Evaluating a Cloud-Based Reactive Architecture using *cloud-ATAM*

David Ebo Adjepon-Yamoah

Centre for Software Reliability
School of Computing Science
Newcastle University

Stakeholders’ Group Work (July 2016)
Outline

Motivation
Present the ATAM/cloud-ATAM
Present the Business Drivers
Present the Architecture
Identify the Architectural Approaches
Generate the Quality Attribute Utility Tree and Scenarios
Analyse the Architectural Approaches
Present the Results
WHY?
Motivation

The existing architecture evaluation methods have limitations when assessing architectures interfacing with unpredictable environments such as the Cloud.

It is however, important to focus on holistic approaches combining aspects of both dynamic and static analysis of architecture dependability attributes.

This exercise introduces an ATAM derived methodology - cloud-ATAM - for evaluating the trade-off between multiple dependability quality attributes (i.e. availability and performance) of a cloud-based Reactive Architecture.
Why Early Phases to System Development?

Present the ATAM/cloud-ATAM

- Identify potential **risks**, and verify the **quality** before building.
- All *design* involves **trade-offs**.
- A *software architecture* is the earliest life-cycle artefact that embodies significant **design decisions**: choices and trade-offs.
System Quality Attributes
Present the ATAM/cloud-ATAM

- Non-functional characteristics
- Desired combination of system attributes/drivers:
  1. *Maintainability*:
     corrections, improvements, adaptation
  2. *Modifiability*:
     change quickly and cost effectively
     (Extensibility, portability, restructuring)
  3. *Flexibility*:
     ease of modification for case NOT designed
  4. *Availability*:
     ratio of time a system is functional as expected
  5. *Performance*:
     total effectiveness of a system, ....

ⓘ: [Keller 1990], [Avizienis et al 2004] are very good starting places for quality attributes.
Architecture Evaluation Method
Present the ATAM/cloud-ATAM

We need a method in which the **right questions** are asked early to:

- Discover **sensitivity points** – alternatives for which a slight change makes a significant difference in some quality attribute
- Discover **risks** – alternatives that might create future problems in some quality attribute
- Discover **trade-offs** – decisions affecting more than one quality attribute
## Architecture Evaluation Methods Comparison

**Present the ATAM/cloud-ATAM**

<table>
<thead>
<tr>
<th>Evaluation Method</th>
<th>ATAM (Kazman et al., 1990)</th>
<th>SAAM (Kazman et al., 1994)</th>
<th>AIMA (Lassing et al., 1999)</th>
<th>SAAMCS (Lassing et al., 2004)</th>
<th>SBAR (Bengtsson and Bösch, 1998)</th>
<th>ALPSM (Bengtsson and Bösch, 1999)</th>
<th>ESAAM (G. Möller, 1990)</th>
<th>SAAAMC (Bergey et al., 1990)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Method's Goal</strong></td>
<td>Sensitivity and trade-offs analysis</td>
<td>Risk identification and architectural suitability</td>
<td>Maintainability, cost prediction, risk assessment, SA selection</td>
<td>Risk assessment, developing complex scenarios for flexibility quality attributes</td>
<td>SA re-engineering to achieve quality attributes</td>
<td>Prediction of software maintainability</td>
<td>Integrates SAAM in a reuse-based, development &amp; domain-centric process</td>
<td>Comparison of several software architectures from different domains</td>
</tr>
<tr>
<td><strong>Evaluation Approaches</strong></td>
<td>Integrates questioning and measuring techniques</td>
<td>Scenario-based functionality and change analysis</td>
<td>Depends on analysis objectives</td>
<td>Scenario-based for complex scenarios</td>
<td>Multiple approaches</td>
<td>Scenario-based</td>
<td>Preparation of analysis template to collect protoscenario</td>
<td>Collect comparison criteria</td>
</tr>
<tr>
<td><strong>Quality Attributes</strong></td>
<td>Multiple quality attributes</td>
<td>Modifiability, maintainability</td>
<td>Flexibility</td>
<td>Multiple quality attributes</td>
<td>Maintainability</td>
<td>Modifiability</td>
<td>Multiple quality attributes</td>
<td></td>
</tr>
<tr>
<td><strong>Applicable Project Stage</strong></td>
<td>After SA, detailed or iterative improvement process</td>
<td>Once functions assigned to modules</td>
<td>During design phase</td>
<td>In the final version of the software architecture</td>
<td>System extension or re-engineering stage</td>
<td>During architectural design to predict software maintenance</td>
<td>In the final version of the software architecture</td>
<td>During architectural design</td>
</tr>
<tr>
<td><strong>Method's Activities</strong></td>
<td>Six activities in four phases</td>
<td>Six activities with some activities in parallel</td>
<td>Five activities executed sequentially</td>
<td>Three activities, two in parallel executed</td>
<td>Three activities carried out iteratively</td>
<td>Six activities</td>
<td>Same as SAAM but use reusable knowledge base</td>
<td>Six activities, one executed repeatedly</td>
</tr>
<tr>
<td><strong>Stakeholders' Involvement</strong></td>
<td>All relevant stakeholders</td>
<td>All major stakeholders</td>
<td>Various for different activities</td>
<td>All major stakeholders</td>
<td>Software architect</td>
<td>Designer</td>
<td>All</td>
<td>Designer</td>
</tr>
<tr>
<td><strong>Method's Application</strong></td>
<td>Remote temperature sensor, Battlefield Control System and many more</td>
<td>GIS, WRC5, keyword in context (KWIC), and embedded audio system</td>
<td>Telecommunications systems, information systems, and embedded systems</td>
<td>Various business information systems</td>
<td>Fire-alarm systems, measurement systems, and dialysis systems</td>
<td>Hemodialysis machine</td>
<td>Measurement system</td>
<td>Envisioned software proposed by various contractors</td>
</tr>
<tr>
<td><strong>Tool Support</strong></td>
<td>Not available, Partial, Ex-SAAMTOOL</td>
<td>Not available</td>
<td>Not available</td>
<td>Not available</td>
<td>Not available</td>
<td>Not available</td>
<td>Not available</td>
<td>Not available</td>
</tr>
</tbody>
</table>

**Figure:** Comparison of Architecture Evaluation Methods
Evaluating a Cloud-Based Reactive Architecture using cloud-ATAM

David Ebo
Adjepon-Yamoah

Architecture Trade-off Analysis Method (ATAM)
Present the ATAM/cloud-ATAM

- Proposed by Kazman as a technique for understanding the trade-offs points or dependencies among the attributes, inherent to architecture evaluation.
  "The purpose of the ATAM is to assess the consequences of architectural decision alternatives in light of quality attribute requirements [Kazman 1998]"

- Provides a way to evaluate software architecture’s fitness with respect to multiple competing quality attributes.

- Since these attributes interact, the method helps to reason about architectural decisions that affect quality attribute interactions.

- Follows a spiral model of design, postulating candidate architectures followed by analysis and risk mitigation, leading to refined architectures.
Spiral ATAM Model

Present the ATAM/cloud-ATAM

**Figure: Spiral ATAM Model**
Overview of ATAM
Present the ATAM/cloud-ATAM

- The purpose of an ATAM is **NOT to provide precise analyses.** The purpose IS to **discover risks** created by architectural decisions.
- We want to find trends: correlation between architectural decisions and predictions of system properties.
- Discovered risks can then be made the focus of mitigation activities: e.g. further design, further analysis, prototyping.
- **Examples of ATAM Projects:** US DOD Battlefield Control System, NASA Earthquake Monitoring System, etc.

**Overview of ATAM**

**Present the ATAM/cloud-ATAM**

- The purpose of an ATAM is **NOT to provide precise analyses.** The purpose IS to **discover risks** created by architectural decisions.
- We want to find trends: correlation between architectural decisions and predictions of system properties.
- Discovered risks can then be made the focus of mitigation activities: e.g. further design, further analysis, prototyping.
- **Examples of ATAM Projects:** US DOD Battlefield Control System, NASA Earthquake Monitoring System, etc.

**Overview of ATAM**

**Present the ATAM/cloud-ATAM**

- The purpose of an ATAM is **NOT to provide precise analyses.** The purpose IS to **discover risks** created by architectural decisions.
- We want to find trends: correlation between architectural decisions and predictions of system properties.
- Discovered risks can then be made the focus of mitigation activities: e.g. further design, further analysis, prototyping.
- **Examples of ATAM Projects:** US DOD Battlefield Control System, NASA Earthquake Monitoring System, etc.
ATAM: Scenario-Based Approach
Present the ATAM/cloud-ATAM

Figure: Scenario-Based Approach
Nine (9) ATAM Steps
Present the ATAM/cloud-ATAM

1. Present the ATAM
2. Present the Business Drivers
3. Present the Architecture
4. Identify the Architectural Approaches
5. Generate the Quality Attribute Utility Tree*
6. Analyse the Architectural Approaches**
7. Brainstorm and Prioritise Scenarios*
8. Analyse the Architectural Approaches**
9. Present the Results

(Steps 5 & 7 [*], and Steps 6 & 8 [**]: Focus of adaptation)
ATAM Concept Interactions

Present the ATAM/cloud-ATAM

Figure: ATAM Concept Interactions
If ATAM, why *cloud-ATAM*?

Present the ATAM/cloud-ATAM

- Variation of ATAM for *small-to-medium size* (i.e. ISO/IEC 14143:1998 & COSMIC Full FP 2.2) cloud-based systems/architecture. Avoids repetitions in steps that are not required. (*Towards analysis automation*)

- Provision of guidelines for qualitative or quantitative reasoning of architectures using Attribute-Based Architectural Styles (ABAS) (i.e. Continuous Markov Chain Analysis).
cloud-ATAM Overview

Present the ATAM/cloud-ATAM

Figure: cloud-ATAM: Adapted ATAM with Defined 3-Step Analysis Approach
Evaluating a Cloud-Based Reactive Architecture using cloud-ATAM

David Ebo Adjepon-Yamoah

Motivation

Present the Business Drivers

Present the Architecture

Identify the Architectural Approaches

Generate the Quality Attribute Utility Tree and Scenarios

Analyze the Architectural Approaches

Present the Results

Summary

**ATAM Phases**

Present the ATAM/cloud-ATAM

**Figure: The cloud-ATAM Phases**
Questionnaire: Section 1
Dynamic View: Reactive Architecture

Present the Business Drivers

Figure: Dynamic View of Reactive Architecture
Requirements of Reactive Architecture

Present the Business Drivers

1. **The Reactive Architecture must store all artefacts created in all the components of the main composing systems.**

2. **It must monitor and trace all changes to these artefacts to inform system stakeholders.**

3. The **Reactive Architecture** must support multiple users concurrently.

4. The **Reactive Architecture** must provide capacity to scale quickly to accommodate changing demands of system stakeholders, and failures.

5. The **Reactive Middleware** must enable heterogeneous access and analysis operations on artefacts in the **Shared Artefacts Repository**.

6. The **Shared Artefacts Repository** must be backed up asynchronously to facilitate roll-back of repository artefacts.

7. Critical systems such as the **Reactive Middleware** must not constitute a single point of failure which will affect the uptime of the system and the architecture.

8. The **System Engineering Toolbox** must facilitate sequential and parallel execution of tools in a workflow manner.

9. The **Formal Decomposition System** must provided a high capacity and dedicated channel to coordinate real-time analysis on artefacts for local client computers and on remote cloud environment.

10. The **Cloud Accountability System** must gather dependability metrics from several virtual machines, and perform a synchronous analysis of these metrics.

11. Security mechanisms must not degrade defined performance threshold. Specifically, response time for create, delete, update, and display artefact/data operations should not exceed 5 seconds at peak cloud (i.e. architecture) period and less than 1 second during off-peak period.

12. The **Reactive Architecture** must do all this while meeting the **performance** and **availability** requirements to allow it to keep up with the sturdy stream of data and operations on artefacts from the system engineering processes.
Constraints of Reactive Architecture

Present the Business Drivers

The constraints affecting the Reactive Architecture are identified along two perspectives:

1. Reactive Architecture:
   - Cost of development.
   - Time to market, which is dictated by customer demands. (i.e. changing requirements)
   - Skill set of architect.
   - Technical standards.

2. Environment of Architecture (i.e. Cloud Environment):
   - Architecture is highly reliant on the dependability of the cloud infrastructure.
   - Co-location of potentially risky systems on a cloud server/platform.
   - Security of the architecture and data is largely out of the control of the architect.
   - Reliance on COTS products of the cloud service provider (CSP).
Quality Attributes Under Review

Present the Business Drivers

- Availability
- Performance
Reactive Architecture

Present the Architecture

Composed of:

- Reactive Middleware (RM)
- System Artefacts Repository (SAR)
- System Engineering Toolbox (SET)
- Formal Decomposition System (FDS)
- Cloud Accountability System (CAS)
SYSTEM VIEW: Reactive Architecture

Present the Architecture

Figure: Overview of Reactive Architecture (Redundancy)
SYSTEM VIEW: Reactive Architecture

Present the Architecture

Figure: Component and Connector View of Reactive Architecture
Scenarios of Reactive Architecture

Present the Architecture

Initial....

- **Scenario 1**: User Management
- **Scenario 2**: Adding Tools to the Toolbox
- **Scenario 3**: Saving Artefacts
- **Scenario 4**: Downloading Artefacts
- **Scenario 5**: Sharing Artefacts
- **Scenario 6**: Change Management of Artefacts
- **Scenario 7**: Traceability of Artefacts
- **Scenario 8**: Primary Back-Up Repository
Evaluating a Cloud-Based Reactive Architecture using cloud-ATAM

David Ebo Adjepon-Yamoah

Motivation

Present the ATAM/cloud-ATAM

Business Drivers

Present the Architecture

Identify the Architectural Approaches

Generate the Quality Attribute Utility Tree and Scenarios

Analyse the Architectural Approaches

Present the Results

Summary

Figure: Traceability of Artefacts (Sequence Diagram)
## Requirements of Reactive Architecture

**Present the Architecture**

<table>
<thead>
<tr>
<th>Quality Attribute Goals</th>
<th>ID</th>
<th>Attribute-Specific Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operability</td>
<td>O1</td>
<td>The Reactive Architecture must store all artefacts created in all the components of the main composing systems.</td>
</tr>
<tr>
<td></td>
<td>O2</td>
<td>It must monitor and trace all changes to these artefacts to inform system developers (i.e., clients). (also P1)</td>
</tr>
<tr>
<td></td>
<td>O3</td>
<td>The System Engineering Toolset must facilitate sequential and parallel execution of tools in a workflow manner.</td>
</tr>
<tr>
<td></td>
<td>O4</td>
<td>The Formal Decomposition System must provide a high capacity and dedicated channel to coordinate real-time analysis on artefacts for local client computers and on remote cloud environment. (also P3)</td>
</tr>
<tr>
<td></td>
<td>O5</td>
<td>The Cloud Accountability System must gather dependability metrics from several virtual machines, and perform a synchronous analysis of these metrics.</td>
</tr>
<tr>
<td>Performance</td>
<td>P1</td>
<td>It must monitor and trace all changes to these artefacts to inform system developers (i.e., clients). (also O2)</td>
</tr>
<tr>
<td></td>
<td>P2</td>
<td>The Reactive Middleware must enable heterogeneous access and analysis operations on artefacts in the Shared Artefacts Repository. (also A2)</td>
</tr>
<tr>
<td></td>
<td>P3</td>
<td>The Formal Decomposition System must provide a high capacity and dedicated channel to coordinate real-time analysis on artefacts for local client computers and on remote cloud environment. (also O4)</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>Security mechanisms must not degrade defined performance threshold. Specifically, response time for create, delete, update, and display artefact/data operations should not exceed 5 seconds at peak cloud (i.e., architecture) period and less than 1 second during off-peak period. (also S1)</td>
</tr>
<tr>
<td></td>
<td>P5</td>
<td>The Reactive Architecture must do all this while meeting the performance and availability requirements to allow it to keep up with the steady stream of data and operations on artefacts from the system engineering processes. (also A4)</td>
</tr>
<tr>
<td>Scalability</td>
<td>Sc1</td>
<td>The Reactive Architecture must support multiple users concurrently. (also A1)</td>
</tr>
<tr>
<td></td>
<td>Sc2</td>
<td>The Reactive Architecture must provide capacity to scale quickly to accommodate changing demands of system developers, and failures. (also A1)</td>
</tr>
<tr>
<td>Availability</td>
<td>A1</td>
<td>The Reactive Architecture must provide capacity to scale quickly to accommodate changing demands of system developers, and failures. (also Sc2)</td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>The Reactive Middleware must enable heterogeneous access and analysis operations on artefacts in the Shared Artefacts Repository. (also P2)</td>
</tr>
<tr>
<td></td>
<td>A3</td>
<td>Critical systems such as the Reactive Middleware must not constitute a single point of failure which will affect the uptime of the system and the architecture. (also R1)</td>
</tr>
<tr>
<td></td>
<td>A4</td>
<td>The Reactive Architecture must do all this while meeting the performance and availability requirements to allow it to keep up with the steady stream of data and operations on artefacts from the system engineering processes. (also P5)</td>
</tr>
<tr>
<td>Maintainability</td>
<td>M1</td>
<td>The Shared Artefacts Repository must be backed up asynchronously to facilitate roll-back of repository artefacts.</td>
</tr>
<tr>
<td>Reliability</td>
<td>R1</td>
<td>Critical systems such as the Reactive Middleware must not constitute a single point of failure which will affect the uptime of the system and the architecture. (also A3)</td>
</tr>
<tr>
<td>Security</td>
<td>S1</td>
<td>Security mechanisms must not degrade defined performance threshold. Specifically, response time for create, delete, update, and display artefact/data operations should not exceed 5 seconds at peak cloud (i.e., architecture) period and less than 1 second during off-peak period. (also P4)</td>
</tr>
</tbody>
</table>

**Figure: Mapping Requirements to Quality Attributes**
ACTIVITY

Evaluating a Cloud-Based Reactive Architecture using cloud-ATAM

David Ebo Adjepon-Yamoah

Motivation

Present the ATAM/cloud-ATAM

Present the Business Drivers

Present the Architecture

Identify the Architectural Approaches

Generate the Quality Attribute Utility Tree and Scenarios

Analyse the Architectural Approaches

Present the Results

Summary

Questionnaire: Section 2
Quality Attribute Characterisation
Identify the Architectural Approaches

Figure: Availability Characterisation
Quality Attribute Characterisation

Identify the Architectural Approaches

**Figure**: Performance Characterisation - Stimuli
Quality Attribute Characterisation
Identify the Architectural Approaches

Figure: Performance Characterisation - Parameters
Quality Attribute Characterisation
Identify the Architectural Approaches

Figure: Performance Characterisation - Response
# Attribute-Specific Questions

**Identify the Architectural Approaches**

## Table: 1 Classified Attribute-Specific Questions

<table>
<thead>
<tr>
<th>Attribute-Specific Questions ID</th>
<th>Attribute-Specific Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASQ1</td>
<td>What facilities exist in the software architecture (if any) for self-testing and monitoring of software components? (Availability)</td>
</tr>
<tr>
<td>ASQ2</td>
<td>What facilities exist in the software architecture (if any) for redundancy, liveness monitoring, and fail-over? (Availability)</td>
</tr>
<tr>
<td>ASQ3</td>
<td>How is data consistency maintained so that one component can take over from another and be sure that it is in a consistent state with the failed component? (Availability)</td>
</tr>
<tr>
<td>ASQ4</td>
<td>What is the process and/or task view of the system, including mapping of these processes/tasks to hardware and communication mechanisms between them? (Performance)</td>
</tr>
<tr>
<td>ASQ5</td>
<td>What functional dependencies exist among the software components? (Performance)</td>
</tr>
<tr>
<td>ASQ6</td>
<td>What data is kept in the database? How big is it, how much does it change, who reads/writes it? (Performance)</td>
</tr>
<tr>
<td>ASQ7</td>
<td>How are resources allocated to service requests? (Performance)</td>
</tr>
<tr>
<td>ASQ8</td>
<td>What are the anticipated frequency and volume of data transmitted among the system components? (Performance)</td>
</tr>
</tbody>
</table>
## Architectural Approaches

### Identify the Architectural Approaches

<table>
<thead>
<tr>
<th>Architectural Approach ID</th>
<th>Architectural Approaches</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD1</td>
<td>We use the <strong>component-and-connector architectural style</strong> to represent the various components and connections/interfaces of the Reactive Architecture. This is particularly relevant because it <em>expresses the runtime behaviour of the architecture</em> under review. Also, <em>interfaces</em> are defined as <strong>application programming interfaces</strong> (APIs).</td>
</tr>
<tr>
<td>AD2</td>
<td>We <strong>AVOID</strong> the <strong>distributed data repository</strong> approach in designing the Shared Artefacts Repository. This prevents situations such as issues with <strong>data/database consistency</strong> and possible <em>modifiable concerns</em>.</td>
</tr>
<tr>
<td>AD3</td>
<td>The <strong>client-server</strong> approach is a best fit for the data-centric <strong>Shared Artefacts Repository</strong> system.</td>
</tr>
<tr>
<td>AD4</td>
<td>The <strong>Reactive Middleware</strong> will be adequately represented using the <strong>client-server</strong> approach.</td>
</tr>
<tr>
<td>AD5</td>
<td>Since the <strong>Reactive Middleware</strong> and the <strong>Shared Artefacts Repository</strong> constitute a <strong>single point of failure</strong>, we present the following approaches:</td>
</tr>
<tr>
<td>AD6</td>
<td>- <strong>Back-up</strong> of artefacts in the Shared Artefacts Repository.</td>
</tr>
<tr>
<td>AD6</td>
<td>- <strong>Distributed services</strong> (or modular set of services) for the components of the Reactive Middleware.</td>
</tr>
<tr>
<td>AD7</td>
<td><strong>Schema-free NoSQL data management system</strong> (DMS) is necessary for the Shared Artefacts Repository to minimise or remove <strong>bottlenecks</strong>.</td>
</tr>
<tr>
<td>AD8</td>
<td>An <strong>independent communication components</strong> approach (i.e. <strong>high bandwidth</strong>) for communication among the Reactive Middleware, Shared Artefacts Repository, Cloud Accountability System, and the Formal Decomposition System. Such communication approach is particularly relevant for the distributed components of the Cloud Accountability System.</td>
</tr>
</tbody>
</table>

---

**Table: 2 Architectural Approaches for the Reactive Architecture**

Evaluating a Cloud-Based Reactive Architecture using *cloud-ATAM*

David Ebo
Adjepon-Yamoah
Utility Tree

Generate the Quality Attribute Utility Tree and Scenarios

**Utility**

**Availability**
- Hardware Failures
  - Power outage at Availability Zone 1 requires traffic redirect to Availability Zone 2 in < 5 seconds. [L, M]
  - Disk crash must have a backup that takes over in < 3 seconds. [H, L]
  - Network failure is detected and recovered in 10 seconds. [M, L]
- COTS/Third Party Software Failures
  - COTS/Third party software update with bug that causes failures is reverted to stable version in < 5 seconds. [M, M]

**Performance**
- Data Latency
  - Deliver change requests and reports in real-time. [H, M]
  - Reduce storage latency for users to 200 milliseconds. [H, L]
- Transaction Throughput
  - One system (e.g., Reactive Middleware) should not constitute a lag of > 1 second. [M, L]
  - Accommodate over 500 queries per second. [H, M]

**Figure:** Attribute Utility Tree of Reactive Architecture
Evaluating a Cloud-Based Reactive Architecture using cloud-ATAM

David Ebo
Adjepon-Yamoah

Motivation

Present the Business Drivers

Present the Architecture

Identify the Architectural Approaches

Generate the Quality Attribute Utility Tree and Scenarios

Analyse the Architectural Approaches

Present the Results

Summary

Quality Attribute Scenarios
Generate the Quality Attribute Utility Tree and Scenarios

Table: 3 Prioritised Quality Attribute Scenarios (Ordered)

<table>
<thead>
<tr>
<th>No.</th>
<th>Quality Attribute Scenarios</th>
<th>Scenario ID</th>
<th>Numbered Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Disk (i.e. data repository) crash must have a back-up that takes over in less than 3 seconds</td>
<td>A2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td><em>Deliver change requests and reports in real-time</em></td>
<td>P1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Reduce storage latency for users to 200 milliseconds</td>
<td>P2</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>Accommodate over 500 queries per second</td>
<td>P4</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>Network failure is detected and recovered in 10 seconds</td>
<td>A3</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>COTS/Third party software update with bug that causes failures is reverted to stable version in less than 5 seconds</td>
<td>A4</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>One system (e.g. Reactive Middleware) should not constitute a lag greater than 1 second</td>
<td>P3</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>Power outage at Availability Zone 1* requires traffic redirect to Availability Zone 2* in less than 5 seconds</td>
<td>A1</td>
<td>3</td>
</tr>
</tbody>
</table>
Evaluating a Cloud-Based Reactive Architecture using cloud-ATAM

David Ebo Adjepon-Yamoah

Motivation
Present the ATAM/cloud-ATAM
Present the Business Drivers
Present the Architecture
Identify the Architectural Approaches
Generate the Quality Attribute Utility Tree and Scenarios
Analyze the Architectural Approaches
Present the Results
Summary

Figure: Component and Connector View of Reactive Architecture
### Focal Table for P1 Analysis

#### Analyse the Architectural Approaches

| **Sensitivities:** | * S1: Concern over network latency.  
* S2: Using a data-centric and client-server approach for the central repository can facilitate data integrity and consistency, BUT it makes the architecture sensitive to its faults and bottlenecks.  
* S3: Similarly, the central role played by the Reactive Middleware makes the architecture sensitive to faults, resource (i.e. CPU, memory) malfunctions or unavailability. |
|---|---|
| **Trade-offs:** | * T1: Availability (+) vrs Performance (-) vrs Reliability (-): defining a central artefacts repository makes artefacts readily available, BUT may be faced with bottlenecks when there is a burst of queries on the repository.  
* T2: Availability (+) vrs Performance (+): using APIs for component interfaces facilitate readily access to resources, and boosts performance.  
* T3: Availability (+) vrs Performance (-): client-server approach for the Reactive Middleware allows for multi-client service, BUT there can be an overwhelming network management performance constraint.  
* T4: Availability (+) vrs Performance (-): backing up the artefacts in the primary Shared Artefacts Repository allows for fail-over assurance, BUT the asynchronous (i.e. not in real-time) back-up process can affect performance. |
| **Risks:** | * R1: Data integrity.  
* R2: The risk is that the Reactive Middleware and the Shared Artefacts Repository constitute a single point of failure. |
| **Non-Risks:** | * N1: The non-risk is the use of application programming interface (API) approach which should stay compatible.  
* N2: The independent communication connections should enable real-time data transfer. |
## Analysis of Performance Scenario - **P1**

**Analyse the Architectural Approaches**

<table>
<thead>
<tr>
<th>Scenario ID: <strong>P1</strong></th>
<th>Scenario: Deliver change requests and reports in real-time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute(s)</td>
<td>Performance (influenced by Availability)</td>
</tr>
<tr>
<td>Environment</td>
<td>Normal Operations</td>
</tr>
<tr>
<td>Stimulus</td>
<td>Responsiveness to change events</td>
</tr>
<tr>
<td>Response</td>
<td>real-time</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Architectural Decisions</th>
<th>Sensitivity</th>
<th>Trade-off</th>
<th>Risk</th>
<th>Non-Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD1 C&amp;C + API</td>
<td>S1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AD2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AD3 Client-Server SAR</td>
<td>S2</td>
<td>T1</td>
<td>R1, R2</td>
<td>N1</td>
</tr>
<tr>
<td>AD4 Client-Server RM</td>
<td>S3</td>
<td>T3</td>
<td>R2</td>
<td>N1</td>
</tr>
<tr>
<td>AD5 Back-up</td>
<td>S1, S2</td>
<td>T4</td>
<td></td>
<td>N1</td>
</tr>
<tr>
<td>AD6 DS RM</td>
<td>S1</td>
<td></td>
<td>R1</td>
<td>N1</td>
</tr>
<tr>
<td>AD7 Schema-free-SAR</td>
<td></td>
<td></td>
<td>R2</td>
<td></td>
</tr>
<tr>
<td>AD8 ICC</td>
<td>S1</td>
<td></td>
<td></td>
<td>N2</td>
</tr>
</tbody>
</table>

Overview of Results

Present the Results

- The *cloud-ATAM* delivers the main **products**: sensitivities, trade-offs, and architectural risks in Table 4.

- From Table 5, the *cloud-ATAM* completed a full cycle by linking the **architectural decisions** to the **quality attributes** (i.e. availability, performance), and back to the **business goals** of the Reactive Architecture.
Summary

▶ We have motivated the need for architecture evaluation methods suitable for the dynamic unpredictable cloud environments. In particular, we have presented an evaluation method - *cloud-ATAM* - derived from ATAM for evaluating the *availability* and *performance* quality attributes of a cloud-based Reactive Architecture.

▶ The results from Tables 4 and 5 indicate that the *cloud-ATAM* found some trade-offs (i.e. T1, T3, T4). This answers our *research question*, and validates our *hypothesis* that *the cloud-ATAM is able to identify trade-offs between the availability and performance quality attributes for the Reactive Architecture*.

▶ **To Do**: Attribute Based Architectural Styles (Continuous Markov Chain Analysis)
Evaluating a Cloud-Based Reactive Architecture using cloud-ATAM

David Ebo Adjepon-Yamoah

Motivation
Present the ATAM/cloud-ATAM
Present the Business Drivers
Present the Architecture
Identify the Architectural Approaches
Generate the Quality Attribute Utility Tree and Scenarios
Analyse the Architectural Approaches
Present the Results
Summary

Questionnaire: Section 3