Overcoming the Cloud heterogeneity: from uniform interfaces and abstract models to multi-cloud platforms

Dana Petcu
http://web.info.uvt.ro/~petcu,
petcu@info.uvt.ro

West University of Timisoara & Institute e-Austria Timisoara, Romania
Overview

- Cloud heterogeneity
  - The problem
  - The current solutions
    - Examples
    - Advantages and disadvantages
  - Proposal for a research agenda

- Case study
  - Multi-Clouds
  - Model-driven engineering
Solutions to overcome Cloud heterogeneity

New: proposal of a research agenda
Cloud heterogeneity

- Manifested in
  - the set of interfaces of the services from different Public Clouds
  - the set of services from the same provider
  - the software stacks
  - the hardware
  - the terms of performance or user quality of experience

- Favoring
  - the Cloud service providers allowing them to be competitive in a very dynamic market especially by exposing unique solutions

- Hindering
  - the interoperability between these services
  - the portability of the applications consuming the services
  - the seamless migration of legacy applications towards Cloud environments
Types of solutions

1. adoption of standards
   - existing standards
   - emerging standards

2. usage of intermediary layers
   - libraries for major programming languages
   - tools and services

3. adoption of high abstraction levels
   - semantics
   - model-driven engineering
Standards and protocols

<table>
<thead>
<tr>
<th>Type</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>OCCI, OVF, CDMI, CIMI, TOSCA</td>
</tr>
<tr>
<td>Reference architecture</td>
<td>IETF, DMTF</td>
</tr>
<tr>
<td>Open groups</td>
<td>OCC, GICTF, CSCC, Open Group</td>
</tr>
<tr>
<td>Initiatives</td>
<td>ETSI, IEEE, CSA, NIST, OASIS, OW2, SNIA, TM-Forum</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Advantages</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result of a collective agreement</td>
<td>Not widely adopted</td>
</tr>
<tr>
<td>Extract the key actions and characteristics</td>
<td>From the point of view of the providers,</td>
</tr>
<tr>
<td></td>
<td>hinders diversity</td>
</tr>
<tr>
<td>Should be implementable</td>
<td>No. emerging &amp; overlapping standard makes</td>
</tr>
<tr>
<td></td>
<td>the problem to grow</td>
</tr>
</tbody>
</table>
# Open-source libraries

<table>
<thead>
<tr>
<th>Type</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Libraries</td>
<td>DeltaCloud, fog, jclouds, libcloud, SimpleCloud</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offers an abstraction layer that is simple to use</td>
<td>Usually refers to the common denominator of the Cloud services</td>
</tr>
<tr>
<td>Available for major languages</td>
<td>Language dependent</td>
</tr>
<tr>
<td>Similarity with major Cloud provider APIs</td>
<td>Adaptors needs to be build for emerging new services</td>
</tr>
<tr>
<td>Decouple the application code from the underlying Cloud service</td>
<td>The connected service programming style usually maintained</td>
</tr>
<tr>
<td>Adaptors available for major Cloud services</td>
<td>Require Cloud computing knowledge as deployment is usually not supported</td>
</tr>
<tr>
<td></td>
<td>Introduces an overhead compared with the direct connection to Cloud service</td>
</tr>
<tr>
<td>Type</td>
<td>Examples</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Services/tools</td>
<td>Aoleus, CompatibleOne, Cloudify, ConPaaS, mOSAIC PaaS, Nimbus, OpenNebula, OpenStack, OPTIMIS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offer unique entry point for application deployment and Cloud resource management</td>
<td>The diversity of deployment services raise also another dimension for the portability</td>
</tr>
<tr>
<td>Application deployment can be done by non-Cloud specialists</td>
<td>Manual intervention at deployment phase is usually still needed</td>
</tr>
<tr>
<td>Part of them are offering also APIs for programming applications</td>
<td>Life migration is still not possible</td>
</tr>
<tr>
<td>Usually it offers support for multiple programming languages</td>
<td>Re-deployments are not automated</td>
</tr>
<tr>
<td>Monitoring tools are generate alarms needed to trigger a redeployment</td>
<td>Rely upon adaptors that need to be build for new services or updated when a service version appears</td>
</tr>
</tbody>
</table>
# Semantics

<table>
<thead>
<tr>
<th>Type</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic solution</td>
<td>UCI, mOSAIC Onto &amp;Sem.engine, Cloud4SOA, CPIM/MODAClouds</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Offers an abstraction layer that can support various customers</td>
<td>Not widely adopted</td>
</tr>
<tr>
<td>Offers viable mechanisms for common understanding of service terminology</td>
<td>The variety of taxonomies and ontologies makes the problem to grow</td>
</tr>
<tr>
<td>and actions</td>
<td></td>
</tr>
<tr>
<td>Allow the annotation of services with quality marks by externals from the</td>
<td>The overhead of semantic processing is not negligible</td>
</tr>
<tr>
<td>provider team</td>
<td></td>
</tr>
</tbody>
</table>
# Model-driven engineering

<table>
<thead>
<tr>
<th>Type</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model-driven</td>
<td>MODAClouds, ARTIST, PaaSage</td>
</tr>
</tbody>
</table>

## Advantages

- **Enhance the abstraction layers with an automation process**
- **Allow a feedback from operational modules to the design modules**

## Disadvantages

- Available tools are not yet generating code for various Clouds
- The models that are used potentially omit special features of the services
## Research agenda – short term

<table>
<thead>
<tr>
<th>Approach</th>
<th>To Do</th>
</tr>
</thead>
</table>
| **Standards** | Enhance the number of standard implementations  
Establish standards for metrics, monitoring, accounting, security  
Establish standards for machine-readable representations of services, quality, negotiations, processes |
| **Design** | Support for decision making for Cloud migration  
Introduce Modelling-as-a-Service  
Mechanism for service compositions  
Build use cases and benchmarks for Cloud portability and interop  
Define the portability degree |
| **Runtime** | Adopt open-source platforms  
Increase the use of empirical evidence of portability and interop  
Automate re-deployments |
## Research agenda – long term

<table>
<thead>
<tr>
<th>Approach</th>
<th>To Do</th>
</tr>
</thead>
</table>
| Standards | Unified policy of the service level agreements  
              Establish standards for workload and data migration  
              Reference architecture for basic components of software consuming Cloud services |
| Design    | Define re-engineering process for Cloud  
              Mechanisms for code inspections and rewriting  
              Follow a structural approach in the design of the supporting tools  
              Ensure the portability of elasticity rule engines  
              Combine automation with customization |
| Runtime   | Mechanisms for real-time migration  
              Tools for the full service cycle, including Cloud governance  
              Open-source platforms ensuring automated portability or encompassing various approaches |
Case study: Multi-Clouds

New: classification
Delivery models for Multiple Clouds

1. **Federated Clouds**
   - assumes
   - *a formal agreement between the Cloud providers*
   - service providers
     - are sub-contract capacity from other service providers
     - offer spare capacity to the federated group of providers.
   - the consumer of the service
     - is not aware of the fact that the Cloud provider he or she pays is using the services of another Cloud provider

2. **Multi-Cloud**
   - assumes that
     - *there is no priori agreement between the Cloud providers*
   - a third party (even the consumer) is responsible for the services
     - contacts the service providers,
     - negotiates the terms of service consumption,
     - monitors the fulfillment of the service level agreements,
     - triggers the migration of codes, data and networking from one provider to another.
Examples of open libraries, services, tools
To solve in Multi-Clouds

- Portability
- Resource/service selection mechanism and methodology
- Uniform APIs
- Search engines
- Automated deployment
- Service aggregator
- Governance
- ...
On-going work/ team in Timisoara

- **MODAClouds**
  - FP7-ICT-8, 2012-2015
  - Model-driven engineering for Multi-Clouds

- **SPECS**
  - FP7-ICT-10, 2013-2016
  - Security SLA management
  - Part: Monitoring

- **AMICAS**
  - RO-PNII-PCE, 2012-2015
  - Automation in Clouds

- **HOST**
  - FP7-RegPot, 2012-2014
  - HPC services in Clouds
Model-driven engineering in Clouds

MODAClouds
MODAClouds (www.modaclouds.eu)

- Integrated Project n. 318484
- October 1st 2012 – September 30th 2015
MODAClouds objective

To provide

Methods + decision support system +
+ IDE + runtime environment

to support

- High-level design
- Early prototyping
- Semi-automatic code generation
- Automatic (re)deployment
- Monitoring and self-adaptation

of applications on Multi-Clouds

with guaranteed QoS
An example

CIM
- Access rating agencies
- Order analysis
- Get stock prices
- Place order
- Wait for ack from the stock market
- OK
- Fail
- Update customer trading account

CPIM
- A Reliable Resource
- B High perf. Resource
- C Reliable Resource
- key-valued DB
- SimpleDB

CPSM
- A-1 Medium CPU Instance
- A-2 Worker Role Large
- B-1 Large CPU Instance
- B-2 Worker Role Large
- C-1 Large Memory Instance
- C-2 Worker Role Extra Large

Place order
Wait for ack from the stock market
Update customer trading account
OK
Fail

HPC 2014, Cetraro 7/10/2014
Outputs

Cloud Development Tools
IDE + MODACloudML (agnostic and QoS ready) modelling language

Flexible Multi-Cloud Apps Management, Monitoring & Operation Environment
maximizes automation with QoS Engine, Monitoring, Portability of underlying infrastructure providers (IaaS /PaaS)

Decision Support System
is a system on its own enables selection of provider at development & testing phase; and adds automation of runtime adaptation
Concepts & components maps

Details in the public deliverable D3.2.1
Shared models

Details in the public deliverable D3.2.1
Modelio as IDE

- Functional Modelling Tools
  1. Integrating metamodels and tool chain
  2. Supporting Model validation
  3. Supporting Model transformations
  4. Reverse engineering of legacy applications
  5. Requirements traceability
  6. Code and document generation
Monitoring & statistical data analyzers
LINE:
Performance analyzer

LINE
Scalable queueing network model solver

Project Home Wiki Issues Source
Summary People

Project Information
Project feeds
Code license New BSD License
Labels
Modeling, Performance, Queuing
Members
cgl02@gmail.com, quantum@gmail.com

Line is a solver for queueing network models based on ordinary differential equations. It leverages the concept of mean-field theory, in which the interactions among a large population of entities are approximated by a single averaged effect. Line can be integrated with the Palladio Bench suite used for performance analysis of Palladio Component Models (PCM).

Compared to existing PCM performance analysers, Line can compute analytically percentiles of response times, which are important for SLA assessment. Furthermore, it can describe uncertainty about an operational environment using a modelling abstraction known as random environment.

Releases
The last and previous releases of LINE can be found here.

Documentation
All the documentation of Line can be found on this page.

How to cite Line
To cite Line, please refer to the following article:
CloudML

domain-specific modelling language + run-time environment that facilitate the specification of provisioning & deployment
mOSAIC: Run-time platform for Multi-Clouds
Where to find details

1. **Concepts:**
   - Public deliverables
   - Scientific Publications
   - White paper

2. **Software:**
   - Open-source components
   - Demos videos

...all on www.modaclouds.eu
Papers behind this presentation

- **Research agenda:**
  - Journal paper under evaluation

- **Multi-Clouds:**

- **MODAClouds:**
Q & A ?

From Cloud humour site

From Cloud humour site