RESERVOIR

RESOURCES AND SERVICES VIRTUALIZATION WITHOUT BARRIERS

Deliverable D.6.4.1
Guide to the RESERVOIR Framework
Release 1.0

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.

Project Number	:	215605
Project Title	:	RESERVOIR: Resources and Services Virtualization without Barriers
Deliverable Type	:	Manual

Deliverable Number	:	D.6.4.1
Title of Deliverable	:	Guide to the RESERVOIR Framework
Nature of Deliverable	:	Manual
Dissemination Level	:	Public
Internal Document Number	:	-
Contractual Delivery Date	:	
Actual Delivery Date	:	
Contributing WPs	:	
Editor(s)	:	Philippe Massonet

Document History

Version	Date	Comment
1.0	10/12/09	Initial internal release
1.0	25/01/10	Final structure defined
1.0	23/01/10	removed the RESERVOIR architecture section
1.0	25/02/10	Added table showing correspondance with software prototypes

Contents

List of Figures

List of Tables

1 Introduction

The RESERVOIR framework is a collection of open source software, open specifications and publications produced by the RESERVOIR FP7 project which are available for download through the RESERVOIR web site, www.reservoir-fp7.eu. Note that while some components of the end-to-end RESERVOIR solution are freely downloadable as open source, other components are proprietary to the project. In the latter case, the High Level Architecture Document provided in the Framework will contain the required documentation to allow others to implement their own version of the component - the documentation explains both the functionality of the component as well as the required external interface specifications which the component must provide to integrate in with the rest of the stack.

1.1 Leveraging RESERVOIR open source contributions

As described above, portions of the RESERVOIR stack have been released as open source, and more are due to be released during the third year. This document describes the software which is available as open source, lists where it is downloadable from, and describes the architectural component that it implements.

2 RESERVOIR Framework: Subsystems, Components and Specifications

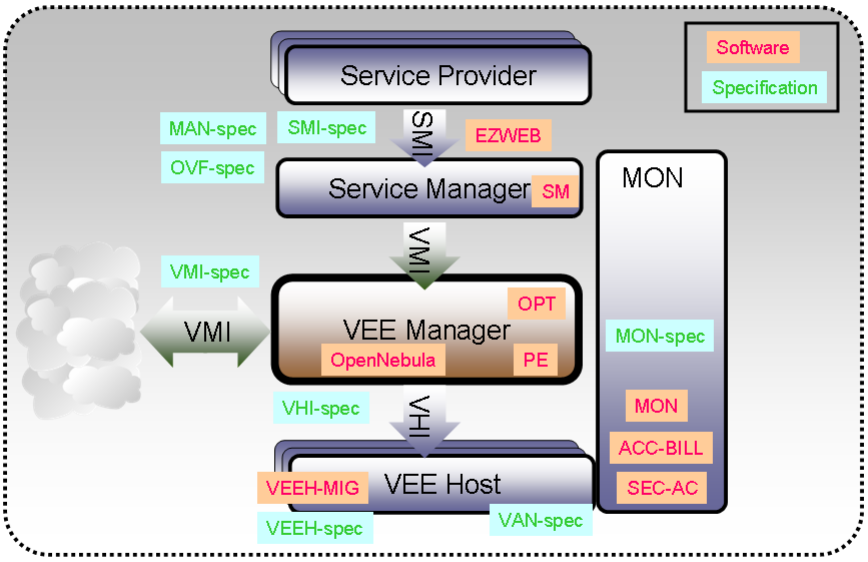

Figure 2.1: RESERVOIR architecture and framework

Figure 2.1 shows the RESERVOIR architecture, depicting the framework subsystems and components. The following software and specifications are available for each layer of the RESERVOIR architecture as will be described:

Service Manager (SM)

- EZWEB software to allow building a portal to the cloud infrastructure,
- SM software allows building a service manager that works with service manifests and OVF files,
- The Service Manifest Language Specification (MAN-spec) describes the service manifest language syntax,
- The Service Manager extensions to OVF (OVF-SPEC) explains how the OVF standard was extended for managing multi-tier applications,
- The SMI specification implementation (SMI-spec) describes the standard interface to the service manager.

Virtual Execution Environment Manager (VEEM)

- OpenNebula software provides the basic functionality to build a virtual execution environment manager.
- The Placement Engine (PE) software can be integrated with OpenNebula to provide policy based placement and management of VEE,

- VMI specification/implementation (VMI-spec) describes the standard interface to the VEEM.

Virtual Execution Environment Host (VEEH)

- : VEEH Migration (VEEH-MIG) software provides support for migrating virtual machines between sites. Modifications have been made to QEMU/kvm and libvirt, and will be available in the next release of QEMU.

In addition to the horizontal component layers, the following software and specifications deal with cross cutting vertical issues:

- The Monitoring system (MON) software provides a cloud monitoring subsystem that can be integrated with the rest of the software stack using publish/subscribe techniques. MON is described in detail by the MON-spec specification.
- Accounting and Billing (ACC-BILL) software provides a subsystem to deal with accounting and billing in a cloud infrastructure,
- Security and Access (SEC-AC) software provides security services for securing access to the SMI and VMI interfaces.

2.1 RESERVOIR component specifications

Table 2.1 describes the RESERVOIR component specifications which can be downloaded from either the "'Documents"' section of the RESERVOIR Framework or an external source. The type of license that applies to the specification is given.

name.	Definition	Licence	URL
VMI specification/implementation (VMI-spec)	The OGF OCCI-WG site maintains the last version of the specification and OpenNebula includes a reference implementation	Apache v2.0	http://www.occi-wg.org/
SMI specification/implementation (SMI-spec)	The SMI is the top-level API to interact with the RESERVOIR Service Manager and will be spoused to Service providers to manage their Services. Specification of the Service Manager Interface will be available at RESERVOIR Site, the implementation will be part of Claudia (Telefónica I+D components)	Creative Commons, AGPL	http://claudia.morfeo-project.org
Service Manager extensions to OVF (OVF-spec)	Specification of the OVF format extensions will be available at RESERVOIR site, the implementation of the OVF parser that generates a Java representation of it will be part of Claudia	Creative Commons, AGPL	A4 Scientific Report in http://www.reservoir-fp7.eu, and http://claudia.morfeo-project.org
VEEH Specification (VEEH-spec)	VEE Host (VEEH) interacts with the different physical VEE Hosts in the site to perform the elementary control and monitoring of VEEs and their resources (e.g. creating a VEE, allocating additonal resources to a VEE, monitoring a VEE, migrating a VEE, creating a virtual network and storage pool, etc.). This code is maintained and distributed by IBM Haifa Research Lab.	...	http://www.reservoir-fp7.eu/ as part of the VEEH A3 Scientific Report Feb 2009
VHI Specification (VHI-spec)	VEE Host Interface (VHI) is the interface provided by VEEH to VEEM. The interface is esentially some modifications to libvirt (libvirt.org) to to support VEE creation, suspension, resumption, and destruction. This code and specificaton is maintained and distributed by IBM Haifa Research Lab.	...	http://www.reservoir-fp7.eu/ as part of the VEEH A3 Scientific Report Feb 2009
VAN Specification (VAN-spec)	Virtual Application Network (VAN), is a level 2 network solution that enables virtual network services across subnets and across administrative boundaries. This code and specificaton is maintained and distributed by IBM Haifa Research Lab.	...	http://www.reservoir-fp7.eu/ as part of the VEEH A3 Scientific Report Feb 2009 VEEH Scientific Report Feb 2009

Table 2.1: RESERVOIR Framework Software Specifications

name.	Definition	Licence	URL
Monitoring System Specification (MON-spec)	The monitoring system is pervasive as: it is required by most of the components of the service cloud; it cuts across the layers of the system creating vertical paths; and it spans out across all the clouds in a federation in order to link all the virtual machines of a service. The monitoring system is designed and built to be fit for the purpose of infrastructure and service monitoring. It needs to be for the whole of service management, and so it should cover SLAs, elasticity, QoS? , etc.. It is the mechanism that closes the loop from the initial deployment, through execution, and back to the Service Manager. The monitoring system is there to gather data from components within the RESERVOIR architecture. Where the VEEM is responsible for managing all of the VEEs within the system, the monitoring system is responsible for the data sources and probes, which measure those components.	...	www.reservoir-fp7.eu
Service Manifest Language Specification (MAN-spec)	The definition of the service manifest language is based on the specification of three complementary facets: the abstract syntax, the well-formedness rules, and the modeling element semantics. The abstract syntax of the manifest language is modeled using the Essential Meta-Object Facility (EMOF), an OMG standard part of the Model Driven Architecture initiative for describing the structure of meta-data, and embedded within an objectoriented model of the RESERVOIR infrastructure. Because the manifest describes the way in which the RESERVOIR infrastructure should provision a service application, the semantics of the language can be expressed as constraints over the operation of infrastructure components. These constraints are formally defined using the Object Constraint Language (OCL), a language for describing consistency properties, providing the static and behavioural semantics of the language. It is then possible to derive a concrete syntax for the manifest using different implementation languages and tools (e.g. XML) and code artifacts that can be used to support the use of the language and verify the correct provisioning and adaptation of services.	...	www.reservoir-fp7.eu

Table 2.2: RESERVOIR Framework Software Specifications

2.2 Publically Available Software Components

2.2.1 Overview

The publically available RESERVOIR framework software components are described in table 2.3.

name.	Definition	Licence	URL
EZWEB	EZWEB: The portal technology that is used to build the RESERVOIR portal (also developed in other projects)	AGPL	http://ezweb.morfeo-project.org
SM	Service Manager: The Service Manager controls the service lifecycle (deployment, reconfiguration, stop, etc) and dynamic scalability (based on SLA and business targets) in the RESERVOIR federated cloud by requesting the apropriate resources.	AGPL, MPL	Will be released under the name Claudia (Telefonica I+D components) http://claudia.morfeo-project.org
VEEM/OpenNebula	VEEM-core/OpenNebula: Open-source Toolkit to build private, public and hybrid clouds	Apache v2.0	www.opennebula.org
VEEH-MIG (VEE/Relocation of VMs without shared storage)	This code supports migraton of VMs between hosts that do not share storage. Modifications have been made to QEMU-kvm and libvirt.	LGPL	www.qemu.org
PE	Policy Engine: Open source configurable resource placement component. The Policy Engine is responsible for the VEE placements within a single Reservoir Site. Considering local policies, plugged optimizer, and VEE requirements/constraints, it periodically and continously determine what is the optimal placemen for each VEE communication such request to the VEEM-core.	Apache v2.0	http://www.reservoir-fp7.eu/
OPT	IBM OPTIMIZER: the IBM optimizer provides a set of algorithms policies to be plugged in and used by the Policy Engine component during its periodical optimization placement cycle.	to be defined	http://www.reservoir-fp7.eu/
MON	Monitoring System Code	GPL	www.reservoir-fp7.eu
ACC-BILL	Accounting and billing: component that manages the federated accounting and billing in RESERVOIR	AGPL	http:// claudia.morfeo.project.org
SEC-AC	Security - Access control: Access control architecture and security services including a DDOS countermeasure strategy and implementation	AGPL, MPL	Will be released on http://claudia.morfeo-project.org with a reference to the SM

Table 2.3: RESERVOIR Framework Software Components

2.2.2 EZWEB Features

EZWEB provides the key technologies for building the front end layer of a new generation SOA architecture that supports the following criteria:

- End-users must be able to access a wide range of content and services, and be able to personalize it themselves.
- End-users are able to create resources as well as share and exchange knowledge and resources with other end users. it aims at accelerating productivity and innovation.
- Context awareness is essential to providing a flexible interaction

The features of EZWEB are shown in table 2.4.

Feature	Function
Mashup Environment	Each user has his/her own customizable workspace where gadgets may be loaded (web widgets) to give access to useful information and services
Gadget Wiring	Through a publish/subscribe mechanisms gadgets may share data to integrate different services at the front-end side (i.e. see the resource status at a monitoring gadget when the resource is selected in the user's resource tree).
Gadget Repository	RESERVOIR's layers (SM, VEE and VEEH) may publish the available gadgets that allows users operating and getting data with their own services and Cloud adminstrators operating the different architecture components. Gadgets are connected to RESERVOIR middleware through RESTFull APIs.

Table 2.4: EZWEB Features

There are no other dependencies with other RESERVOIR components.

2.2.3 VEEM/OpenNebula, VEEM-core/OpenNebula Features

VEEM-core/OpenNebula is an open and flexible tool that fits into existing data center environments to build any type of Cloud deployment. VEEM-core/OpenNebula can be primarily used as a virtualization tool to manage your virtual infrastructure in the data-center or cluster, which is usually referred as Private Cloud. VEEM-core/OpenNebula supports Hybrid Cloud to combine local infrastructure with public cloud-based infrastructure, enabling highly scalable hosting environments. VEEM-core/OpenNebula also supports Public Clouds by providing Cloud interfaces to expose its functionality for virtual machine, storage and network management.

VEEM-core/OpenNebula is capable of running in an autonomous mode (i.e. without other RESERVOIR components), with the features defined in table 2.5.

The dependencies with other RESERVOIR components are described in table 2.6:

Feature	Function
Internal Interfaces for Administrators and Users	Unix-like CLI and XML-RPC API to manage VM life-cycle and physical boxes; and libvirt virtualization API
Scheduler	Requirement/rank matchmaker allowing the definition of workload and resource-aware allocation policies such as packing, striping, load-aware, affinity-aware; and support for advance reservation of capacity through the Haizea VM-based lease manager
Virtualization Management	Xen, KVM and VMware connectors; and generic libvirt connector to other VM managers. Virtual Box planned for 1.4.2
Image Management	General mechanisms to transfer and clone VM images
Network Management	Definition of isolated virtual networks to interconnect VMs
Service Management and Contextualization	Support for multi-tier services consisting of groups of interconnected VMs, and their auto-configuration at boot time
Security	Management of users by the infrastructure administrator
Fault Tolerance	Persistent database backend to store host and VM information
Scalability	Tested in the management of medium scale infrastructures consisting of hundreds of servers and VMs
Installation	Installation on a UNIX cluster front-end without requiring new services in the remote resources; and distributed in Ubuntu 9.04 (Jaunty Jackalope)
Flexibility and Extensibility	Open, flexible and extensible architecture, interfaces and components, allowing its integration with any product or tool in the virtualization and Cloud ecosystems and management tool in the data center

Table 2.5: VEEM/OpenNebula, VEEM-core/OpenNebula Features

Dependency	Description
EZWEB	VEEM-core/OpenNebula can be used through this portal technology, thus providing a web interface
SM	VEEM-core/OpenNebula, in conjunction with the SM, enabled the service lifecycle management
SEC-AC	This component enables VEEM-core/OpenNebula to have an access control architecture
PE	The Policy Engine allows VEEM-core/OpenNebula to manage VEE placements within a single Reservoir Site, considering local policies, plugged optimizer, and VEE requirements/constraints.
OPT	The IBM OPT, in conjunction with the PE, optimizes the VEE placement.

Table 2.6: OpenNebula dependencies

2.2.4 SM Features

SM is responsible for the instantiation of service applications (controlling the Service Lifecycle) and dynamically asking for virtualized resources to the underlying layer (VEEM), trying to avoid over/under provisioning and over-costs based on SLAs and business rules protection techniques. The Service Managers interface (SMI, Service Manager Interface) allows Service Providers to control the service provisioning lifecycle using a Service Manifest (based on the Open Virtualization Format) that declares the service components (packaged in virtual machine images), service requirements, Service QoS (Monitoring, SLA targets and Elasticity Rules) and Business Directives (cost, confidentiality, licensing, regional, providers, restrictions,). Claudia is the core component of the RESERVOIR's Service Manager. The features of the SM are shown in table 2.7.

Feature	Function
The Service Lifecycle Manager	automates the service deployment and dynamic provisioning.
The OVF Manager	is an independent library to manipulate OVF (including the Claudia extensions)
The Scalability and Optimization Manager	it controls the configuration and service scalability.
The Business Model Manager	it controls the business aspects (customers, accounting and billing) and the business rules that affects the service provisioning.

Table 2.7: SM Features

The dependencies with other RESERVOIR components is described in table 2.8:

Dependency	Description
Cloud Dashboard	it is the web GUI based on the EzWeb mashup for the Service Manager.
Monitoring System	it is the component that stores and distributes the status of the deployed Services data to the rest of the Service Manager components and to the Cloud Users. It is based on the WASUP Project.

Table 2.8: SM Dependencies

2.2.5 ACC-BILL Features

Accounting in federated cloud environment faces challenges that are not present in other Grid and Cloud environments, some inferred by the uncertainties associated with cross-domain information sharing and some due to the flexible nature of the services being accounted for. The main responsibility of the accounting system is to collect and manage the current usage of each service, in order to provide the billing system with the data required for billing each consumer. Apart from that, the system is also involved in analysing the feasibility of a given service deployment to ensure that the deployment is profitable from the infrastructure providers point of view.

Similarly to how accounting builds atop on monitoring, the billing layer can safely leave data collection and management concerns to the accounting system and instead focus on the process of billing. The focus of the billing layer is to convert the raw usage data maintained by the accounting system into credits, and also to compose bills and usage listings that are supplied to the infrastructure consumers.

Feature	Function
Location unawareness	The accounting system does not have to be aware of where the VEE is deployed in order to collect the accounting data, and the VEE is not aware of the data being propagated to another site if the VEE is deployed remotely.
Service elasticity	The amount of VEEs making up a service can change dynamically without affecting the accounting and billing system.
Service billing	The billing for the execution is done on a per service basis, and not for each VEE. Multiple billing methods are supported (including, e.g., postpaid and prepaid). A single customer can have several accounts, where each account is of either payment type.
Complex pricing	The function that calculate prices from the accounting information is able to incorporate complex pricing rules depending of several factors such as, e.g., customer history or seasonal discounts. The complexity is managed within the business model, making it a policy decision rather than a mechanism.
Flexible data format	The format used for accounting data includes both hardware measurements, such as CPU or memory consumption, and any application specific KPIs (e.g. database transactions per second). It is also possible to add aggregation functionality in the future, without modifying the format of the accounting data.
Service accounting	The accounting system is suitable for long running services, and not limited to tasks with a limited execution time. It is also important that the solution supports aborting or suspending a running service due to, e.g., a lack of credits.
Compensations	The system enables to account for both usage and compensations (due to breaking SLAs), supporting arbitrary functions for calculating the compensations.

Table 2.9: ACC-BILL Features

The features of the accounting and billing software are shown in table 2.9.
The dependencies with other RESERVOIR components is described in table 2.10:

Dependency	Description
MON	The accounting system relies on the monitoring framework in order to take usage and measures.

Table 2.10: ACC-BILL dependencies

2.2.6 PE Features

The Policy Engine is an open and programmable placement optimizer engine, whose major task is to take as input a list of objects to be placed into a certain number of containers, and produce as output the optimal placement deployment instructions. The output result is depending both on the constraints between the objects/containers (in terms of capacity, consumption, prices, ..) and on the preferred policy and algorithm the Policy Engine Administrator has chosen (load balancing, power saving or any other dynamically pluggable kind of policy/algorithm). In the context of RESERVOIR, the Policy Engine main objectives are to determine admittance of a whole service (by computing the percentage of failover rate upon accepting it)

and to govern VEEM/OpenNebula component by determining, for each VEE/Virtual machine, the optimal destination resource (in terms of Host or Remote Site) where to place/migrate it. Policy Engine features are listed in table 2.11.

Table 2.11: PE Features

Feature	Function
Open Source	The core part of the Policy Engine is completely open and runs under Apache 2.0 license
Programmable	The modular design and architecture is focused on adaptivity to any kind of environment: open and public interfaces can be reimplemented by any third party contributor and the resulting library being attached to a running Policy Engine in a dynamic way
Configurable and Responsive	Immediate reaction upon configuration parameters changes
Performance	Due to its variety of available algorithms (potentially infinite) the Policy Engine is able to respond optimally to any kind of problem/environment by programming the ”suitable” best solution/algorithm
Modularity	very low effort would be needed to adapt the Policy Engine to other framework different from RESERVOIR’s one

The dependencies with other RESERVOIR components is described in table 2.12:

Table 2.12: PE dependencies

Dependency	Description
VEEM-DB	Information on object to be allocated (VEEs) and possible containers (HOSTs and SITEs) relies on a persistend database located at VEEM level
Admission Control	A new service to be provisioned is executed and deployed only upon the approval of the Admission Control component
Placement Optimizer	Built in algorithms used for policies are implemented in a separate plugged component
COIN-OR	Default optimization algorithms are based on libraries provided by an open source community for operation research (http://www.coin-or.org)
Heuristics for Federation	Remote placement solution is performed using the outcome, policies, algorithms and libraries provided separately by a different work package
VEEM-Core	The result of the placement optimization decision are communicated to govern the VEEM-core, which is responsible for actually issue the deployment action
EzWeb	Policy Engine configuration, parameters tuning, information monitoring and policies management is graphically possible due to the development of several widgets attached to the RESERVOIR administration portal

2.2.7 OPT Features

the IBM optimizer provides a set of algorithms and policies to be plugged in and used by the Policy Engine component during its periodical optimization placement cycle. The optimier features are listed in table 2.13.

Table 2.13: OPT Features

Feature	Function
Extensibility	Supports an open-ended set of placement optimization algorithms
Versatility	Can use a variety of back-end commercial and open-source general purpose LP, MIP, and non-linear solvers as back-end solvers. Also allows for custom optimization algorithms implementation
Migration minimization	Implemented optimization policies strive at minimizing the number of VEE replacements (migrations)
Power conservation	Supports power conservation policy that consolidates VEEs on the minimal number of physical hosts; policy is activated on-demand through PE management interface
Load balancing	Supports power conservation policy that spreads the VEEs on the maximal number of physical hosts to equalize between the average load in the system and the local load; policy is activated on-demand through PE management interface
Profit maximization	All optimization policies maximize perceived placement benefit as expressed by the SLA levels of the VEEs
Set constraints	Maximizes the number of completely placed VEE sets as required by elasticity rules execution
Affinity/antiaffinity constraints	Allows to specify collocation and anti-collocation constraints for service components at the level of hosts and sites
Assignment constraints	Allows to specify constraints preventing VEEs' placement on hosts/sites such as geography-based constraints

The dependencies with other RESERVOIR components is described in table 2.14:

Dependency	Description
COIN-OR	Used in current implementation as a back-end solver
Federation heuristics	The federation heuristics decide on the site destinations for VEEs, while local optimizers at each site optimize the local placement
PE	Policy engine communicates VEEs to be placed to the optimizer and acts on the output of optimization
VMI	Through VMI the service admission control descriptor is communicated. This descriptor specifies SLA pertaining information and placement constraints for the components comprising a service.

Table 2.14: OPT dependencies

2.2.8 VEE-MIG: Migration of VMs between hosts that do not share storage

This code supports migraton of VMs between hosts that do not share storage. Modifications have been made to QEMU-kvm and libvirt. This code was developed by IBM Haifa Research Lab. But we are in the process of submitting it for open source distribution. The QEMU-kvm is scheduled to be incorporated in the QEMU release 0.12.0. The libvirt patch will be submitted once all changes are offically accept to the QEMU release.

This is shown in table 2.15.

Feature	Function
Migration	Migrations of virtual machines between hosts that do not share storage

Table 2.15: VEE-MIG Features

The dependencies with other RESERVOIR components is described in table 2.16:

Dependency	Description
QEMU-kvm	modifications have been made to KVM which is a virtualization solution for Linux with virtualization support
Libvirt	modifications have been made to libvirt which is a toolkit to interact with the virtualization capabilities of Linux

Table 2.16: VEE-MIG dependencies

2.2.9 MON Features: A Monitoring Framework for Service Clouds

This section presents Lattice, a framework that has been designed and implemented to support monitoring of Service Clouds. The section presents the main architectural elements and how they are related to each other. The implementation is discussed and we present details of how values are encoded and decoded for network transmission. Finally, we show the context within which Lattice has been deployed for real by presenting a testbed that has been created at UCL. The Lattice monitoring framework described here was devised as part of the RESERVOIR [1] FP7 project.

RESERVOIR requires a monitoring system that has a minimal runtime footprint and is not intrusive, so as not to adversely affect the performance of the cloud itself or the running service applications. As a consequence, we need to ensure that service management elements only receive data that is of relevance. In a large distributed system such as RESERVOIR there may be hundreds or thousands of measurement probes which can generate data. It would not be effective to have all of these probes sending data all of the time, so a mechanism is needed that controls and manages the relevant probes.

For full operation of a RESERVOIR service cloud we need to remember that monitoring is a vital part of the full control loop that goes from the service management, through a control path, to the Probes which collect and send data, back to the service management which makes various decisions based on the data. The monitoring is a small but fundamental part of RESERVOIR as it allows the integration of components in all of the layers.

Monitoring is a fundamental aspect of a service cloud such as RESERVOIR because it is used by the infrastructure itself and for service management. The monitoring system needs to be pervasive as:

- it is required by most of the components of the service cloud;
- it cuts across the layers of the cloud system creating vertical paths; and
- it spans out across all the service clouds in a federation in order to link all the elements of a service.

This is shown in table 2.17.

Feature	Function
Data source probes	each data source can have multiple probes, with each probe returning its own data.
Flexible probes	data sources are be able to turn on and turn off probes, or change their sending rate
Fully dynamic data sources	ability to add new probes to a data source at run-time.

Table 2.17: MON Features

The are no dependencies with other RESERVOIR components.

2.2.10 SEC-AC Features

The SEC-AC software provides access control for the SMI and VMI interfaces. The access control architecture provices access control for service providers accessing the service manager, service manager administrators accessing the service manager. For the service providers the objective is to provide role based access control for the service lifecycle. The access control implementation is based on SUN XACML implementation. This is an open source implementation of the OASIS XACML standard5, written in Java(TM) programming language. XACML is an initiative to develop a standard for access control and authorization systems. A typical access control request includes three main entities: the subject that request a permission to perform an action on a resource, the recource on which the subject will execute an action, and the action that the subject will execute on the resource

This is shown in table 2.18.

Feature	Function
access control	provides access control at the EZWEB portal and service manager API levels
OVF integrity verification	provides signture verification functions to verify that the OVF file submitted by the service provider is the one that has been uploaded to the service manager.

Table 2.18: SEC-AC Features

The dependencies with other RESERVOIR components is described in table 2.19:

Dependency	Description
EZWEB	the access control and integrity verification is integrated into the EZWEB portal
SM	the access control of the SMI is integrated with the service manager

Table 2.19: SEC-AC dependencies

2.3 Framework Software and Software Prototypes

This guide describes the RESERVOIR framework results in terms of released open source software and specifications. A more detailed description of the software prototype.A detailed description of the proto-

types is available in [2]. The correspondance between available RESERVOIR open source software and the software prototypes is described in table 2.20:

Released software	Software prototypes
EZWEB	-
SM	the ”SM” includes ”Service Definition and Lifecycle management”,
VEEM/OpenNebula	”VEEM/OpenNebula” includes ”VEE Provisioning and Supervision”, ”Federation of Management Domains” and ”SLA Management and Service Request Dispatching”
VEEH-MIG	”VEEH-MIG” corresponds to ”Relocation Enablement”
PE	the ”PE” corresponds to ”Allocation Policy Management”
OPT	”OPT” is part of ””Allocation Policy Management””
MON	”MON” is part of ”VEE Technologies”
ACC-BILL	”ACC-BILL” corresponds to ”Accounting, payment and billing”

Table 2.20: Released software and software prototypes

3 Building Clouds with the RESERVOIR Framework

This section descibes how to use some of the RESERVOIR framework and specifications to build private, hybrid and community clouds?

3.1 Configurations

A list of possible RESERVOIR configurations built from the framework software and specifications is described in table 3.1.

name.	Definition	Components	Status
PCC	Private Cloud Computing (PCC): Capabilities for the management of the private data center or cluster	VEEM/OpenNebula	Available
HCC	Hybrid Cloud Computing: Capabilities for the extension of the local infrastructure with remote Cloud resources	VEEM/OpenNebula	Available
COMC	Community Cloud: Capabilities for creating a federated cloud supporting remote provisioning and migration	VEEM/OpenNebula	Available

Table 3.1: RESERVOIR Framework Configurations

3.2 VEEM/OpenNebula private cloud

3.2.1 Features

Table 3.2 shows the features of a Private Cloud Computing configuration.

Feature	Function
Internal Interfaces for Administrators and Users	Unix-like CLI and XML-RPC API to manage VM life-cycle and physical boxes; and libvirt virtualization API
Scheduler	Requirement/rank matchmaker allowing the definition of workload and resource-aware allocation policies such as packing, striping, load-aware, affinity-aware; and support for advance reservation of capacity through the Haizea VM-based lease manager
Virtualization Management	Xen, KVM and VMware connectors; and generic libvirt connector to other VM managers. Virtual Box planned for 1.4.2
Image Management	General mechanisms to transfer and clone VM images
Network Management	Definition of isolated virtual networks to interconnect VMs
Service Management and Contextualization	Support for multi-tier services consisting of groups of interconnected VMs, and their auto-configuration at boot time
Security	Management of users by the infrastructure administrator
Fault Tolerance	Persistent database backend to store host and VM information
Scalability	Tested in the management of medium scale infrastructures consisting of hundreds of servers and VMs
Installation	Installation on a UNIX cluster front-end without requiring new services in the remote resources; and distributed in Ubuntu 9.04 (Jaunty Jackalope)
Flexibility and Extensibility	Open, flexible and extensible architecture, interfaces and components, allowing its integration with any product or tool in the virtualization and Cloud ecosystems and management tool in the data center
Cloud Interfaces for Users	Implementation of a subset of the EC2 Query API and the OGF OCCI API
Extensibility	The new OpenNebula Cloud API allows the implementation of new Cloud interfaces

Table 3.2: Private Cloud Computing Features

3.2.2 Configuration Description

This is illustrated in figure 3.1.

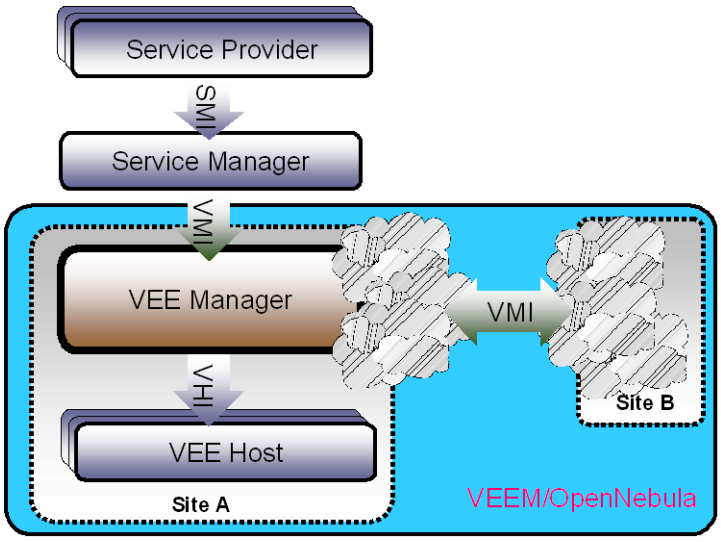

Figure 3.1: Private Cloud

3.3 VEEM/OpenNebula Hybrid cloud

3.3.1 Features

Table 3.3 describes the features of the Hybrid Cloud Computing configuration. These features extend the features of the private cloud presented in the previous setion.

Feature	Function
Cloud Plugins	Amazon EC2 and ElasticHosts connectors
Federation	Support for simultaneous access to several remote clouds
Extensibility	Modular approach to develop new connectors

Table 3.3: Hybrid Cloud Computing Features

3.3.2 Configuration Description

This is illustrated in figure 3.2.

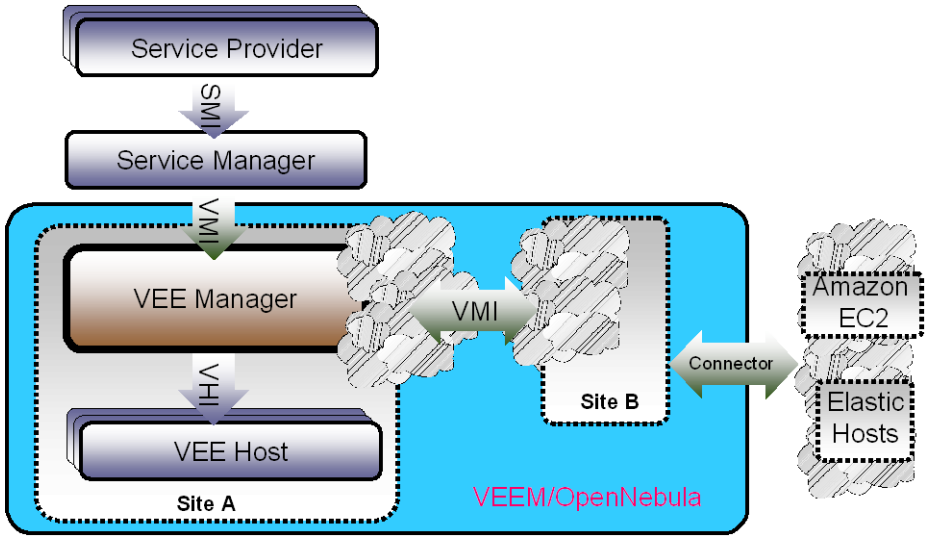

Figure 3.2: Hybrid Cloud Computing

3.4 VEEM/OpenNebula Community cloud

3.4.1 Features

Table 3.4 describes the features of the Community Cloud. They extend the features of the private and hybrid clouds.

Feature	Function
Federation	migration and provisioning between federation sites

Table 3.4: Community Cloud Computing Features

3.4.2 Configuration Description

Figure 3.3 illustrates a community cloud with a federation of two sites.

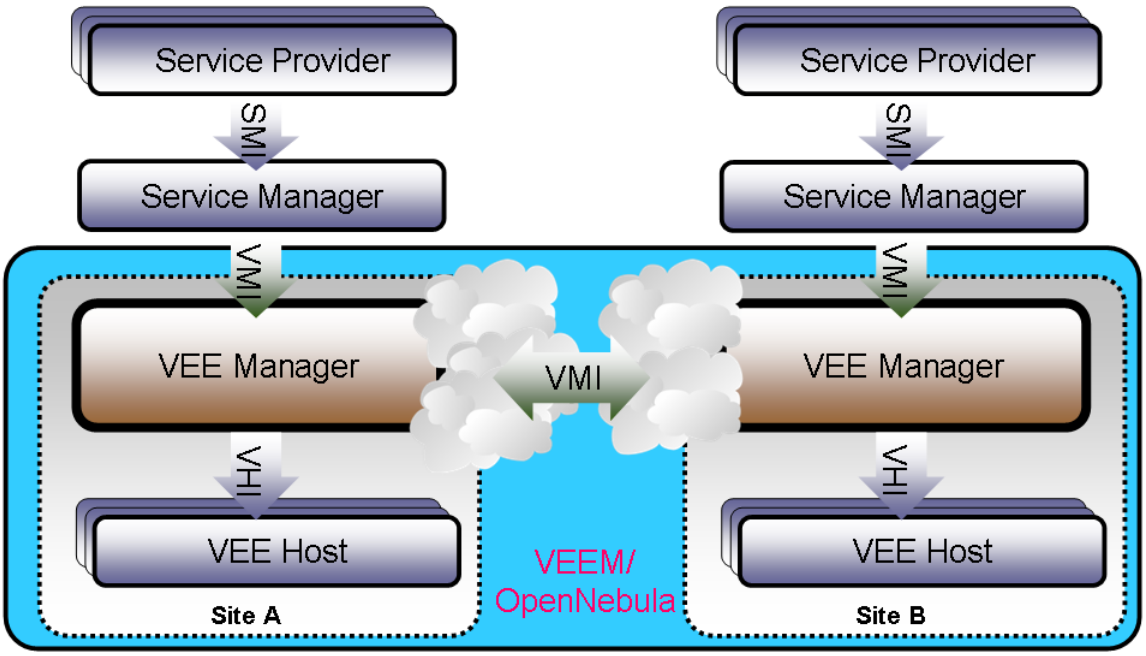

Figure 3.3: Community Cloud

4 Conclusions

This document has presented the results of the RESERVOIR european project in the form of a framework. This framework groups all the software and specifications needed to implement the RESERVOIR architecture for federated infrastruicture as a service. The document has shown how each individual piece of software could be used to implement a part of the RESERVOIR architecture. References to the code and specifications is given.

Bibliography

[1] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, *et al.*, "The reservoir model and architecture for open federated cloud computing," *IBM Journal of Research and Development*, vol. 53, no. 4, 2009.

[2] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, *et al.*, "Deliverable D1.1.1R2 - DRAFT High Level Architectural Specification Release 2.0", RESERVOIR project deliverable D1.1.1R2, http://www.reservoir-fp7.eu/.

[3] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, *et al.*, "Software Prototype Report Release 2.0", RESERVOIR project deliverable D2.1.3, D2.2.3, D2.3.3, D3.1.3,D3.2.3, D3.3.3, D4.1.3, D4.2.3, D4.3.3, http://www.reservoir-fp7.eu/.