Challenges in Establishment of Open Cloud Markets

Ivona Brandic
ivona@infosys.tuwien.ac.at
Vienna University of Technology, Austria

HPC Advisory Council Switzerland Conference 2012
March 13-15, Lugano
- Challenging scenarios in electronic marketplaces
  - specification of non-standardized goods
  - dynamic evolution of user requirements
  - high diversity of user requirements
Characteristics of Electronic Markets: Market Liquidity

Credits: Marcel Risch, Jörn Altmann, Ivona Brandic
Today’s Cloud landscape

- Large variety in services
- Fragmented dynamic participation

- Low market liquidity, stability and sustainability
1. Definition

Adding autonomic capabilities solves these problems!

2. Self-aware markets

Infrastructure
STANDARDIZED
GOODS
VIRTUAL
GOODS
Applications
Challenges

(1) Definition of virtual products

(2) Self-aware markets
From Autonomic Computing to Green ICT and self-aware Markets

Autonomic Manager

QoS Example

QoS Metric Protocol

analysis
planning

Knowledge

monitoring
execution

Service Compositions Mapping Strategies

Negotiation using VieSLAF framework

QoS Metric Protocol Evaluation

Sensor

Actuator

SLA template A

Markets

Markets

Markets

3

Adaptation

System Set up

Run time

A-MAPE-K Phases

3 Adaptation Phase

Knowledge Management Phase

Monitoring Phase

Traditional MAPE-K phases

FOSII Infrastructure
Definition of Virtual Products: The SLA mapping approach

Step 1: Assign service to a template

Service requirements

Changing properties or adding/deleting parameters

Step 2: Define SLA mappings

Public SLA template

Step 3: Lookup

Standardized goods

Step 4: Define mappings

M

Step 5: Adapt public templates

SLA Template Adaptation Component

Step 6: Modify SLA mappings

SLA Mapping Modification Component

Step 7: Assign new templates to users

Private template

Service provider

Service consumer

Credits: Ivan Breskovic
SLA Mapping Scenario

```
<xsl:template match="/ns:SLA/ns:ServiceDefinition/
  ns:WSDLSoapOperation/
  ns:Metric[@name='calculatePrice']/
  ns:Function">
  <ns:Function xsi:type="Times
    resultType="float">
    <ns:Operand>
      <xsl:copy>
        <xsl:copy-of
          select="@*|node()"/>
      </xsl:copy>
      </ns:Operand>
      <ns:Operand>
        <ns:FloatScalar>
          0.68096718
        </ns:FloatScalar>
      </ns:Operand>
    </ns:Function>
</xsl:template>
```
Step 1: Public SLA template management

Requirements:
1. function for finding distance between clustering items
2. methods for computing cluster centroids

Credits: Ivan Breskovic
Evolution of Public SLA Templates: Clustering Algorithms

- **Clustering**
  - Finding a structure in a collection of unlabeled data
  - Goal: determine the intrinsic grouping in a set of unlabeled data

- **Algorithms**
  - **DBSCAN**
    - based on density distribution of nodes
    - finds an appropriate number of clusters
    - two input values: distance $e$ and set of points that are not farther away than min distance
  - **k-Means**
    - assigns observations to clusters with closest means
    - New means to centroids
    - Stops when there are no significant changes

Source: http://home.dei.polimi.it/matteucc/Clustering/tutorial_html/
Computing distance between items

A. distance between template structures

\[ d_p(T_1, T_2) = \begin{cases} 
0 & \text{if properties of } p \text{ are same in } T_1 \text{ and } T_2 \\
1 & \text{if } T_1 \text{ or } T_2 \text{ does not contain } p \\
1 & \text{if only one property of } p \text{ differs in } T_1 \text{ and } T_2 \\
2 & \text{if both properties of } p \text{ differ in } T_1 \text{ and } T_2 
\end{cases} \]

B. distance between template objectives

\[ d_H(X, Y) = \max\{\sup_{x \in X} \inf_{y \in Y} d(x, y), \sup_{y \in Y} \inf_{x \in X} d(x, y)\} \]

\[ d_H(V_{T_1}, V_{T_2}) = \max(\|x_1 - x_2\|, \|y_1 - y_2\|) \]

\[ d_V(T_1, T_2) = \frac{d_H(V_{T_1}, V_{T_2})}{\max(d_{H,V})} \cdot \frac{\min(d_{H,V})}{\max(d_{H,V})} \]
Computing cluster centroids

- **Maximum method**
  - selects the maximum candidate

- **Threshold method**
  - selects the maximum candidate if it is used more than a given threshold

- **Significant-change method**
  - selects the maximum candidate if the percentage difference between the candidate and the current value exceeds a threshold

<table>
<thead>
<tr>
<th>Name of an SLA parameter</th>
<th>Cost</th>
<th>Charge</th>
<th>Rate</th>
<th>(Price)</th>
</tr>
</thead>
<tbody>
<tr>
<td>name:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># mappings:</td>
<td>15%</td>
<td>15%</td>
<td>40%</td>
<td>(30%)</td>
</tr>
</tbody>
</table>
Computing cluster centroids

- **Maximum method**
  - selects the maximum candidate

- **Threshold method**
  - selects the maximum candidate if it is used more than a given threshold

- **Significant-change method**
  - selects the maximum candidate if the percentage difference between the candidate and the current value exceeds a threshold

### Name of an SLA parameter

<table>
<thead>
<tr>
<th>name:</th>
<th>Cost</th>
<th>Charge</th>
<th>Rate</th>
<th>(Price)</th>
</tr>
</thead>
<tbody>
<tr>
<td># mappings:</td>
<td>15%</td>
<td>15%</td>
<td>40%</td>
<td>(30%)</td>
</tr>
</tbody>
</table>
Computing cluster centroids

- Maximum method
  - selects the maximum candidate

- Threshold method
  - selects the maximum candidate if it is used more than a given threshold

- Significant-change method
  - selects the maximum candidate if the percentage difference between the candidate and the current value exceeds a threshold

<table>
<thead>
<tr>
<th>Name of an SLA parameter</th>
<th>name:</th>
<th>Cost</th>
<th>Charge</th>
<th>Rate</th>
<th>(Price)</th>
</tr>
</thead>
<tbody>
<tr>
<td># mappings:</td>
<td>15%</td>
<td>15%</td>
<td>40%</td>
<td>(30%)</td>
<td></td>
</tr>
</tbody>
</table>
Step 2: Autonomic mapping modification

- **Input**
  - SLA templates: $T_{user}$, $T_{initial}$, $T_{new}$
  - SLA mappings: $T_{user} \leftrightarrow T_{initial}$, $T_{initial} \leftrightarrow T_{new}$

- **Result**
  - SLA mapping(s): $T_{user} \leftrightarrow T_{new}$
Step 2: Autonomic mapping modification

- **Input**
  - SLA templates: $T_{user}$, $T_{initial}$, $T_{new}$
  - SLA mappings: $T_{user} \leftrightarrow T_{initial}$, $T_{initial} \leftrightarrow T_{new}$

- **Result**
  - SLA mapping(s): $T_{user} \leftrightarrow T_{new}$
Step 2: Autonomic mapping modification

- **Input**
  - SLA templates: $T_{user}$, $T_{initial}$, $T_{new}$
  - SLA mappings: $T_{user} \leftrightarrow T_{initial}$, $T_{initial} \leftrightarrow T_{new}$

- **Result**
  - SLA mapping(s): $T_{user} \leftrightarrow T_{new}$
Step 2: Autonomic mapping modification

- **Input**
  - SLA templates: $T_{\text{user}}$, $T_{\text{initial}}$, $T_{\text{new}}$
  - SLA mappings: $T_{\text{user}} \leftrightarrow T_{\text{initial}}$, $T_{\text{initial}} \leftrightarrow T_{\text{new}}$

- **Result**
  - SLA mapping(s): $T_{\text{user}} \leftrightarrow T_{\text{new}}$
Clustering - Challenges

- **Case 1:** SLA parameter deleted from the template
  - *Is the new public SLA template good enough for the user?*

- **Case 2:** SLA parameter added to the template
  - *How to recognize new parameter’s equivalent in user’s private SLA template?*
Utility and cost model

- **Utility**
  0 - parameter does not exist in one of the templates
  1 - both parameter properties differ in templates
  2 - only one property differs in templates
  3 - templates do not differ with respect to the parameter

- **Cost**
  0 - user needs no new mappings
  1 - user must create new mapping(s)
  2 - user must create new adding/deleting wishes
  5 - user must rectify autonomically created mappings

**Overall net utility** = overall utility – overall cost
Simulation results

Overall net utility

Overall cost
Adding autonomic capabilities solves these problems!

1. Definition of virtual products
2. Self-aware markets
Autonomic market platform

- A marketplace is autonomic if it, once established, can exist and operate without human intervention.

Self-awareness:
- *knowledge of the self and the self’s state, such that autonomic capabilities can be introduced.*
Monitoring methodology

**Goals**

- Liquidity
  - Immediacy
  - Bid-ask-spread
  - Market depth
- Provider revenue
- Transaction volume
- Platform revenue
- No. of allocations
- Execution costs
- No. of active participants

**Metrics**

- Immediacy of matching
- Average price bids
- Average price requests
- No. of allocated resources
- Price of matching
- Participation costs on market platform
- Computation time
- Costs for platform resources
- Active traders in last observation period
Simulation environment

GridSim

- Grid Scenario
- User Requirements
- App Configuration

Grid Resource Brokers or Schedulers

- Data Set
- Grid Information Service
- Job Management
- Resource Allocation

... Resource

SimJava Simulation Kernel

OS / Hardware Infrastructure

Monitoring

- Mechanism Sensor
- Market Sensor
- Infrastructure Sensor
Case study: buyers stop bidding

Applied mechanism: continuous double auction
Infrastructure observations

Simulation properties
- 200 buyers, each with 100 requests
- 100 sellers

Used infrastructure
- MacBook Pro
- Dual Core 2.66 GHz
- 4 GB RAM
Goals affected

Goals

- Liquidity
- Immediacy
- Bid-ask-spread
- Market Depth

- Provider Revenue
- Transaction Volume
- Platform Revenue
- No. Of Allocations
- Execution Costs
- No. Of Active Participants
Applications: Scientific Computing

- Data analysis in bioinformatics

- Joint work with D. Kreil and P. Labaj, Chair of Bioinformatics, BOKU Vienna.
## Resulting Workflow

### SLA Template A
- Markets
- Markets
- Markets

### SLA Template B
- System Set up
- System Set up
- System Set up

### 3 Adaptation

### Planning
- Monitoring
- Execution
- Analysis

### Execution
- Monitoring
- Run-Time
- Sensor Net

### Monitoring
- Control loop
- Knowledge access
- Output sensor values

### Traditional MAPE-K Phases
- A-MAPE-K Phases

### Monitoring Phase
- Monitoring Agent

### Knowledge Management Phase
- Cloud Infrastructure

### Monitoring Framework
- Knowledge Management

### Million reads
- Cloud Infrastructure

### Mapping to genome
- Monitoring Agent

### Aligned reads
- Splitting to sub-reads

### Not-aligned reads
- Searching for splice-junctions

### Sub-reads Set 1
- Monitoring Agent

### Sub-reads Set 2
- Monitoring Agent

### Sub-reads Set n
- Monitoring Agent

### Reads aligned to the reference

### Calculation of other summary statistics
- Coding sequence, exon, gene span, splice forms, ...

### Estimation of gene/transcripts abstractions
- Abundance statistics
- Normalization within and between samples
- DE testing: GCSeq, baySeq, ...
- List of differentially expressed genes/transcripts

### GO/pathways enrichment testing: GOrank, ...

### Integration with other sources of knowledge
Green ICT

Cloud Visualization

Cloud Visualization

Utilization

Resource / PM

VM0 (CPU) VM0 (Memory) VM1 (CPU) VM1 (Memory) VM2 (CPU) VM2 (Memory) VM3 (CPU) VM3 (Memory) VM4 (CPU) VM4 (Memory) VM5 (CPU) VM5 (Memory) VM6 (CPU) VM6 (Memory) VM7 (CPU) VM7 (Memory) VM8 (CPU) VM8 (Memory) VM9 (CPU) VM9 (Memory) VM10 (CPU) VM10 (Memory) VM11 (CPU) VM11 (Memory) free (CPU) free (Memory) powering on (CPU) powering on (Memory) powering off (CPU) powering off (Memory) off (CPU) off (Memory)
Green ICT

Cloud Visualization

![Cloud Visualization Diagram]

Resources / PM:
- VM0 (CPU)
- VM0 (Memory)
- VM1 (CPU)
- VM1 (Memory)
- VM2 (CPU)
- VM2 (Memory)
- VM3 (CPU)
- VM3 (Memory)
- VM4 (CPU)
- VM4 (Memory)
- VM5 (CPU)
- VM5 (Memory)
- VM6 (CPU)
- VM6 (Memory)
- VM7 (CPU)
- VM7 (Memory)
- VM8 (CPU)
- VM8 (Memory)
- VM9 (CPU)
- VM9 (Memory)
- VM10 (CPU)
- VM10 (Memory)
- VM11 (CPU)
- VM11 (Memory)
- free (CPU)
- free (Memory)
- powering on (CPU)
- powering on (Memory)
- powering off (CPU)
- powering off (Memory)
- off (CPU)
- off (Memory)
Green ICT

Cloud Visualization

- t-1
- t=4 (Max t = 181)
- t+1

Cloud Visualization

Utilization

Resource / PM

VM0 (CPU)  VM0 (Memory)  VM1 (CPU)  VM1 (Memory)  VM2 (CPU)  VM2 (Memory)  VM3 (CPU)  VM3 (Memory)  VM4 (CPU)  VM4 (Memory)  VM5 (CPU)  VM5 (Memory)  VM6 (CPU)  VM6 (Memory)  VM7 (CPU)  VM7 (Memory)  VM8 (CPU)  VM8 (Memory)  VM9 (CPU)  VM9 (Memory)  VM10 (CPU)  VM10 (Memory)  VM11 (CPU)  VM11 (Memory)  free (CPU)  free (Memory)  powering on (CPU)  powering on (Memory)  powering off (CPU)  powering off (Memory)  off (CPU)  off (Memory)
Green ICT

- Joint work with Jean-Marc Pierson, Georges Da Costa, Damien Borgetto, University Paul Sabatier, Toulouse, France.
Conclusions and future work

- Self-aware market platforms are crucial for the future development of utility computing
- First step towards the vision: a monitoring methodology
  - a series of realistic market goals
  - sets of extractable metrics from a market platform
  - mappings for each goal as a proxy for platform performance

Future work
- consider additional case studies
- explore the remaining adaptation phases
- connect and integrate with existing platforms
Thank you for your attention!

Ivan Breskovic, Ivona Brandic, Vincent Emeakaroha, Michael Maurer
Vienna University of Technology, Austria

Christian Haas and Simon Caton
Karlsruhe Institute of Technology, Germany

Jörn Altmann
Seoul National University, Korea