CompatibleOne Open Source Cloud Broker Architecture Overview

WHITE PAPER

April 2012

Table of Contents

List of Figures

CompatibleOne
Open Source Cloud Broker
Architecture Overview

Abstract

In this white paper, we discuss the CompatibleOne open source cloud broker. CompatibleOne is an open source collaborative project based on open standards. The CompatibleOne approach is based on the open cloud computing interface OCCI. This white paper introduces the four-step functional manifest-to-service provisioning cycle of the CompatibleOne broker and provides an overview of its logical model and core concepts and its communication architecture.

Background

The white paper is based on a presentation given by Iain James MARSHALL of Prologue SA, at OW2con'11 on 24 Nov 2011. The video capture of this presentation includes a live demonstration and is available at: http://compatibleone.org/bin/view/Videos/.

CompatibleOne originated as a collaborative project endorsed by the Systematic Competitive Cluster and supported financially by the Paris Region. The partners in the CompatibleOne project include ActiveEon, Bull, City Passenger, Enovance, Eureva, Mandriva, Nexedi, Nuxeo, Prologue, Xwiki, Inria, Institut Telecom and OW2.

Disclaimer

The purpose of this white paper is to provide a technical overview of the CompatibleOne Open Source Broker architecture with specific attention given to the positioning of new concepts. This paper does not provide an exhaustive discussion of the CompatibleOne Open Source Broker and is not a competitive analysis between the CompatibleOne Open Source Broker and other cloud computing interoperability mechanisms.

Introduction

CompatibleOne was launched as a collaborative project to perform research work and to come up with ideas addressing the need for interoperability in the field of Cloud Computing. The project quickly evolved until it converged in developing a cloud computing broker as defined by Gartner[1], i.e. a core cloud computing mechanism providing Intermediation, Aggregation and Arbitration of cloud services. Moreover, the CompatibleOne platform is aligned with the Cloud Computing Reference Architecture[2] of the National Institute of Standards and Technology (NIST, U.S. Department of Commerce).

This white paper presents an overview of CompatibleOne the first open source cloud computing broker. It is blueprint for Developers, Operators and DevOps. In CompatibleOne, models and documents are shared by developers and operators, customers and vendors in the form of a Manifest. CompatibleOne provides support for interoperability, portability and reversibility.

Section A: CompatibleOne: Open Standards and Open Source

CompatibleOne is an open source project. OW2, an independent open source organization, is one of the project's partners. The work performed is published under open source license on the OW2 forge.

The CompatibleOne project is based upon the strong belief that open source and the adoption of open standards enable open innovation and foster an open cloud computing ecosystem.

For this reason, CompatibleOne is driven by open standards and open source. The project partners have identified a series of open standards that currently provide the backdrop of the project. These include the following five standards:

- Distributed Management Task Force (DMTF)
- Cloud Data Management Interface (CDMI)
- Storage Networking Industry Association (SNIA)
- Open Cloud Computing Interface (OCCI)*
- Open Grid Forum (OGF)

*A special mention goes to OCCI, the simplicity and powerful potential of which inspired the attention of all project partners.

Developed in close coordination with these different bodies, the CompatibleOne project partners are now working to find out how to leverage these standards and specifications in building a cloud stack blueprint.

1 Gartner, Defining Cloud Service Brokerage: Taking Intermediation to the Next Level, by Daryl C. Plummer, Benoit J. Lheureux, Frances Karamouzis, 8 October 2010, G00206187
2 http://www.nist.gov/customcf/get_pdf.cfm?pub_id=909505

Section B: CompatibleOne Functional Cycle

In Figure 1 below, we provide a high-level overview of the CompatibleOne platform architecture in four quadrants. These quadrants represent the four steps of the functional cycle of the platform and are also consistent with the four agents or cloud computing stakeholders identified in the NIST report.

Step One: Handling the user's requirements

The first step deals with the user interactivity. Here users of the platform specify their requirements in terms of the CompatibleOne Resource Description Schema (CORDS) document, describing the infrastructure they require. It is a service manifest document which describes in detail the services to be delivered, the technical and economical criteria, and specifications and constraints that are to be taken into consideration. The outcome of step one is the Manifest.

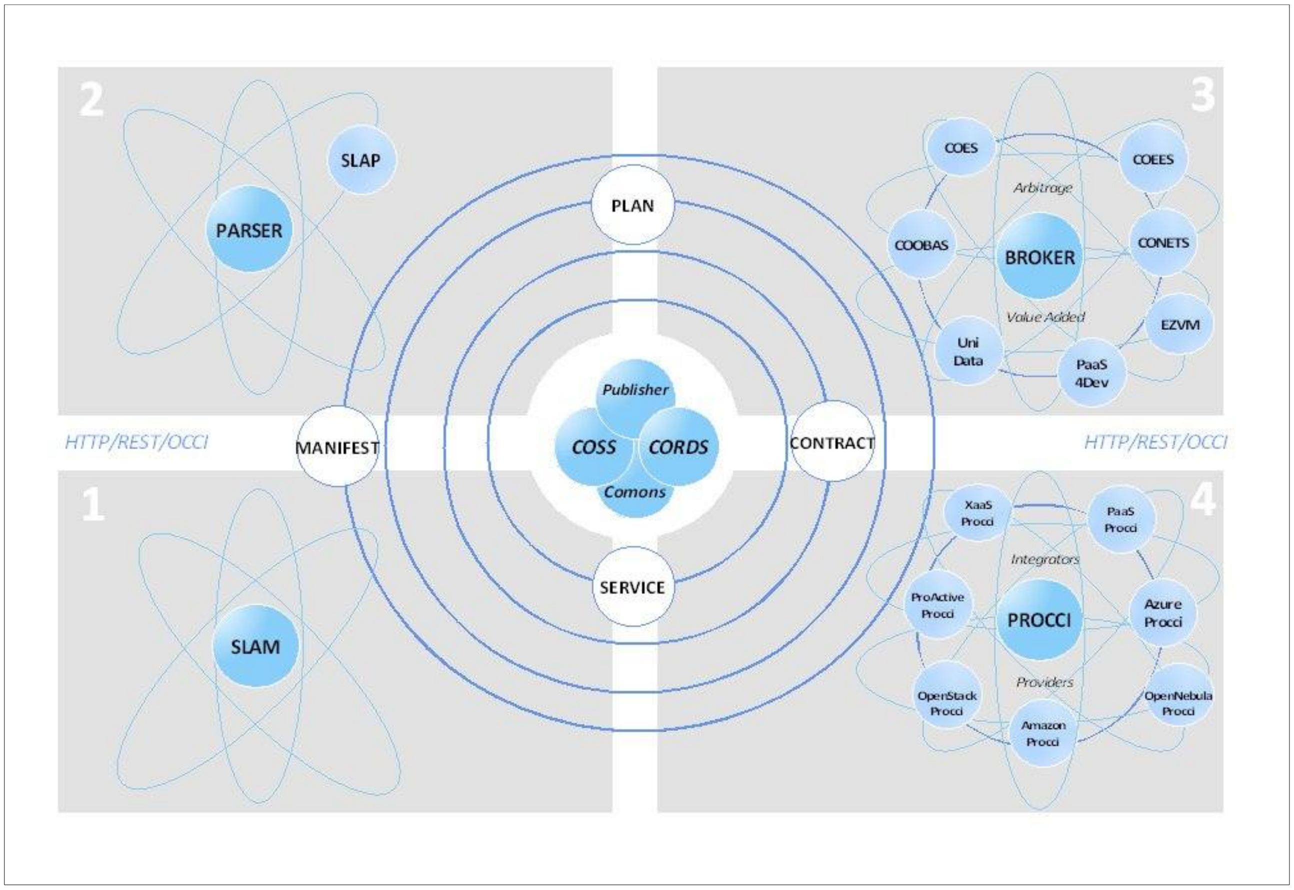

Figure 1: CompatibleOne Architecture v2.10

Step Two: Validation and provisioning plan

In the second step, the manifest is received by the SLAP MASTER allocation engine which initiates the transactional event management, destined to be used by the

accountancy and billing system. The manifest is then transferred to the CORDS Parser which is responsible for parsing of the manifest and validating the XML syntax, the Schema conformance and the Infrastructure feasibility
For feasibility, the CORDS Parser will analyze the document and localize service providers within their CORDS platform community capable of understanding and satisfying the terms described within the manifest. If this process completes successfully, the XML description will be transformed into a fully qualified and validated provisioning plan. The Provisioning plan is like a blueprint, containing information required to achieve the infrastructure described by the user or initiator within the manifest.

Step Three: Execution of the provisioning plan

Once the plan has been validated it is made available to the CORDS Broker, responsible for control of the actual provisioning operation. The CORDS Broker is accompanied by a collection of CompatibleOne service providers. The non exhaustive service list shown in the third quadrant represents the collection of service components which are currently being developed within the CompatibleOne project group. They include the following eight services:

- COOBAS (CompatibleOne Ordering Billing and Accounting Services)
- COES (CompatibleOne Elasticity Scalability Services)
- COMONS (CompatibleOne Monitoring Services)
- CONETS (CompatibleOne Network Services)
- COEES (CompatibleOne Energy Efficiency Services)
- EZVM (VM Creation of System Services)
- UniData (Unification of Data storage and block storage for the EZVM)
- PaaS4Dev – OSGi Interface Management

The above services assist the CORDS Broker when performing the operations described by the plan and required for the creation of the control graph. This graph provides the framework through which the management of the individual components resulting from all subsequent provisioning operations will be performed.

Step Four: Delivering the cloud services
In the fourth step, under the guidance of the Provisioning Control Graph, the Broker negotiates and engages the resources to be provisioned through the CORDS Procci. The CORDS Procci enables the Broker to negotiate the placement and provisioning of the resources required to satisfy the needs expressed within the Manifest and qualified by the plan. This will be delivered to the user in the form of the Service.
Communication with the different Carriers is performed through client/server interfaces described using standard OCCI specifications and implemented on the CompatibleOne side within the CORDS Procci and on the Carrier side within a Carrier-specific Procci component. CompatibleOne is currently capable of provisioning on both OpenNebula and OpenStack platforms. Other Procci components being developed include SlapOS, ProActive, Amazon and Windows Azure and DeltaCloud*.

*It is important to note that the proxy defined by CompatibleOne is an OCCI proxy because the entire architecture has been built on standard, loosely coupled, OCCI REST API interfaces. Consequently every single component communicates with the other components though these REST OCCI interfaces.

Section C: Communication Support Facilities

A CompatibleOne service bus does not exist. Many options with service buses were being evaluated until it became apparent that service buses were not necessary and were redundant.

However, the following three Communication Support Facilities provide support to all interactions:

- **Knowledge Base:** A collection of information maintained within the system. The elements stored in the Knowledge Base are the Manifests, the Plans, the Contracts and the Services. A unique and universal identifier is attributed to each element in the Knowledge Base and may be used to reference the element for use in subsequent plans.
- **CompatibleOne Security Services (COSS):** A highly security conscious system where everything is performed using Transport Layer Security (TLS). Each component in the system is required to obtain and present its own identity. Each component must also be authenticated and authorized to use the system.
- **Publisher:** Provides publication services, or repository services, for the connection between the components.

All interactions within ACCORDS (Advanced Capabilities for CompatibleOne Resource Distribution Services) are performed in conjunction with the Publication Service offered by the Publisher, in compliance with COSS (CompatibleOne Security Service) and using OCCI (Open Cloud Computing Interface) over HTTP (Hyper Text Transfer Protocol). In the end, this collection of protocols and specifications constitutes the Cloud middleware offered by ACCORDS.

Section D: CompatibleOne Logical Model

The CompatibleOne approach is based on the open cloud computing interface OCCI, which provides a simple approach for the description of almost anything. The OCCI OGF Group defines an Entity that is derived to give Resources, Links and Mixins. Below that, the entity defines three basic elements: Network, Compute and Storage to which have been added StorageLink and NetworkLink. The entire CompatibleOne resource description schema is compliant with this architecture.

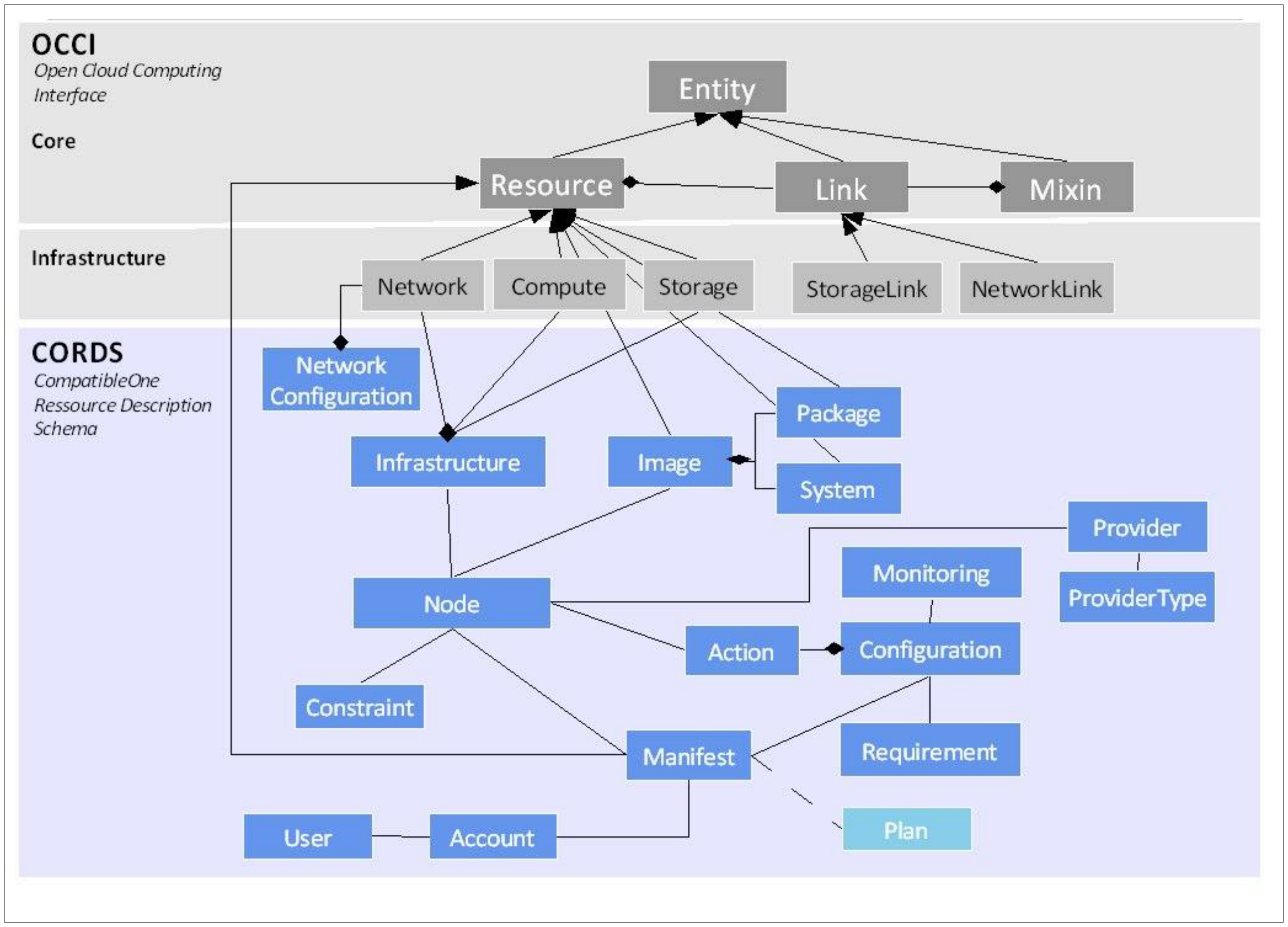

Figure 2: CompatibleOne Description Model Logical View

In Figure 2, which shows the logical view of the taxonomy or nomenclature, each of the blue boxes represents a Category and each of these Categories is a resource. CompatibleOne's rationale is completely compliant with an OCCI Interface and an OCCI Description Understanding Mechanism. In the OCCI Protocol there is a capabilities discovery request that allows any client to discover the categories of services these OCCI servers are capable of delivering. The illustration shows the logical descriptive view of how the Manifest is organized internally to give rise to the Plan.

A Manifest is described in terms of Nodes. The Node is defined as a unit, which is described in terms of its Infrastructure and application Image. Its Infrastructure provides its Network, Compute and Storage requirements and is described using the standard OCCI Category elements. The Image is described in terms of its base operating system and the collection of packages that are required to be installed to achieve the desired application functionality.

A Manifest can be composed of as many Nodes as required to provide the right functionality. Each Node may suggest a particular type of provider. By type of provider, a Node may require specific services available only within, for instance, an OpenNebula platform, or an OpenStack platform or a ProActive platform with a section script.

If required, the Node may state its needs explicitly. It may however suggest that “any will do”, in which case the overall configuration of the platform will be allowed to apply its preferences.

For a Node there is also the notion of Configuration and Monitoring actions. These actions describe the way a Node will co-operate with other Nodes within the Manifest. They also describe the requirements and the constraints which are placed upon it in this operating scenario.

Section E: CompatibleOne Description Model

The processing of a manifest gives rise to a Plan. The Plan is produced and corresponds to a particular Manifest to which is attached a pricing scheme. The Plan, when it is processed by the Broker, will give rise to a Service. Several instances of Service may be produced from and linked to a single Plan. Each Service comprises the collection of Contracts required for the representation of each of the individual Nodes expressed in the Manifest. Attached to the Contracts are the collections of instructions required for its Configuration and to establish the necessary conditions for Monitoring. The Contract may provide and make use of the user subscription account for a particular provider on which the Service is to be deployed.

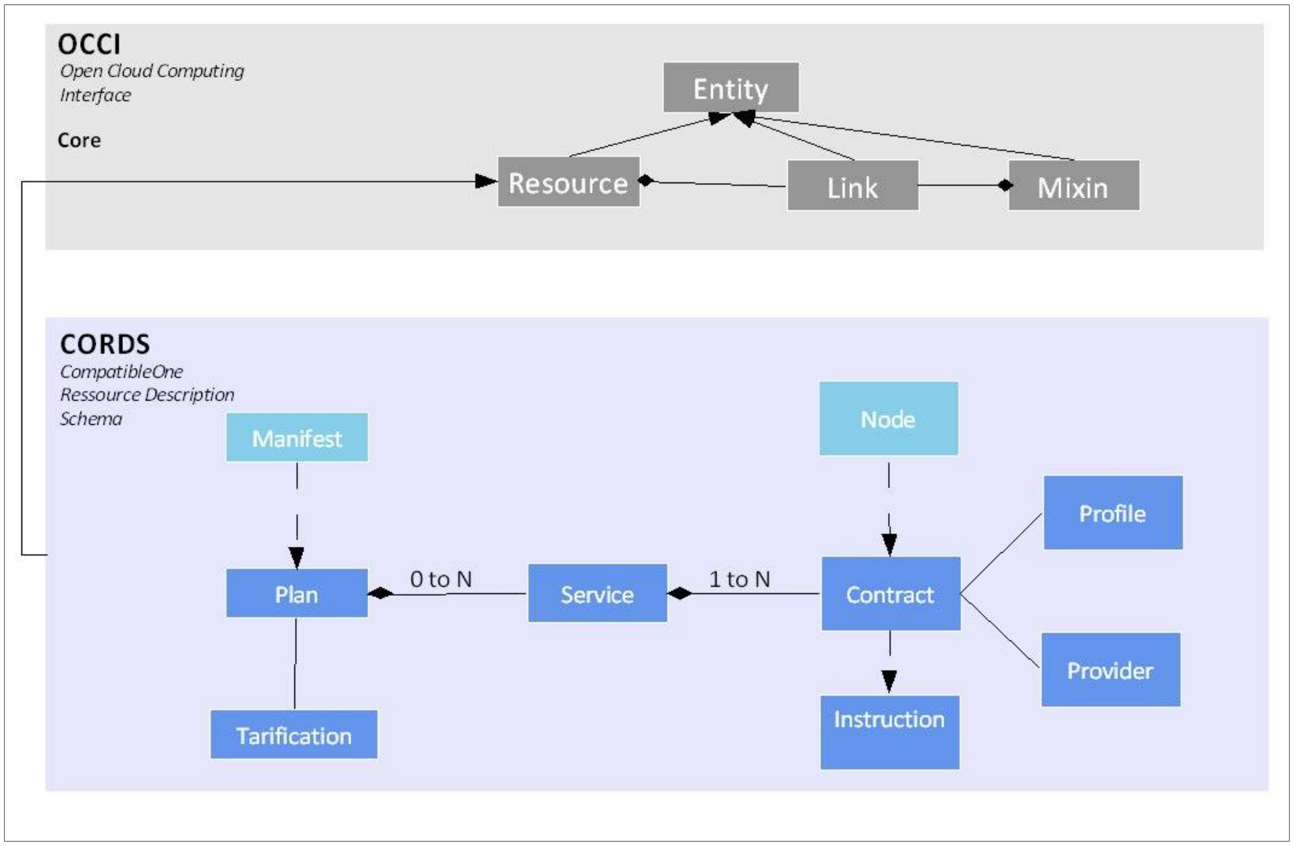

Figure 3: CompatibleOne Description Model (Virtual Instance View)

Section F: CompatibleOne Communication Architecture

Figure 4 shows the CompatibleOne Communication Architecture. The User submits a CORDS request to the Parser and Broker. The Broker engages the Service consumers which in turn discover the Service providers through the central Publisher.

The Service consumers communicate using their REST OCCI interface with individual providers to achieve the desired results. The monitoring agent under the control of a monitoring collector, or consumer, collects the various audit and monitoring data in order to ensure that the provisioning is correctly performed and correctly operated in compliance with the terms described in the Manifest or the SLA.

Figure 5 shows the schematic for the OpenStack Nova or OpenNebula provisioning. The User instantiates a CORDS plan via the Broker. For OpenStack, the Broker communicates with the Publisher and the proxy to look for an OpenStack provider. It performs the required provisioning through the REST interface. It happens in the same way with OpenNebula. The OpenNebula provisioning system is perfectly symmetrical. The only difference is the change in the nature of the final proxy, i.e. the proxy component that is responsible for handling the communication with a particular platform.

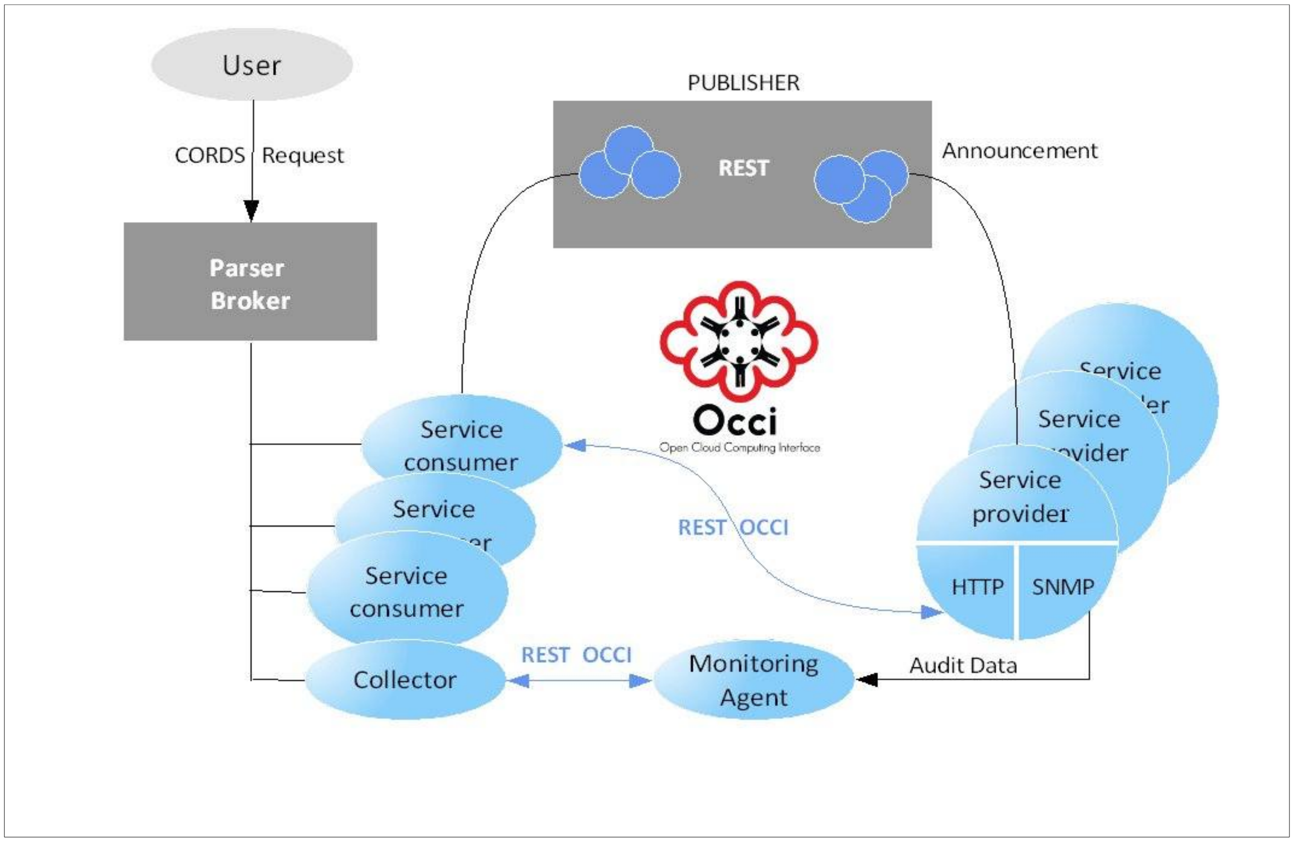

Figure 4: CompatibleOne ACCORDS (Communication Architecture)

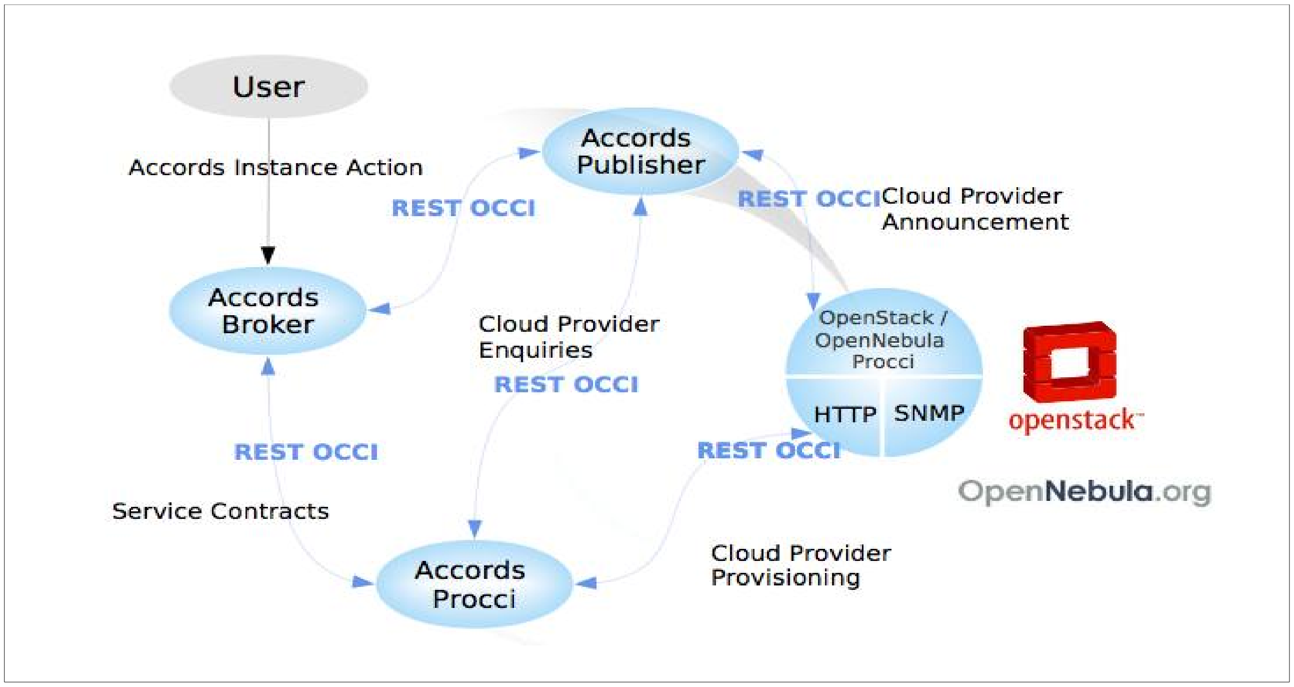

Figure 5: CompatibleOne ACCORDS (OpenStack or OpenNebula Provisioning)

Section G: Next Steps

Having achieved proof of concept, the CompatibleOne project partners have already announced automatic provisioning and automatic configuring. The configuration of Cloud Computing resources will be the next step. This will integrate the new components of the brokering system in terms of monitoring, where the blueprint and primary work has been performed. Meanwhile, the coding phase is in full swing. Security has already been defined and coded. Financial and accountancy control channels have been defined and developed for the circulation of monetary transactions resulting from the different operations performed by the platform. Further work is underway for the elaboration of the financial controls that will guide the system in order to provide intelligent provisioning respecting financial, energetic and legal constraints and conditions.

The CompatibleOne open source broker will further be validated with three use cases:

- Extensions for use of PaaS Services (providing Elasticity, Scalability and Storage)
- Cloud Automation – whereby the manifest will be used to provide automated instancing on demand by business processes
- Provision of HPC resources and 3D visualization (a highly interesting proof of concept whereby Graphic Processor units will be called to operate within high performance grids and provide heavy duty calculation units for 3D visualization for gaming platform delivery).

Further information

To learn more about the CompatibleOne open source cloud broker, please visit:
http://www.compatibleone.org

To join the conversation on our developer community, you are welcome to subscribe to:
http://mail.ow2.org/wws/subscribe/compatible-dev_contrib

The video capture of the CompatibleOne Open-source Cloud Broker Quick Demonstration, upon which this white paper is based, includes a live demonstration and is available at:
http://compatibleone.org/bin/view/Videos/