A Generic Architecture for Developing Cloud Applications

Mohammad Hamdaqa, Tassos Livogiannis
Group 1
Department of Electrical and Computer Engineering
University of Waterloo
ECE750 Topic 5 – Spring 2010
Project supervisor: Ladan Tahvildari
{mhamdaqa, livtasos}@uwaterloo.ca
July 7, 2010
Outline

• Introduction
  – What is cloud computing?
  – Key benefits
  – Classic IT problem / managed cloud

• Project description
  – Motivation
  – Problem
  – Contribution / Expected results

• Methodology
What is Cloud Computing?

• Virtually infinite computing resources
• Provisioned on-demand
• Pay per use
• Available over the internet – “in the cloud”
Key Benefits

• Business agility

• Cost reduction
  – Reuse, Reuse, Reuse (AS A SERVICE)
  – Reduce Time To Market
  – Provisioning time from 35000 to 30 minutes [Lilly, David Powers]

• Reliability & Flexibility
  – Disaster recovery
  – Redundancy
Classic IT problem

Solution: managed cloud

From http://www.rightscale.com
Motivation

• Vendor Lock-in
  – What is the reason behind cost reduction?
    • Elasticity and portability
  – Many providers (MS, Google, SalesForce)
  – Different cloud platforms
    • Virtual Machine
    • API
• PSM that is not suitable for generic design
• Similar Concepts (virtual machines, persistence)
Problem

• Lack of modeling language, to model cloud applications
• Lack of cloud computing standards
• Lack of vocabulary and design patterns
Project contribution

• Providing a generic cloud architecture, that provides stakeholders with the basic components for modeling and developing cloud applications, Independent of the platform

• Generating a profile for cloud architecture using the eclipse modeling framework (EMF)

• Using the architecture to design a cloud application

• Using the generated profile to generate a skeleton code for a sample cloud application
Where are we?

- **Software-as-a-service**
  - Microsoft Online Services
  - Salesforce
  - Google Apps

- **Platform-as-a-Service**
  - Windows Azure
  - force.com
  - Google App Engine

- **Infrastructure-as-a-Service**
  - Amazon Web Services
  - GoGrid
Methodology

1. Study Current Cloud Platforms
2. Extract Main Cloud Components
3. Define the Relation between Cloud Components
4. Draw the Generic Cloud Architecture
5. Building a UML Profile for The Cloud Architecture
6. Design a Case Study
7. Generate Skeleton Code
Methodology (1/6)

- Study Current Cloud Platforms
- Extract Main Cloud Components
- Define the Relation between Cloud Components
- Draw the Generic Cloud Architecture
- Building a UML Profile for The Cloud Architecture
- Design a Case Study
- Generate Skeleton Code
Methodology (2/6)

- Not regular components
- Not regular services
- Special types of services
- Has its own characteristics
  - Size of VM
  - Number of instances
  - Back end or web service
  - Endpoints
  - Certificates
  - Miscellaneous properties
Methodology (3/6)

• Types of relationships
• How different cloud components intercommunicate
  ○ Ex. What is the relationship between back end services and web services
Methodology (4/6)

Ex. How can a Generic Architecture capture the scenario below:

![Diagram](image)

- Study Current Cloud Platforms
- Extract Main Cloud Components
- Define the Relation between Cloud Components
- Draw the Generic Cloud Architecture
- Building a UML Profile for The Cloud Architecture
- Design a Case Study
- Generate Skeleton Code

07/07/2010
Methodology (5/6)

1. Study Current Cloud Platforms
2. Extract Main Cloud Components
3. Define the Relation between Cloud Components
4. Draw the Generic Cloud Architecture
5. Building a UML Profile for The Cloud Architecture
6. Design a Case Study
7. Generate Skeleton Code
Methodology (6/6)

- Study Current Cloud Platforms
- Extract Main Cloud Components
- Define the Relation between Cloud Components
- Draw the Generic Cloud Architecture
- Building a UML Profile for The Cloud Architecture
- Design a Case Study
- Generate Skeleton Code
References


