Cloud Computing for Global Software Engineering: Designing Infrastructures and Supporting Migration

M. Ali Babar

Lancaster University, UK & ITU of Copenhagen, Denmark

Tutorial @ ISSSE, Salerno, Italy

July 8, 2013.
M. Ali Babar
Reader @ Lancaster University
PhD in CSE, University of New South Wales

Work History:
  ITU, CPH: Dec. 2009 …
  NICTA, Australia: 2003 - 2007
  JRCASE, Macquarie University: 2001 – 2003
  Various industrial roles in IT: Prior to 2001

Research in software architecture,
Service Orientation, Cloud Computing, and
Software Development Paradigm

http://malibabar.wordpress.com
Tutorial Goals

• Reflect about the areas of software engineering research motivated by cloud computing.

• Learn about some of the key challenges of developing software with distributed teams.

• Understand how cloud computing can be leveraged for global software engineering.

• Gain knowledge of the process and practices for migrating systems to cloud infrastructures.
Overview

• Cloud computing & research issues.

• Global software engineering & challenges.

• How to leverage cloud computing for GSE.

• Tool as a Service (TaaS) for GSE.

• Infrastructure requirements & potential solutions.

• Process and practices for migrating to cloud computing.
What is Cloud Computing?

“Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.” (A definition by the US National Institute of Standards and Technology (NIST))

• Some of the characteristics
  – Distributed computing at a massive scale.
  – On demand elasticity.
  – Exploiting existing technologies (Grid, Utility, Virtualization).
  – Pay per use model.
  – Driven by economies of scale.
A Snapshot of Potential Areas of Interest

Reproduced from Figure 1 of The Future of Cloud Computing: Opportunities for European Cloud Computing beyond 2010.
Service and Deployment Models

- **Software as a Service (SaaS)**
  - Google Apps
  - Zoho
  - Salesforce CRM
  - Microsoft Cloud Services

- **Platform as a Service (PaaS)**
  - Google App Engine
  - Microsoft Azure
  - Force.com
  - Yahoo Open Strategy

- **Infrastructure as a Service (IaaS)**
  - Amazon EC2
  - Eucalyptus
  - IBM – Computing On Demand (DoC)
  - VMWare vSphere

Deployment Models:
- Public Clouds
- Private Clouds
- Community Clouds
- Virtual Private Clouds
- Hybrid Clouds
A generic architecture that highlights the common elements of a cloud-based infrastructure – each element and its interactions can present several interesting research challenges.
Stakeholders Involved & their Perspectives

Use APIs offered by Cloud providers to exploit their resources and deliver high-value services for end-users.

The Cloud extends the general internet-based service provisioning model with aspects of high availability, reduced cost (through improved resource usage) and ease-of-use.

Dynamic management of hosted services (*aaS) to guarantee availability, reliability and related quality aspects through automation in order to optimise the overall resource utilisation.

Source: EASI-Cloud Review Presentation

Source: EC

Source: EASI-Cloud Review Presentation
Actors and Reference Architectures

**Cloud Consumer**
Person or organization that maintains a business relationship with, and uses service from Cloud Providers.

**Cloud Provider**
Person, organization or entity responsible for making a service available to Cloud Consumers.

**Cloud Auditor**
A party that can conduct independent assessment of cloud services, information system operations, performance and security of the cloud implementation.

**Cloud Carrier**
The intermediary that provides connectivity and transport of cloud services from Cloud Providers to Cloud Consumers.

**Cloud Broker**
An entity that manages the use, performance and delivery of cloud services, and negotiates relationships between Cloud Providers and Cloud Consumers.

Source: European Cloud Partnership Info Day 12/04/2012, inspired by the NIST roadmap
Some of the Barriers to Adoption in Europe

• **Data protection**
  – In particular US PATRIOTIC Act, problem for critical data, e.g., pharmaceutical research.

• **Legal**
  – Healthcare data cannot be processed out of country.

• **Market structure**
  – European market is mostly composed of small and diversified players with a general focus on B2B.

• **Migration to cloud computing**
  – Process and tool support.
A public agency wants to use cloud computing for storing and processing highly sensitive data.

An engineering company intends to use cloud computing but its documents are highly confidential.

Leveraging cloud bursting without violating the legal constraints and agreements with customers.
Cloud Computing Research Threads

- Decision Support Systems
- Processes for Engineering Clouds
- Architecting Cloud Systems & Services
Global Software Engineering
Many Meanings & Forms of GSE

- Open Source Software Development
- Inner Source Software Development
- Offshore Outsourcing
- Near-Shore Outsourcing
- Offshore Development Centers
- Global Software Development
- Distributed Software Development
Types of GSE Arrangements

Source: Gallivan and Oh, 1999

<table>
<thead>
<tr>
<th>One Client</th>
<th>Many Vendors</th>
</tr>
</thead>
<tbody>
<tr>
<td>One Vendor</td>
<td>Many Vendors</td>
</tr>
<tr>
<td>Simple</td>
<td>Multi-Vendors</td>
</tr>
<tr>
<td>Co-Sourcing</td>
<td>Complex</td>
</tr>
</tbody>
</table>

Source: Gallivan and Oh, 1999
Scenarios of Tool Support Challenges
Tools for Distributed Architecting Process

Main activities of software architecture process


Types of tools required

Architectural knowledge repositories → Decision support systems → Architecture modeling tools → Web 3.0 technologies
Tools for Knowledge Ecosystems

Private Ecosystem A
- Company
- Employee
- Implementing
  - create customized AK input form

Public Ecosystem
- collaboration
  - Private Ecosystem B
  - Private Ecosystem C

IDE
- AK Consume
- AK Extraction

KBase
- integration

Modeling Tool
- AK Consume
- AK Extraction

Requirement
- Integration

CM/Issue Tracking
- integration
Tool Support for GSE

Integration provided at the application level
Leveraging Cloud Computing for GSE
Why Cloud Computing Matters for GSE?

- Software testing
  - Stress testing with cloud-based infrastructure.
  - Enabling testing of application for sensitive data by offshore testers.

- Collaborative environments in Clouds
  - Just-in-Time (JIT) tool composition.
  - Processes, tools, & context aligned.

Source: http://aws.amazon.com/solutions/case-studies/8kmiles/
NexGen Infrastructure for GSE

• **Context**
  – Providing supportive technologies to GSE teams.

• **Challenges**
  – Dozens of different tools required.
  – Some commercial tools (IBM SameTime and MS Communicator) available but across vendor integration is problematic and the tools are expensive.
  – No Just In Time (JIT) composition and use of services.
  – Misalignment between tools, processes, and culture.

• **Proposed solution**
  – Cloud-based infrastructure for supporting GSE through Tool as a Service (TaaS).
Advantages of TaaS for GSE

- On demand tools acquisition & access to a wide range of tools.
- Processes and tools alignment – acquiring tools for process requirements of each project.
Advantages of TaaS for GSE

- Artifacts’ traceability across multiple sites.
- Implicit support for real-time awareness and collaboration.
Advantages of TaaS for GSE

• Access to sensitive and massive amount of data without data movement.

• Easy access to expensive and sensitive technologies.
A Coherent Picture of TaaS Benefits
A Chinese organization X is working on two GSD projects for Danish clients. Project A will be developed using open source technologies and requires access to private data and other services during unit and integration testing in different releases.

This project requires access to security and privacy sensitive data and services. Danish data security and privacy laws restrict the transfer of such data outside of Denmark.

If the data and services cannot be made available to organization X, complete development cannot be carried out on their premises in China and requires development of artifacts dealing with sensitive data and services inside Denmark.
Project B needs access to the real-time data about the traffic flows on Danish roads and expensive and proprietary platforms for simulation of the traffic flow patterns on road networks.

The organization may also need to shuffle developers within projects depending on the workload and criticality of the release. Configuration of development environments requires significant effort during resource shuffling.
How Would a Potential Solution Look Like?

Persistence Handling Services
Multi-tenancy Management
Virtual Nodes Manager
Response Transformation for Mobile Devices

SLA Compliance & Brokerage Services
Versioning Management

Data Management Services
Services Management Engine
Services Composition Modules

APIs For Integration With External Tools
Monitoring Services
SLA Modeling & Specification Services

Middleware Infrastructure

Expensive/Sensitive Hardware

Host Type 3 MS Project IBM Rational Suite

Project Managers' Tablets

Developers' Terminals

Private Cloud Hosting Sensitive Data
<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AgileZen</td>
<td>Collaborative project management.</td>
</tr>
<tr>
<td>LucidChart</td>
<td>Tool to create flowcharts, organizational charts, UML designs, mind maps etc.</td>
</tr>
<tr>
<td>MeetingSphere</td>
<td>Group meeting and decision support system.</td>
</tr>
<tr>
<td>Microsoft Live Meeting</td>
<td>Web conference service.</td>
</tr>
<tr>
<td>Microsoft Project</td>
<td>Project management solution.</td>
</tr>
<tr>
<td>Microsoft Team Foundation Server</td>
<td>Source control, data collection/reporting and project tracking</td>
</tr>
<tr>
<td>Pidoco</td>
<td>Software to design GUIs for web and mobile apps and make it live to share with other users.</td>
</tr>
<tr>
<td>IBM Rational Suite</td>
<td>Suite of tools for different phases of software development life cycle.</td>
</tr>
<tr>
<td>Cloud9 IDE</td>
<td>Cloud-enabled Online IDE.</td>
</tr>
<tr>
<td>Eclipse Orion</td>
<td>Cloud-enabled IDE that can be hosted on private/public clouds.</td>
</tr>
<tr>
<td>eXo Platform</td>
<td>Collaboration Platform and IDE.</td>
</tr>
</tbody>
</table>
Requirements & Potential Solutions!!!
• **Support for multiple organizations & teams**
  – Multi-tenancy for providing isolation between multiple services.
  – Privacy and security handling services.

• **Tools versioning and bundling**
  – Version management for maintaining partitioning between different versions of tools & combining them as a tool suite.

• **Integration with commercially available tools**
  – Platform neutral APIs and compatible data structure.

• **Tools working with private data and artefacts**
  – Workflow like distributable data processing services.
Requirements & Solutions

• Support for multiple types of persistence methods
  – Design & exploit a multi-tenant database driver.

• Accessibility from multiple types of devices
  – Dynamic distribution of processing load on devices & clouds.
  – Provide hooks for implementing or interacting with services for defining & selecting optimal configuration strategy for tools.
  – Transform responses to formats recognizable by client devices.

• Compliance with Service Level Agreement (SLA) – QoS
  – Specify machine readable & dynamically changeable SLAs.
  – Continuous monitoring & dynamic execution of services according to SLA specification (e.g., scalability rules & elasticity algorithms).
TaaS and SaaS: Consider the Differences

• **TaaS** contains executable artifacts – needs new security mechanisms.

• **TaaS** needs integration with other tools and middleware components.

• **TaaS** allows development of artifacts, which can consume unpredictable resources.

• **TaaS** needs different kinds of SLA.
Knowledge-Driven Infrastructure Design
Determining Architectural Requirements

• Sometime called:
  - architecturally significant requirements
  - architecture use cases

• Essentially the quality and non-functional requirements for a system
Key Requirements

- Supporting heterogeneous applications.
- Authentication and authorization.
- Applications, tools and service repository management.
- Applications and tools integration.
- Applications, tools and service composition.
- Application and tools alignment with process.
- Support for collaboration and traceability.
- Supporting virtual teams.
High Level View of the Middleware Platform
## Scenarios for Key Quality Attributes

<table>
<thead>
<tr>
<th>Quality Attributes</th>
<th>Scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-tenancy</td>
<td>The platform shall provide isolation between tenant specific data and services.</td>
</tr>
<tr>
<td>Versioning</td>
<td>The platform shall host multiple versions of the tools and composed services within tools and shall compose them according to subscription specifications of the tenants.</td>
</tr>
<tr>
<td>Tools interoperability</td>
<td>The platform shall provide interfaces so that tools can be composed and made interoperable with each other.</td>
</tr>
<tr>
<td>Decentralized artifacts</td>
<td>The platform shall allow services to work on decentralized artifacts by deploying services that will process sensitive data on cloud nodes.</td>
</tr>
<tr>
<td>SLA compliance</td>
<td>The platform shall allow autonomous composition of tools and associated services according to SLA specifications. SLA specification may include performance parameters like efficiency and throughput, or data privacy parameters such as location constraints.</td>
</tr>
</tbody>
</table>
Attribute Driven Design Method

Business Goals

Quality Attribute Scenarios
Functional Requirements
Constraints

Architectural Drivers

Architectural Pattern

Architectural Representation

Concurrency view
Decomposition view
Module decomposition view

Structure

Instance

Architecture for Version Management

- Tenancy partitioning module
- Version management module
- Configuration module
- Contract management module
- Instance pool module
- Pricing catalog

Architecture for Service Mediation

- Processes
- Relevant components

Source: Brandic et. al., Service Mediation and Negotiation Bootstrapping as First Achievements Towards Self-adaptable Grid and Cloud Services, 2009, ACM.
Design Flow and Backlog

Layered Model of the Infrastructure

- Interface Layer
  - Platform Interfaces
    - Repository Manager
    - Service Composer & Orchestrator
    - Multi-tenancy and Location Preference Manager
    - Provisioning Manager
    - Lifecycle Manager
  - Accounting Services
    - Data Manager
    - Authentication Manager
    - Collaboration Handler
  - Process Manager
- Integration & Process Management Layer
- Integration Bus
Lifecycle Provisioning & Management
Process Management

- Development Process Handler
  - Define Development Process – (1)

- Re-projection Components
  - Collaborative Artifacts – (4)
  - Define Collaboration & Traceability Activity – (2)

- Collaboration & Traceability Handler
  - Synthesized artifacts – (5)
  - Application/Tool Activities, Roles & User associated with a Collaboration Point – (3)

- Application & Tools Association
- Activity Manager
- Role Manager
- User Manager

- Data APIs

- Process Pipeline
  - CP-N
  - CP-3
  - CP-2
  - CP-1

- Integration Bus

- Workspace
  - Authentication Manager
  - Authentication Information
Collaboration Flow
Evaluate the Designed Architecture

1. Determine Quality Attributes
2. Generate Key Scenarios
3. Define Quality Rating Scale
4. Determine Architecture Alternatives
5. Prototype
6. Evaluate Quality Attributes
7. Present Evaluation Results

Main concerns of quality attributes
Utility tree
Scale definition

Architecture design

Output

Analyse Tradeoff
Suggest Improvement

Result presentation
Ratings for each quality attribute and architecture
Deployable implementation
Migrating to Cloud Computing
Migrating to Cloud Computing

• Methods, processes, and tools are required

• Supporting migration by providing process framework and guidelines

• A case of migrating a software metrics system - Hackystat
  – Supporting large number of organizations for process and product metrics in GSE
  – Requires elastic computing and storage resources
  – SaaS on IaaS (Amazon) or SaaS on PaaS (Goolge)
A Generic Migration Process

A: Requirements identification
   - System requirements

B: Identification of potential cloud hosting environments
   - List of potential cloud environments

C: Analyzing applications compatibility with potential cloud environments
   - Selected cloud environments

D: Identification of potential architecture solutions
   - Tradeoff analysis of cloud environments

E: Evaluation of cloud platforms for cloud specific quality attributes
   - Tradeoff analysis of quality attributes w.r.t. clouds

F: Evaluation of potential architecture solutions, their tradeoff analysis with respect to cloud environments and quality attributes supported by platforms.

G: Finalized design decision and modified system architecture
   - Implementation and system refactoring
   - Legend
     - Activities
     - Artifacts
Architectural View of HackyStat

- Provides visualization of different metrics through GUIs
- Generates reports for external clients
- Provides weekly, monthly and yearly abstractions of metrics
- Provides daily abstraction of data
- Receives and stores data and provides daily abstractions

Diagram:
- TickerTape
- ProjectBrowser
- SensorBase
- DailyProjectData
- Telemetry
- Database

Data Sensor1

Data Sensor2
**Characterizing Quality Attributes**

<table>
<thead>
<tr>
<th>Id No.</th>
<th>Quality Attributes</th>
<th>Scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>RQ2</td>
<td>Interoperability</td>
<td>A system’s components shall be deployable on commercial clouds providers (public cloud) as well as on organizations’ private clouds.</td>
</tr>
<tr>
<td>RQ3</td>
<td>Storage scalability</td>
<td>A system’s shall be able to handle unexpected increase in storage requirements by utilizing storage resources and storage services.</td>
</tr>
<tr>
<td>RQ4</td>
<td>Backward compatibility</td>
<td>End users of a system shall have consolidated view of the system through unified service interfaces, irrespective of the actual deployment configuration.</td>
</tr>
<tr>
<td>RQ5</td>
<td>Reliability &amp; auto-scaling</td>
<td>An end user’s operation and service requests shall not be affected during a system’s scalability operation.</td>
</tr>
<tr>
<td>RQ6</td>
<td>Optimal resource acquisition</td>
<td>A system shall efficiently acquire resources when needed and release those resources when not required in order to have more cost effective and green solution.</td>
</tr>
</tbody>
</table>
# Refined Form of a Scenario

<table>
<thead>
<tr>
<th>Scenario</th>
<th>System should be able to scale according to specific quality requirement (e.g. efficiency, throughput and response time).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute</td>
<td>Scalability</td>
</tr>
<tr>
<td>Attribute Concern</td>
<td>Replication of the system’s components according to the scalability requirements.</td>
</tr>
<tr>
<td>Scenario Refinement</td>
<td>Business and technical stakeholders.</td>
</tr>
<tr>
<td>Stimulus</td>
<td>Stakeholders desire to utilize on demand resource acquisition features of cloud computing environments.</td>
</tr>
<tr>
<td>Environment</td>
<td>Design time and run time.</td>
</tr>
<tr>
<td>Artifact</td>
<td>Wrapper services to control access to the replicated instances of the same service.</td>
</tr>
<tr>
<td>Response</td>
<td>Number of live instances of a service can be increased or decreased according to the number of requests targeted for a particular service.</td>
</tr>
<tr>
<td>Response Measure</td>
<td>Capability of the system to manage variable instances its composed services.</td>
</tr>
<tr>
<td>Quality Attributes</td>
<td>Architectural Decisions</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td><strong>Amazon EC2 &amp; S3</strong></td>
<td><strong>Google App Engine</strong></td>
</tr>
</tbody>
</table>
| Scalability | Replication of system services to meet performance requiremints.  
Separation of database layer into a new service that utilizes platform specific persistency features. | No action required. Scalability is handled by platform.  
Refactoring of persistency components to make it compatible with Google datastore persistence. |
| Portability | A wrapper layer is added to ensure platform independence.  
A separate database layer to provide seamless transfer of database layer. | Portability to other platforms is not possible. |
| Compatibility | System features are exposed through origonal REST API.  
A wrapper layer is added to provide abstraction to services cluster and their deployment configuration. | System features are exposed through origonal REST API. |
| Reliability & Autonomous Scalability | Façade/Waper layer to provide abstraction.  
Amazon’s Elastic Load Balancer ensures autonomous scalability. | Ensured by platform. |
| Efficient & effective deployments | Amazon Elastic Load Balancer ensures auto scaling as well as efficient and cost effective deployment configuration. | Deployment of application components on cloud is managed by platform. |
Architectural Views of Hackystat in Cloud
Key Observations for Research

1. Heavy reliance on cloud infrastructure
   - SaaS on PaaS or IaaS are tightly coupled with the APIs provided by a service provider
   - Huge efforts required for porting applications

2. Vendor dependent technology enhancements
   - Make it difficult to deploy and evolve a complex system in public clouds

3. Lack of tooling support
   - Existing tools only cover software specific details and do not consider the underlying virtual environment
Key Observations for Research

• Evaluation of quality attributes
  – Supporting implementation level investigation of static quality attributes e.g., complexity
  – Modeling & testing unknown users and infrastructures for scalability, accountability…..
  – Modeling & reasoning fine-grained control over privacy and security

• Testing
  – Change in the infrastructure of PaaS/SaaS platforms necessitate continuous testing
Architecture-Based Migration

- **Architecture Transformation**
  - Architectural Rep’n Concepts
  - Function-Level Rep’n
  - Code Structure Rep’n
  - Source Text Rep’n
  - Design Patterns & styles
  - Program Plans
  - Code styles
  - Legacy Source
  - New System Source

- **Architecture-Based Development**
Building a Body of Knowledge

Legend
- Categories
- Sub categories
- Problems addressed
- Technologies being used

Cloud Architectures

Management of cloud resources and services according to SLAs and GoS requirements.

Middleware Infrastructure & Platforms

Architecture considerations for porting existing applications on cloud.

Architecture Consideration For Migration

Challenges and research direction.

Architecture Challenges & Emerging Research

Domain Specific Architectures

General Guidelines

Quality Attributes

- Quality Attributes
- Adaptability
- Reliability
- Resource Provisioning
- SLA
- SaaS Integration
- Performance

Adaptability, scalability, SLA compliance, services interoperability, services integration and services accountability.

Multi-tenancy

Isolation of components, services and data.

- WSO2 Carbon Platform
- JavaEE
- JDBC

Frameworks

Frameworks for cloud application development and deployment.

- Aneka
- InterGrid
- Xen
- VMware
- KVM
- OpenPEX
- Hadoop
- Hbase
- Postgre
- Tomcat

Dataflow Security and Workflow Processing

Workflow management on cloud according to SLAs and GoS requirements.

- JVM
- Perl
- MeDICi
- Restlet
- Windows Azure Platform
- Amazon EC2

Supporting Hybrid Devices

Distribution of data and processing elements for providing applications for mobile devices.

- XMPP
- SQS
- OpenID
- OpenAuth
- REST
- NIX
Concluding Remarks!!!

• Cloud Computing matters – need to quickly learn how to exploit the promised benefits and address the challenges

• Cloud computing stimulates new research directions for new and novel ways of developing software & services

• Tools as a Service (TaaS) has a huge potential for SE in general and GSE in particular

• Imminent need for process and tool support for migration and/or upgrade of applications and tools for leveraging cloud computing paradigm
Acknowledgements

• Several colleagues and PhD students have hugely contributed to the work on GSE.
• A number of academic and industrial colleagues have contributed to the research problems and solutions incorporated in this presentation.
• TaaS work is being performed with Aufeef Chauhan through his PhD research.
• Tools review work has been performed with Paolo Tell for his PhD research.
• Socio-technical challenges and solution strategies work has been performed with Mansooreh Zahed for her PhD research.
References


References


• I. Brandic, et al., Service mediation and negotiation bootstrapping as first achievements towards self-adaptable grid and cloud services, Proceedings of the 6th international conference industry session on Grids meets autonomic computing (GMAC '09), 2009.

Thank You!

Questions

M. Ali Babar

alibabar.m@gmail.com
malibabar.wordpress.com