. His current research is concerned with cloud computing server virtualization.

Ignacio M. Llorente Universidad Complutense de Madrid, C/ Profesor José García Santesmases, 28040 Madrid, Spain (llorente@dacya.ucm.es)

Dr. Llorente has over 15 years of research experience in the field of High-Performance

Parallel and Distributed Computing. Currently, he is a full professor in computer architecture and technology at UCM, where he leads the Distributed Systems Architecture Group.

Ruben Montero Universidad Complutense de Madrid, C/ Profesor José García Santesmases, 28040 Madrid, Spain (rubensm@dacya.ucm.es)
Dr. Montero is an associated professor in computer architecture and technology in the Department of Computer Architecture and System Engineering at UCM, where he is part of the Distributed Systems Architecture Group. He has held several research appointments at ICASE - NASA LaRC.

Yaron Wolfsthal IBM Haifa Research Lab., University Campus, Haifa 31905, Israel (wolfsthal@il.ibm.com)
Dr. Wolfsthal heads the System and Services area at the IBM Haifa Research Lab. This area is one of IBM's EU innovation centers and its mission is to develop leading-edge technologies for IBM's advanced ICT products and services, including virtualization infrastructures and systems management. Dr. Wolfsthal has 16 years of experience in various R&D and management roles. He holds B.Sc., M.S., and Ph.D. degrees in computer science, all from the Technion - Israel Institute of Technology.

Erik Elmroth Umeå University, SE-901 87 Umeå Sweden (elmroth@cs.umu.se)
Dr. Elmroth is the scientific leader of the Grid computing research group at Umeå University, Sweden. He is also an associate professor and serves as Deputy Head of the Department of Computing Science and Deputy Director of the High Performance Computing Center North (HPC2N). He has held positions at NERSC, LBNL, University of California Berkeley, and MIT. He is also a member of the Swedish Research Council's Committee for Research Infrastructures, vice chair of its expert panel on e-science, and Scientific Secretary of the former e-science group of the Nordic Council of Ministers.

Juan Cáceres Telefónica I+D., C/ Emilio Vargas, 6. 28043 Madrid, Spain (caceres@tid.es)
Mr. Cáceres has a Research M.Sc. (Bologna Process) and an M.Sc. in computer science from the Technical University of Madrid (UPM) and is currently working toward his Ph.D. in cloud computing at the Complutense University of Madrid (UCM). Juan has over eight years of experience in middleware and distributed systems development at Telefónica I+D. He has also experience in open source software by setting up the MORFEO Community where he leads the Middleware Platform Group.

Muli Ben-Yehuda IBM Haifa Research Lab., University Campus, Haifa 31905, Israel (muli@il.ibm.com)
Mr. Ben-Yehuda is a systems researcher at IBM's Haifa Research Lab, where he was recently named an IBM Master Inventor. His research interests include I/O for virtualized systems, IOMMUs, smart IO devices, and neat things you can do with virtual machines. He has contributed to numerous operating systems and hypervisors, including the Linux kernel, the Xen virtual machine monitor, and the Linux Kernel-based Virtual Machine project (KVM).

Wolfgang Emmerich University College London, Torrington Place, London WC1E 7 JE, United Kingdom (W.Emmerich@cs.ucl.ac.uk)
Dr. Emmerich graduated from the Universität Dortmund and obtained his doctorate from the Universität of Paderborn. He is a professor of Distributed Computing in the Department of Computer Science, where he is Director of Research and Head of the Software Systems Engineering Group. He is a member of the Editorial Board of IEEE Transactions

of Software Engineering and a member of the IET and Chartered Engineer. Aside from a wide range of research publications, he is author of "Engineering Distributed Objects", a major textbook published by Wiley.

Fermín Galán Telefónica I+D., C/ Emilio Vargas, 6. 28043 Madrid, Spain (fermin@tid.es)
Mr. Galán has a M.Sc. degree in telecommunications engineering from the Technical University of Madrid and currently is working towards his Ph.D in telematics at the same university. He is an R&D Engineer at Telefónica I+D in the Business Oriented Infrastructure group and has been involved in several R&D European collaborative projects (Euro6IX, DAIDALOS or NOBEL2).