Vendor Agents for IAAS Cloud Interoperability

24 Sept 2012

Alba Amato, Luca Tasquier, Adrian Copie

Univerza *v Ljubljani*

Outline

- **Introduction**
  - Problems
  - Cloud Agency
  - The mOSAIC project
- **Vendor Agents**
  - Amazon EC2 vs Open Nebula
  - Vendor Agent for EC2
  - Vendor Agent for ON
- **Conclusion**

Introduction

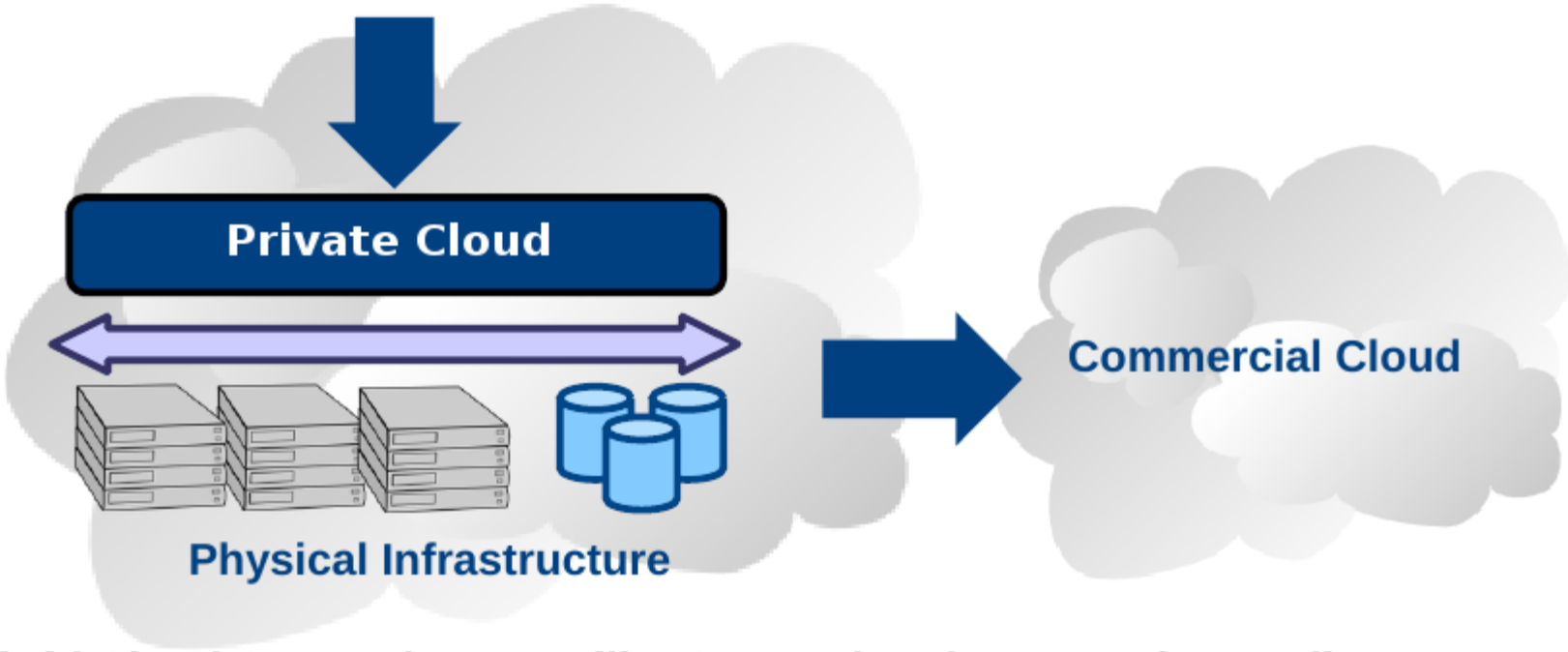
- Provisioning and management of resources at Cloud infrastructure level (IAAS)
- with particular attention to ***problems of interoperability***
- and ***portability*** arising from the ***heterogeneity of technology***

Problems

- **Heterogoeneous providers**
  - Technology
  - Service offers
- **Lock-in problem**
- **Lack of standards**

Private Cloud Computing => A "Public Cloud behind the firewall"

- Simplify and optimize internal operations
- Service flexibility and elasticity
- Higher utilization & operational savings
- Security concerns



Hybrid Cloud Computing => Utility Computing dream made a reality!

- Supplement the capacity of the Private Cloud

Lack of Interoperability

- Different technology
- Different service interface
- Different service semantics
- Different service offer
- Different service levels
- Different terms of services

Amazon EC2 vs Open Nebula

	Amazon	OpenNebula
Offered Resources	Computation, Storage, Network, Communication, Database	Storage, Network, Computation
Image Supported Operations	Bundle, Upload, Download, Delete	Upload, Publish, Unpublish, Allocate, Enable, Disable, Delete
VM Supported Operations	Run, Stop, Start, Reboot, Terminate	Start, Stop, Suspend, Reboot, Resume, Hold, Release
Access VMs	REST, SOAP, Java API	REST (OCCI compliant), EC2 compliant API, XML-RPC API, Ruby API, Java API
Security	IP ranges, Identity and Access Management	XML-RPC or by delegating to external modules (SSH, X.509)
SLA Negotiation	No	No
Instance type	Fixed Number	Depends on provider

The mOSAIC project

mOSAIC: Open-Source API and Platform for Multiple Clouds

Intends to improve the state-of-the-art in Cloud computing by creating, promoting and exploiting an open-source Cloud application programming interface and a platform targeted for developing multi-Cloud oriented applications.

- **An API**
  - Open Source and platform-independent API
  - Extends the existing language-or platform-dependent API capabilities with composite features
- **A framework**
  - Semantic engine
  - Cloud agency
- **An open-source platform**

Cloud Agency

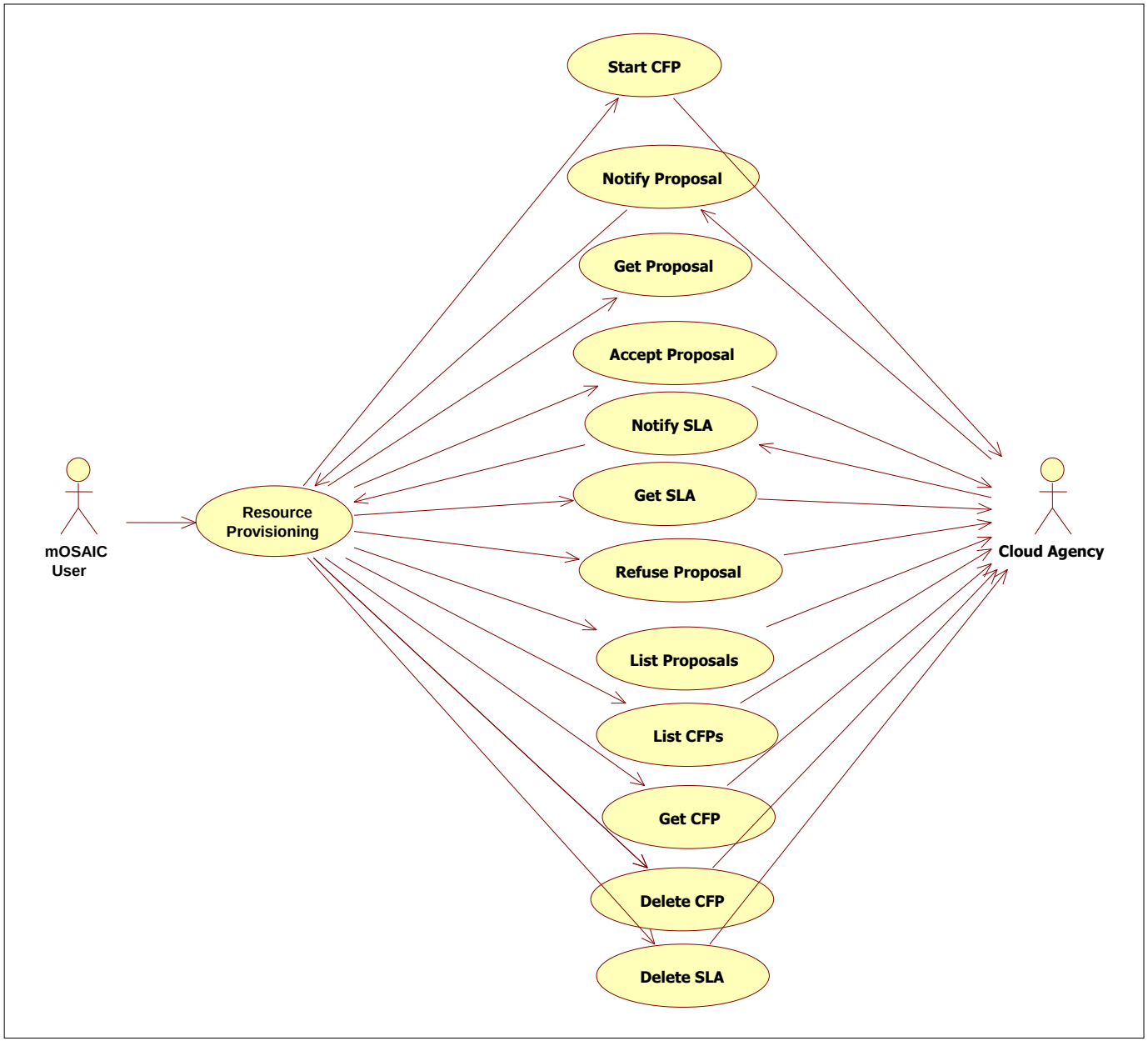
- **Cloud Agency is a Provisioning System**
  - IAAS Dynamic Provisioning
  - IAAS Interoperability
  - Provider independent IAAS Monitoring
  - Autonomic IAAS reconfiguration

The main task is to dynamically select a set of Cloud resources, from different vendors, that best fit the users requirements

The Vendor Agent Abstraction

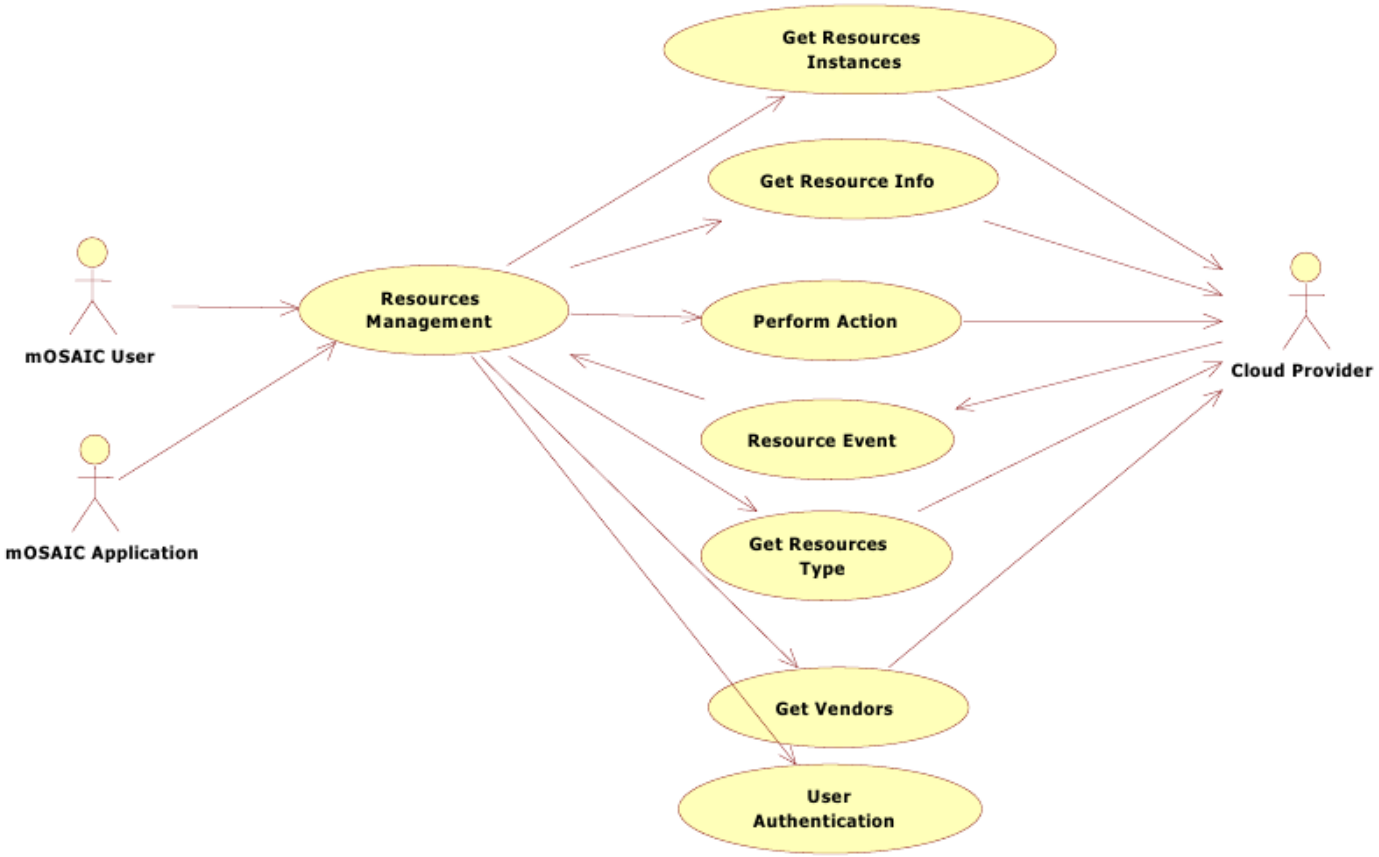
- **Is a wrapper to a specific provider technology**
- **Implements the Provisioning Use Case**
- **Implements the Management Use Case**

Provisioning

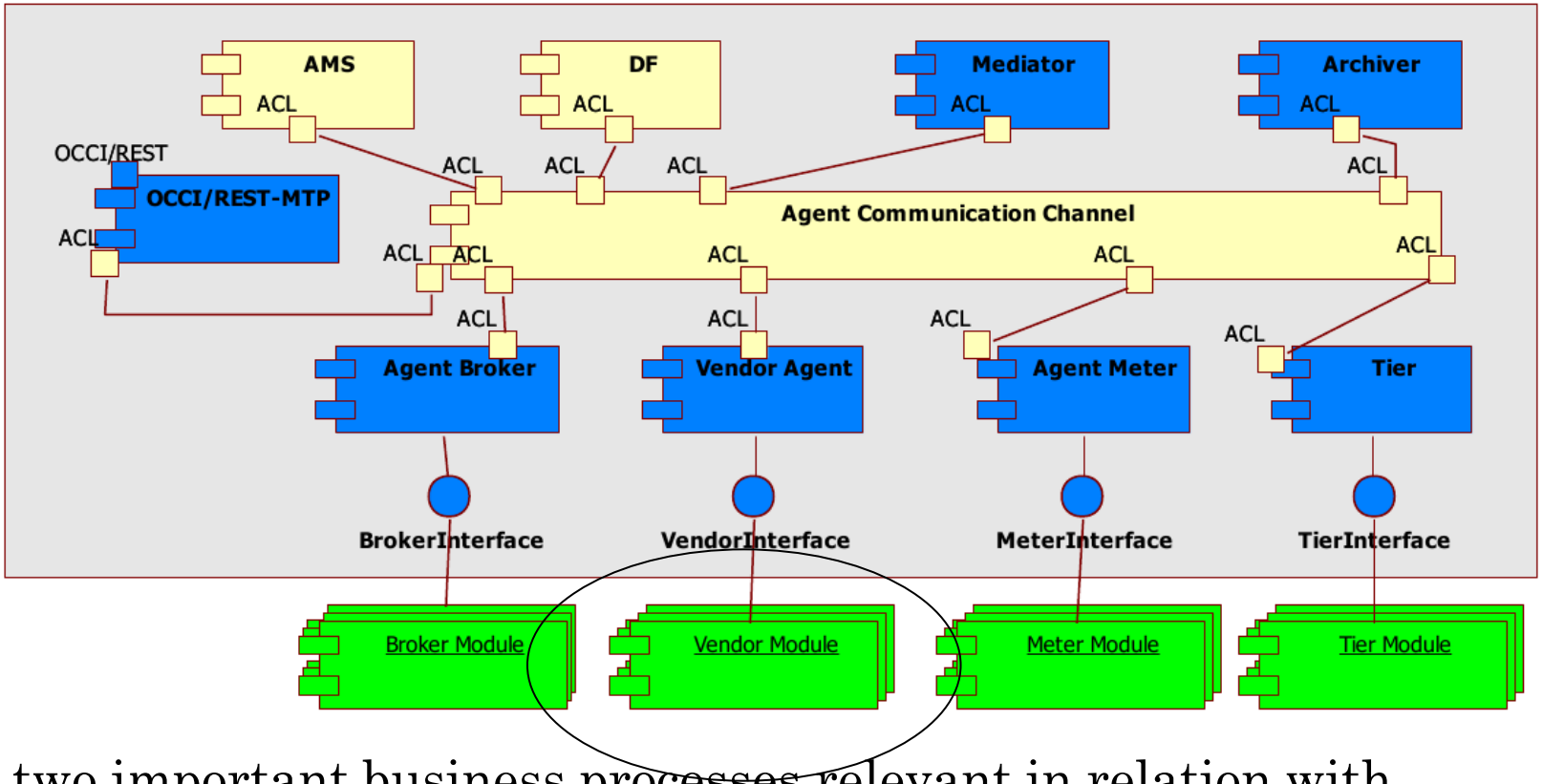


Management

Allows operations such as the creation or the deletion of cloud resources but also to perform a set of operations on the created resources.

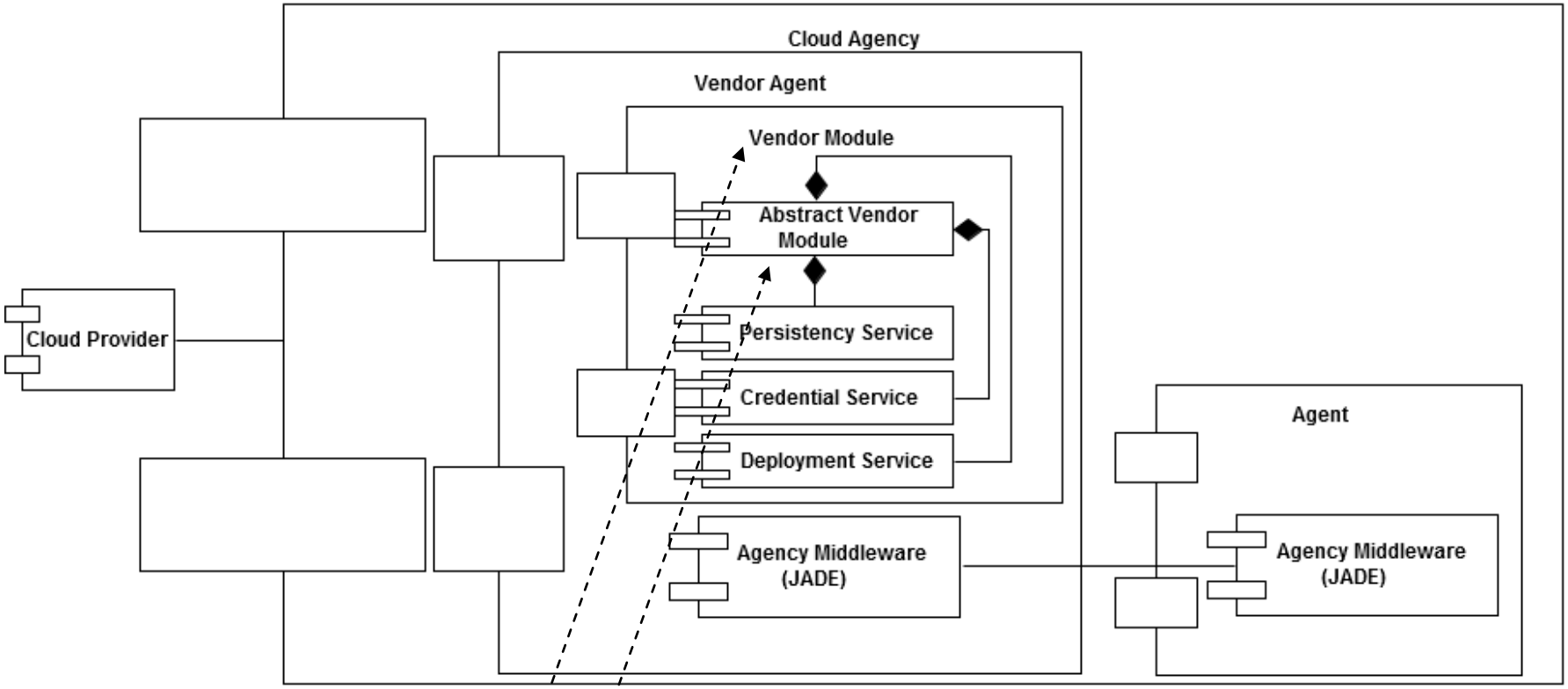


Cloud Agency Components Diagram



two important business processes relevant in relation with the vendor agents are: ***the resource provisioning and the resource management.***

The Vendor Agent



The ***Vendor Module (VMod)*** component is intended to cover the cloud provider specifics.

The Vendor Module extends the functionality of an ***Abstract Vendor Module (AVMod)*** which is intended to implement the common aspects on each provider.

The Abstract Vendor Module

- The resource provisioning operations provided at the level of the AVMod are:
  - submitCFP: accepts a CFP and delegates to the VMod the servicing;
  - onProposalAvailable: callback for VMod in order to signal the availability of a proposal;
  - onCFPReject: callback for VMod to signal CFP rejection because of not being able to create a proposal;
  - deleteProposal: the client specifies that a proposal is no longer interesting and should be discarded;
  - acceptProposal: the client accepts a proposal and thus become an SLA. It also provides a deployment description which allows the agent to manage the proposal's resources;
  - refuseProposal: the client refuses a specific proposal.

The Vendor Module

- The operations related to the resource management includes requests coming from the clients which will be implemented at the level of VMods:
  - createResource: creates a resource of a specified class for the specified SLA. Resources can be created at any moment;
  - releaseResource: releases a previously created resource;
  - performAction: performs a specified action on a resource.

- There are also some callbacks being provided for the VMods:
  - onResourceCreation: signals the client that a resource was created and information about it is available;
  - onResourceRelease: confirms the resource release to the client;
  - onResourceActionPerformed: confirms an action on a resource;
  - onResourceStatusUpdate: updates the client about one resource's status.

Vendor Agent for Open Nebula

- Developed using the Java API
  - compatibility : Cloud Agency is written in Java
  - the Java API is a wrapper of the XML-RPC so the communication between the Vendor Module and the OpenNebula core can occur without any other management operation.
- overridden methods are:
  - submitCfp: the module receives the Call For Proposal, translates it in a OpenNebula compliant format and queries the OpenNebula core to retrieve the possible offers. After that it compares the received offers against the Call For Proposal and chooses the one that best fits with it.
  - acceptProposal: retrieves the accepted offer ID; the proposal; the user's credential to access the OpenNebula account.
  - performAction: recovers the selected resource by using the resource's ID. After that it uses the passed action and parameters to translate the agnostic request into the operations which lead to the compliance with the request. If the action succeeds, the Vendor Module notifies this event to the Cloud Agency via performed action method. Otherwise it notifies a message to the Cloud Agency by using the performAction Failure method

Vendor Agent for Amazon EC2

- It is described in a class called AmazonVendorModule, and it is implemented in Java;
- the typica library was used to connect to Amazon and perform the resource provisioning and resource management;
- the typica library transforms the Java calls to REST calls in the Amazon Web Services.

- Amazon Vendor performs operations described before and also some operations that are specific to Amazon:
  - doCreateResource method for resources creation,
  - doReleaseResource for resources destruction and
  - performAction for the resources management operations.;

  The specific operations supported in performAction method are stop, start and reboot the virtual machines.

  The Amazon credentials are managed outside the vendor agent

Differences between Amazon and OpenNebula Vendor Modules

	Amazon	OpenNebula
Currently Supported Actions	Create, Start, Stop, Reboot	Create, Start, Stop, Reboot, Destroy
Used Libraries	Typica	Java OCA
Credentials Management	Outside Vendor Agent	Inside Vendor Agent
Credentials Implementation	A tuple access key: secret key is used to access the Amazon resources and to perform various actions on them	A tuple username: password is used and for each operation credentials are needed
Wrapper Used	REST Calls	RPC-XML Calls

Conclusion

- We presented an agent based solution :
  - to abstract IAAS services for negotiation and management of Cloud resources that actively perform its functions within the Cloud Agency;
  - integrated within the Cloud Agency
  - and its implementation for supporting Open Nebula and Amazon cloud solutions.

The vendor agent represents a clear innovation in its ability to abstract the functionalities of provisioning and management and in its proactivity in offering the proposal and participating in the negotiation.

Thanks