Applying Architectural Patterns for the Cloud: Lessons Learned During Pattern Mining and Application

Ralph Retter (Daimler TSS GmbH)  
ralph.retter@daimler.com

Christoph Fehling (University of Stuttgart, Germany)
Problem
We need to do Cloud Computing!

- elasticity
- pay-per-use
- SaaS
- IaaS
- cost
- standardization
- PaaS
- multi-tenancy
- network
- provider
- hybrid
- public
- collaboration
- storage
- messaging
- process
- business
- virtualization
- scale
- community
- dynamic
- management
- business model
- data center
- availability
- platform
- architecture
- compute
- SLA
- resource sharing
- outsourcing
- private
- location
- tooling
- SLA
- data center
- network
Cloud Computing Architectural Patterns
A Structured Approach

• Structures the Problem using a Pattern Language
  • Focus: Application Architecture for the cloud

patterns mined at different enterprises by different people!
The Reason

Why did we mine the patterns?
We need to do Cloud!
Typical Resulting Questions

• "Which cloud infrastructure (provider) is the right one for our enterprise?“
• "Is this application suitable for the cloud?"
• "Why isn't it as easy to deploy an application in our data center as it is to deploy a sample application in my favorite public cloud?“
What Happens Next...

- Business Process
- Application
- Platform
- Infrastructure

bottom up approach
Typical Result of Bottom Up

Business Process

Applications

(Virtual) Infrastructure

Cloud Infrastructure

Virtual Infrastructure
What You Really Want

Business Process

Applications

(Virtual) Infrastructure

Cloud native applications

Applications

Cloud Infrastructure

Infra-structure
The Approach

What we learned by mining and applying the patterns, and what they are good for!
Real Requirements != We need to do Cloud!

- Example requirements:
  - Deal with dynamic load patterns without provisioning for peak-load
    - ➔ save money!
  - Make application deployments easier and faster ...
    - ➔ save time through standardization!
  - ...

---

10
Better Questions
(Requirement Driven)

1. Which of my business processes benefit from cloud properties of underlying applications?
   - dynamic load patterns
   - Pay-per-use
   - Self service required?

2. Which applications drive these business processes and can they deal with:
   - Resource sharing / pooling
   - Elasticity as a result of requirement for dynamic load patterns and pay per use?

3. What (Cloud) Infrastructure and platforms are needed to support these applications?
Top Down Approach

- Business Process
- Application
- Platform
- Infrastructure
How To Use the Patterns

illustrative example - the coffee shop
Requirements of Illustrative Example:

• Functional
  • Make coffee-related specialities and sell them to walk-in customers
• Non-functional requirements
  • Deal with varying amount of simultaneous customers according to the time of the day
  • Maximize order throughput!
  • Keep lines as short as possible
Traditional Small Coffee Shop Architecture

Coffee Shop

Coffee Machine

Blender

Guy who
• Takes order
• Makes drink
• Takes cash

You
Cloud-Based Bigger Coffee Shop Architecture

Coffee Shop

- Coffee Machine
- Blender

Guy who
- Takes order
- Makes drink
- Takes cash

Coffee Shop

- Coffee Machine
- Blender

Guy who
- Takes order
- Makes drink
- Takes cash

You
Example: Cloud Native Application
Coffee-Shop Architecture

Elastic Complex
Business Logic

Order (cup with correlation token)

Data Store

Message Queuing

Cashier counter

Pick-up counter

Yummy Sweets

Elastic Web Frontend / Apps

Enjoy!

Pick up With correlation token

Notification With correlation token

Correlation Token (name)

Cash +name

„invoice“

Cash

order

Adapted from: Gregor Hohpe: Your Coffee Shop Doesn’t Use Two-Phase Commit, IEEE Software 2005
Cloud-Native Bigger Coffee Shop Architecture
Example Pattern: 3-Tier Cloud Native Application

**Presentation Tier**
- Load Balancer
- Presentation Application Component
  - Stateless Component
  - User Interface Component

**Business Logic Tier**
- Business Logic Application Component
  - Stateless Component
  - Processing Component

**Data Tier**
- Data Access Component
- Storage Offerings
- Elastic Queue

**Required Services @ Cloud Provider**
- Elastic Infrastructure
- Elastic Platform
- Node-based Availability
- Environment-based Availability

**Message-oriented Middleware**
- Virtual Networking
- Message-oriented Middleware
- Exactly-once Delivery

**Communication Offerings**
- At-least-once Delivery
- Transaction-based Delivery
- Timeout-based Delivery

**Data Store**
- Strict Consistency
- Eventual Consistency
- Relational Database

**Key-value Storage**
- Blob Storage
- Block Storage
Requirements on Cloud Platform / Infrastructure

Requirements resulting from Application Design

Environment-based availability

Node-based availability

VS.
Some More Lessons Learned

• Moving to virtualized Machines is not Cloud! Even if you automate it!
• Consider environment-based availability and dynamic horizontal scaling
  • Use highly available messaging and storage platform offerings
    • Make sure you are aware of the CAP Theorem and it’s implications
      – Decision to trade towards higher availability and eventual consistency has impact on business processes!
• Modularization!
  • Make sure you can scale different parts of your system according to their workload
  • Separate short-running transactions with the user from long-running transactions in backend
    • Use asynchronous messaging and compensation-based transaction models in backend
• Resource Sharing!
Top-Down Approach + Pattern Map

- Composite (Hybrid) Cloud Applications
- Fundametal Architecture
- Multi-Tenancy
- Integration
- Application Components
- Workload Patterns
- Cloud Native Applications
- Application Development & Management
- Infrastructure & Platform Offerings (Network, Compute, Storage) & their Properties
Pattern Format

**Title** = Unique Name

**Intent** = Purpose and Goal

**Icon** to use in Diagrams

**Driving Question**

**Context**: When is this pattern applicable

**Solution**: Brief description how problem is solved

**Sketch**: Big Picture how Pattern works

**Result**: Detailed description of solution and its results

**Related Patterns**: Links to other patterns

**Known Uses**: Publicly Accessible Services, Solutions, Descriptions... that implement the pattern
Cloud Computing Patterns - Summary

• **Composite Cloud Applications**
  • Common use cases
  • Example Applications

• **Cloud Application Architectures**
  • Building cloud applications
  • Integrating different clouds

• **Cloud Offerings**
  • Processing, storage, and communication functionality
  • Behavior of cloud offerings
    → Provide runtime functionality

• **Cloud Computing Fundamentals**
  • Cloud Service Models
  • Cloud Types
  • Application Workloads
    → Characterize the environment

• **Cloud Application Management**
  • Elasticity, resiliency, updates etc.
  • Automation of management

Not all Cloud Computing Patterns are new!

Many existing patterns can be transferred or simply used in the area of cloud computing.

http://cloudcomputingpatterns.org
Questions?

http://cloudcomputingpatterns.org

ralph.retter@daimler.com