soCloud: distributed multi-cloud platform for deploying, executing and managing distributed applications

Fawaz PARAISO
PhD Defense

Advisors: Lionel Seinturier, Philippe Merle

University Lille 1, Inria, SPIRALS research team
Cloud computing in nutshell

Virtualization

On-demand

Pay-per-use

Elasticity
Context and motivation

Application fil rouge

Go to the Cloud!
So many problems!

Vendor Lock-in

Geo-location

Cloud-specific services

Failures
solutions: Multi-Cloud

Do not put all your eggs in one basket

Multi-Cloud

Why not applying this precept of caution for cloud computing?
What is Multi-Cloud?

Definition

• **Multi-Cloud Computing**
  ✷ using multiple cloud providers
  ✷ independent
  ✷ no agreement between providers
Context and motivation

Enterprise Cloud Strategy

- Multi-Cloud: 74%
- Single public: 13%
- Single private: 9%
- No plans: 4%
- Multiple private: 11%
- Multiple public: 15%
- Hybrid cloud: 48%

Source: RightScale 2014 State of the Cloud Report

74% Enterprises have a Multi-Cloud strategies
Multi-cloud is supposed to be the solution but...
1. Context and motivation

2. Challenges

3. State of the art

4. Contributions
   4.1. soCloud Model
   4.2. soCloud Platform

5. Validation

6. Conclusion & Perspectives
Challenges

Multi-cloud Portability

Multi-cloud Provisioning

Multi-cloud Elasticity

Multi-cloud High-Availability
Write once, deploy anywhere without any modification
<table>
<thead>
<tr>
<th>Challenges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-cloud Portability</td>
</tr>
<tr>
<td>Multi-cloud Provisioning</td>
</tr>
<tr>
<td>Multi-cloud Elasticity</td>
</tr>
<tr>
<td>Multi-cloud High availability</td>
</tr>
</tbody>
</table>

Resources

App Deployment
Challenges

Multi-cloud Portability

Multi-cloud Provisioning

Multi-cloud Elasticity

Multi-cloud High availability

Resources

Provisioning

CloudBees

Provisioning

EC2

Provisioning

Windows Azure
Challenges

Multi-cloud Portability  Multi-cloud Provisioning  Multi-cloud Elasticity  Multi-cloud High availability

App deployment
Challenges

Multi-cloud Portability

Multi-cloud Provisioning

Multi-cloud Elasticity

Multi-cloud High availability
Challenges

- Multi-cloud Portability
- Multi-cloud Provisioning
- Multi-cloud Elasticity
- Multi-cloud High availability

[Diagram showing multi-cloud connectivity with a server labeled Windows Azure being crossed out]
Challenges

- Multi-cloud Portability
- Multi-cloud Provisioning
- Multi-cloud Elasticity
- Multi-cloud High availability
Outline

1. Context and motivation
2. Challenges
3. State of the art
4. Contributions
   4.1. soCloud Model
   4.2. soCloud Platform
5. Validation
6. Conclusion & Perspectives
## State of the art

<table>
<thead>
<tr>
<th>Approach</th>
<th>Layer</th>
<th>Multi-Cloud Portability</th>
<th>Multi-Cloud Provisioning</th>
<th>Multi-Cloud Elasticity</th>
<th>Multi-Cloud High-Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>moSAIC</td>
<td>PaaS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>STRATOS</td>
<td>IaaS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MODAClouds</td>
<td>PaaS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CompatibleOne</td>
<td>IaaS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cloud4SOA</td>
<td>PaaS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## State of the Art

<table>
<thead>
<tr>
<th>Approach</th>
<th>Layer</th>
<th>Multi-Cloud Portability</th>
<th>Multi-Cloud Provisioning</th>
<th>Multi-Cloud Elasticity</th>
<th>Multi-Cloud High-Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>moSAIC</td>
<td>PaaS</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>STRATOS</td>
<td>IaaS</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>MODAClouds</td>
<td>PaaS</td>
<td></td>
<td></td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>CompatibleOne</td>
<td>IaaS</td>
<td></td>
<td></td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>Cloud4SOA</td>
<td>PaaS</td>
<td></td>
<td></td>
<td></td>
<td>-</td>
</tr>
</tbody>
</table>
## State of the art

<table>
<thead>
<tr>
<th>Approach</th>
<th>Layer</th>
<th>Multi-Cloud Portability</th>
<th>Multi-Cloud Provisioning</th>
<th>Multi-Cloud Elasticity</th>
<th>Multi-Cloud High-Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>moSAIC</td>
<td>PaaS</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>STRATOS</td>
<td>IaaS</td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>MODAClouds</td>
<td>PaaS</td>
<td></td>
<td></td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>CompatibleOne</td>
<td>IaaS</td>
<td></td>
<td></td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>Cloud4SOA</td>
<td>PaaS</td>
<td></td>
<td></td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>soCloud</td>
<td>PaaS</td>
<td></td>
<td></td>
<td>+ +</td>
<td></td>
</tr>
</tbody>
</table>
## State of the art

<table>
<thead>
<tr>
<th>Approach</th>
<th>Layer</th>
<th>Multi-Cloud Portability</th>
<th>Multi-Cloud Provisioning</th>
<th>Multi-Cloud Elasticity</th>
<th>Multi-Cloud High-Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>moSAIC</td>
<td>PaaS</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>STRATOS</td>
<td>IaaS</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>MODAClouds</td>
<td>PaaS</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>CompatibleOne</td>
<td>IaaS</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Cloud4SOA</td>
<td>PaaS</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>soCloud</td>
<td>PaaS</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
</tbody>
</table>
1. Context and motivation
2. Challenges
3. State of the art
4. Contributions
   4.1. soCloud Model
   4.2. soCloud Platform
5. Validation
6. Conclusion & Perspectives
soCloud Overview
Outline

1. Context and motivation
2. Challenges
3. State of the art
4. Contributions
   4.1. soCloud Model
   4.2. soCloud Platform
5. Validation
6. Conclusion & Perspectives
“Provides a **model** to design a distributed applications in a simple and concise manner for a Multi-Cloud environment”
Identify requirements for engineering distributed application for the Multi-Cloud environments

- **Multi-Cloud Portability**
  - Abstraction
  - Standard
  - Structure

- **Multi-Cloud Provisioning**
  - Placement
  - Resources
  - Granularity

- **Multi-Cloud Elasticity**
  - DSL
  - Simple

- **Multi-Cloud High-availability**
  - Failures
  - Diversity
soCloud Model

Extended SCA Model
• Service Component Architecture (SCA)
  ✤ Set of OASIS specifications
  ✤ Distributed applications
  ✤ Using SOA
soCloud Model based on SCA

Component
+name
+autowire

Property
+name
+mustSupply
+many
+value
+element

ConstrainingType
+name
+autowire
+targetNamespace
+local

Annotation
+name
+value

Implementation
+requires
+policySets

Implementation.Contribution
+name

Implementation.Composite
+name

Implementation.Java
+class

Implementation.C++
+class

Implementation.BPEL
+process

Service
+name

Operation
+name
+requires
+policySets

Binding
+uri
+name

Reference
+name
+requires
+policySets

Interface
+conversational
+callBackInterface
+interface

Interface.java
+callBackInterface
+interface

Interface.WSDL
+callBackInterface
+interface

Wire
+source
+target

Composite

PolicyIntent

PolicySet

PolicySet

Service

Binding

Reference

Interface

Wire

Extension 1

Extension 2
soCloud Model based on SCA

Why **extend** SCA model?

**Implementation.Contribution**
+name

[Diagram showing extension 1 and 2 with annotations: +name, +value]
soCloud Model: implementation

**Implementation.Contribution**

+ name

**Extension 1**

- Provides **high level conceptual** view to a component
- Allows the **deployment** of the component as execution unit
- **Structured** components of distributed applications
soCloud Model: annotations

- Allowing to associate **non-functional** requirements to a **component**
- The **SCA model** does not allow us to take into account these non-functional requirements
soCloud Model: annotations

Annotation
+ name
+ value

Placement annotation
- location
- closer

Execution annotation
- vm
- database

Availability annotation
- replication

Elasticity annotation
- elasticity
soCloud Model: annotations

**Annotation**
- name
- value

**Placement annotation**
- location
- closer

**Execution annotation**
- vm
- database
- service

**Availability annotation**
- replication

**Elasticity annotation**
- elasticity
soCloud Model: annotations

1 @ location = ‘value’
2 @ closer = ‘value’
soCloud Model: annotations

@location = ‘value’

Clouds provider

Amazon EC2 Windows Azure

Africa America Asia Europe Oceania

South Africa New York California Singapour Amazon Ireland Ireland France Australia

Paris Roubaix
soCloud Model: annotations

\[ 2 \text{@closer} = \text{‘value’} \]

C1 \[\text{@closer} = \text{‘C2’}\] Latency \(\text{C2}\)
soCloud Model: annotations

Annotation
+ name
+ value

Placement annotation
- location
- closer

Execution annotation
- vm
- database
- service

Availability annotation
- replication

Elasticity annotation
- elasticity
soCloud Model: annotations

1. @vm = ‘type_vm’
2. @database = ‘name -> version’
3. @service = ‘name -> version’

Example

1. @vm = ‘micro’
2. @database = ‘MySQL’
3. @service = ‘IronMQ -> 2.8.9’
soCloud Model: annotations

Annotation
+ name
+ value

Placement annotation
- location
- closer

Execution annotation
- vm
- database

Availability annotation
- replication

Elasticity annotation
- elasticity
soCloud Model: annotations

1. `@replication = 'number'`

Availability annotation

```
replication
```

Example

```
C
```

become

```
LB
```

```
C
```

@replication=5
soCloud Model: annotations

Annotation
+ name
+ value

Placement annotation
- location
- closer

Execution annotation
- vm
- database

Availability annotation
- replication

Elasticity annotation
- elasticity
A **DSL** for describing elasticity
soCloud Model: elasticity language

Event Action Condition

scaling up when (average (cpuUsage, 120s) > 80%)

minimize availability when (totalCost(costCompute, 24 h) > 900)

Elasticity is expressed on the Resources, Cost, Quality
soCloud Model: elasticity language

Trigger

scaling in 5 at (20:00 Friday)
soCloud Model: annotations

3-Tiers Application

@closer="Storage"
@vm="xlarge->Ubuntu"

@elasticity="Scaling in
When ( totalCost(computeCost, 24 h) > 200 )"

@database="MySQL"
@location="France"
Summary

• We show how we use annotation to describe non-functional properties and manage each component as unit of execution.
• New language is proposed to effectively express the elasticity.

1. Context and motivation
2. Challenges
3. State of the art
4. Contributions
   4.1. soCloud Model
   4.2. soCloud Platform
5. Validation
6. Conclusion & Perspectives
- The **expectations** in term of **execution support** for distributed applications built with soCloud Model are **differents**

- We need to provide a Platform that manages:
  - Multi-Cloud environments
  - Distributed applications in Multi-Cloud environments
We need to build Multi-Cloud Platform that:

- **react** to load  —> Scalable
- **react** to event —> Event-Driven
- **react** to failure —> Fault-Tolerance
- **react** to change —> Responsive
- self management —> Autonomic
- is flexible —> Component-based

**Reactive, flexible and self management platform**
soCloud Platform

- soCloud Platform is a distributed component-based PaaS for managing
  - Portability
  - Provisioning
  - Elasticity
  - High-availability
soCloud platform high level view

User

soCloud master

soCloud agent

soCloud applications
soCloud Platform: Multi-Cloud centric Architecture

Trend in the soCloud Platform Architecture

**single Cloud** centric Architecture

**Multi-Cloud** centric Architecture
soCloud Platform detail view

soCloud master

Load Balancer

Service Deployer
Constraint Validator

Controller
Workload Manager

Node Provisioning
PaaS Deployment
SaaS Deployment

Monitoring

soCloud agent
soCloud Platform: Fault Tolerance

Let it Crash

Application level

Replication in different clouds

Platform level

Replication in different clouds
To achieve this

1. **Transparency** is the ultimate goal [Waldo et. al]

2. **Automatic** component and applications **replication** [Waldo et. al]

3. All replications are **equal** and **deterministic** [Waldo et. al]

[Waldo et. al]-*Classic paper: A Note On Distributed Computing*
soCloud Platform: Replication features

• A cluster of N servers distributed across several Clouds

• Any (exactly one) component can be leader

• Active replication by the leader

• Consensus election of the leader

• Automatic failover

• Automatic recovery
soCloud Platform: deployment stack

- master
- agent

soCloud

IaaS

SCA container
Servlet container
Java runtime
Linux/OS
Resources
soCloud Platform: deployment stack

- soCloud
- agent
- master
- PaaS
- SCA container
- Servlet container
Summary

- Runtime support for managing Multi-Cloud **portability**, **provisioning**, **elasticity** and **high-availability**
- **Reactive** Platform


Outline

1. Context and motivation
2. Challenges
3. State of the art
4. Contributions
   4.1. soCloud Model
   4.2. soCloud Platform
5. Validation
6. Conclusion & Perspectives
Validation

soCloud Model

soCloud Platform
Validation: soCloud Model

Modeling of three concrete applications using the soCloud Model

1. APISENSE application
2. DiCEPE application
3. P2P Monitoring application
Validation: soCloud Model

1. APISENSE application [Nicolas Haderer]

APISENSE Central Node

- Create task
- Store data

APISENSE Sensing Store

- Publish task
- Subscribe task

Sensing Node

- Download sensing task
- Reward
- Send data

Send data

Download sensing task

Share this:

Like this:

65
Validation: soCloud Model

- Geo-location
  - Paris
- Unpredictable growth of smartphones
- Availability despite failures
- Cost control
Validation: soCloud Model

```xml
<composite name="Application-APISENSE">
  <component name="SensingNode">
    <implementation.contribution contribution="sensingnode.zip"/>
    <reference name="compute" target="CentralNode/compute"/>
    <reference name="storage" target="SensingStorage/storage"/>
    <annotation name="location">Paris</annotation>
    <annotation name="replication">2</annotation>
    <annotation name="elasticity">
      scaling in when (totalCost(computeCost, 24h) > 1000)
    </annotation>
  </component>
</composite>
```
Summary

• The soCloud Model has enabled us to build an App for collecting data from smartphones, an App to integrate heterogenous CEP Engines and make Big Data, and finally a P2P distributed App.


Validation: soCloud Platform

1. Portability
2. High-availability
3. Elasticity
4. Overhead introduced by soCloud
Portability
Validation: soCloud Platform

Deployed on 10 Clouds

IaaS and PaaS
High-availability
Validation: soCloud Platform

Availabilty = \frac{MTBF^*}{MTBF + MTTR^{**}} [Marcus et. al.]

MTBF^* = \text{Mean Time Between Failure}
MTTR^{**} = \text{Mean Time To Recover}

[Marcus et. al.] : Blueprints for High availability
# Validation: soCloud Platform

<table>
<thead>
<tr>
<th></th>
<th>MTTR* (Hour)</th>
<th>MTTR (Minute)</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>soCloud</td>
<td>0.06 Hour</td>
<td>3.6 Minutes</td>
<td>-</td>
</tr>
<tr>
<td>Public clouds [IWGCR]</td>
<td>7.5 Hours</td>
<td>450 Minutes</td>
<td>125</td>
</tr>
</tbody>
</table>

**MTTR** = Mean Time To Recover  
[http://iwgcr.org](http://iwgcr.org)
If it is assumed that a failure occurs once per year

**MTBF = 8760 Hours**

<table>
<thead>
<tr>
<th></th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>soCloud</td>
<td>( \frac{8760}{8760 + 0.06} = 99.999% )</td>
</tr>
<tr>
<td>Public clouds</td>
<td>( \frac{8760}{8760 + 7.5} = 99.914% )</td>
</tr>
</tbody>
</table>
Elasticity
Flash crowd effect

3-Tiers application was deployed on ten cloud providers

![Graph showing time vs. number of requests with two phases: Phase 1 and Phase 2.]

Total Number of Request = 3020000
1.3% of requests are failed that correspond to **34039**

Response Time = **65.90 s**
Flash crowd effect with soCloud elasticity

No request has failed
Without soCloud elasticity, the Response Time = 65.90 s
Response Time = 37.3 s  Response Time = 23.38 s
Phase 1  Phase 2
### Overhead introduced by soCloud

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Execution time</th>
<th>Overhead introduced by soCloud</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Application + FraSCAti)</td>
<td>10.85 sec</td>
<td></td>
</tr>
<tr>
<td>(Application + FraSCAti + soCloud)</td>
<td>11.10 sec</td>
<td>2.3%</td>
</tr>
</tbody>
</table>
### Overhead introduced by soCloud

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Execution time</th>
<th>Overhead introduced by soCloud</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Application + FraSCati)</td>
<td>10.85 sec</td>
<td>-</td>
</tr>
<tr>
<td>(Application + FraSCati + soCloud)</td>
<td>11.10 sec</td>
<td>2.3%</td>
</tr>
</tbody>
</table>

The **benefit** provided by the soCloud Platform **outweighs** the difference in the execution time.
Summary

• Reactivity face:
  ✤ Failures (*High-availability*)
  ✤ Flash crowd effect (*Elasticity*)
• Negligible *Overhead* introduced.
Outline

1. Context and motivation
2. Challenges
3. State of the art
4. Contributions
   4.1. soCloud Model
   4.2. soCloud Platform
5. Validation
6. Conclusion & Perspectives
Conclusion

soCloud Model

• We use annotations to express non-functional requirements.
• New language is proposed to effectively express the elasticity.
• The soCloud Model is illustrated on three distributed applications deployed in Multi-Cloud environments.

soCloud Platform

• Multi-Cloud PaaS for deploying, executing and managing distributed application.
• It was deployed on ten IaaS/PaaS clouds providers.
• soCloud Platform is capable of providing Multi-Cloud high-availability and elasticity to applications deployed on it.
Perspectives

Short-term further work

• The high-availability management despite software bugs.
• The elasticity management using reinforcement learning.

Further Research Directions

• Security for Multi-Cloud.
• Sharing state between replicates.
• Take into account changes of the underlying platforms.


• HADERER Nicolas, PARAISO Fawaz, RIBEIRO Christophe, MERLE Philippe, ROUVOY Romain and SEINTURIER Lionel: A Cloud-based Infrastructure for Crowd-sourcing Data from Mobile Devices. Springer Review (To appear)