

**Fault Tolerant, Self-Healing and Vendor Neutral
Multi-Cloud Patterns and Framework
Focusing on Deployment and Management**

by
Andrey Rybka

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Professional Studies
in Computing

at

School of Computer Science and Information Systems

Pace University

October 2017

Version 2.6

Copyright © 2017 Andrey Rybka

ProQuest Number: 10791155

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



ProQuest 10791155

Published by ProQuest LLC (2018). Copyright of the Dissertation is held by the Author.

All rights reserved.

This work is protected against unauthorized copying under Title 17, United States Code
Microform Edition © ProQuest LLC.

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

We hereby certify that this dissertation, submitted by Andrey Rybka, satisfies the dissertation requirements for the degree of *Doctor of Professional Studies in Computing* and has been approved.

Lixin Tao
Dr. Lixin Tao
Chairperson of Dissertation Committee

10/28/2017
Date

Charles Tappert
Dr. Charles Tappert
Dissertation Committee Member

10/28/2017
Date

Ronald J. Frank
Dr. Ronald Frank
Dissertation Committee Member

10/28/2017
Date

School of Computer Science and Information Systems
Pace University

Abstract

Fault Tolerant, Self-Healing and Vendor Neutral Multi-Cloud Patterns and Framework Focused on Deployment and Management

by

Andrey Rybka

Submitted in partial fulfillment
of the requirements for the degree of
Doctor of Professional Studies in Computing
October 2017

Many organizations are looking to migrate to the cloud and looking for the best way to do it securely, reliably and without vendor lock in. Most organizations have to pick a cloud provider that uses proprietary APIs and Software. Most vendors currently do not implement any cloud API standards i.e. TOSCA or OASIS CAMP. Therefore, to date, the standard approaches to cloud computing have not been successful. In addition, cloud providers experience outages, frequently with serious business impact – so fault tolerance in cloud environment still needs more research and clear and prescriptive guidance. Variable performance is also an issue because most cloud providers overprovision their virtualized infrastructure and it results in degradation of performance and quality of service for customers depending on overprovisioning factor set by the cloud provider.

This study focuses on development of multi-cloud vendor neutral framework and patterns that deliver non-proprietary APIs and Software above IaaS layer with the functionality that will be on par with proprietary software or service offered by an individual cloud provider. We demonstrate how to design Fault Tolerant, Self-Healing, Performant, Secure and Cost Efficient Deployment using Patterns in the Multi-cloud environment without vendor lock in. We detail and catalog the developed solutions to common multi-cloud problems via patterns and multi-cloud framework using open source software which ensures portability across cloud providers. Framework and patterns can be reproduced by setup scripts, code and examples which are provided in the accompanying code repository. All of the failure scenarios are validated and demonstrated clearly showing the fault tolerance and limits of each solution.

Acknowledgements

First I would like to thank my advisor - Dr. Lixin Tao for all for all the help, patience, support and guidance provided over the years. In addition, I am deeply grateful to my family for their patience and support over all these years. I would also like to thank dissertation committee members Dr. Ronald Frank and Dr. Charles Tappert as well as DPS faculty who have helped me a great deal on this journey.

PREVIEW

Table of Contents

Abstract.....	iii
Acknowledgements	iv
List of Figures	ix
Chapter 1 - Introduction	1
1.1 Multi-Cloud Deployment for Fault-Tolerance, Performance, Security and Cost Efficiency.....	6
1.2 Current Solutions and Their Limitations	10
1.3 Problem Statement.....	16
1.4 Solution Methodology	19
1.5 Expected Contributions.....	21
1.6 Dissertation Roadmap/Outline	27
1.7 Conclusion.....	27
Chapter 2 – Survey of Relevant Research.....	28
2.1 Relevant Topics for Literature Review.....	28
2.2 Relevant Definitions and Examples.....	28
2.3 Existing Work around Cloud Design Patterns	35
2.4 Existing Work around Fault Tolerance within Distributed and Cloud Systems.....	39
2.5 Existing Work Around Multi-Cloud Deployments and Standards.....	41
2.6 Security Related Papers Focused on Cloud and Multi-Cloud Deployments	46
2.7 Self-Healing Cloud Research.....	49
2.8 Additional Relevant Distributed Systems and Multi-cloud Research	51
2.9 Cloud Cost Efficiency Related Research.....	52
2.10 Conclusion.....	53
Chapter 3 - Patterns and Open-Source Framework for Effective Multi-Cloud Deployment	55
3.1 Assumptions and Objectives for Supporting Open-Source Multi-Cloud Deployment.....	55
3.1.1 Problem Statement.....	55
3.1.2 Key Assumptions	56

3.1.3 Objectives and Scope of the study	60
3.1.4 Solution Methodology - how do we approach addressing these objectives? ...	61
3.1.5 Solution Limitations.....	62
3.2 Key Challenges while Deploying Enterprise Computing in Multi-Cloud Environment.....	64
3.1.1 Initial Multi-Cloud Deployment Challenges	64
3.2.1 Challenges related to Multi-Cloud Management after initial deployment and dealing with failures	65
3.2.3 Cost Efficiency Multi-Cloud Challenges	65
3.2.4 Security Multi-Cloud Challenges	65
3.2.5 General Application Deployment Multi-Cloud Challenges	66
3.3 Open-Source Multi-Cloud Deployment Solution Framework.....	67
3.4 Solutions to Major Multi-Cloud Computing Challenges.....	70
3.4.1 Initial Multi-Cloud Deployment Solutions.....	70
3.4.2 Multi-Cloud Management after Initial Deployment and Dealing with Failures	80
3.4.3 <i>Cost Efficiency Multi-Cloud Challenges</i>	90
3.4.4 <i>Security Challenges in Multi-Cloud Environment</i>	94
3.4.5 <i>General Multi-Cloud Application Deployment Challenges</i>	97
3.5 Summary	101
Chapter 4 - Detailed Multi-Cloud Design Patterns and Multi-Cloud Based on Open-Source Technologies	107
4.1 Multi-Cloud Foundation Patterns focused on Fault Tolerant Deployment Solutions	107
4.1.1 <i>General Multi Availability Zone Fault Tolerant Pattern</i>	107
4.1.2 <i>General Multi-Cloud Fault Tolerant Routing Pattern</i>	116
4.1.3 <i>Multi-Cloud Cloud Blueprint Pattern</i>	122
4.1.4 <i>Image Build Pipeline Pattern for Multi-Cloud Deployment</i>	125
4.1.6 <i>Multi-Cloud Service Registry and Discovery API</i>	132
4.2 Multi-Cloud Management after Initial Deployment and Dealing with Failures in Automated Way.....	138
4.2.1 <i>Multi-Cloud Telemetry and Log Aggregation Pattern</i>	138

4.2.2 SLA Enforcer Rules Engine	141
4.2.3 Multi-Cloud Data Replication	146
4.2.4 Reactive Multi-Cloud Health Check and Load Balancing Pattern	148
4.2.5 Proactive Multi-Cloud SLA Policy Enforcement Pattern	150
4.2.6 Public Cloud Bursting Pattern	155
4.2.7 Multi-Cloud Disaster Recovery Pattern	158
4.3 Cost Efficiency Patterns.....	162
4.3.1 Multi-cloud Aggregate Billing and Chargeback Pattern.....	162
4.3.2 Cost Efficiency Discount Multi-Cloud Pattern	166
4.4 Security Related Patterns	171
4.4.1 Multi-Cloud Secret Storage and Retrieval – Secrets Vault Pattern	171
4.4.2 Multi-Cloud Auditor Pattern	173
4.5 Multi-Cloud Control Plane Framework.....	176
4.6 Combining Control Plane Framework with Additional Application Use Case Patterns	179
4.6.1 Multi-Cloud Web Application / Service Pattern.....	179
4.6.2 Multi-Cloud Internet of Things Event Stream Ingesting Pattern and Big Data Pipelines.....	182
4.6.3 Container Orchestration Pattern for Multi-Cloud Deployments	185
4.7 Conclusion and Pattern Mappings to the Particular Problem	190
Chapter 5 - Experimental Validation.....	198
5.1 Implementations for Key Components of the Multi Cloud Framework.....	198
5.2 General Approach to Validation.....	199
5.3 General Multi-Cloud Cloud Fault Tolerant Routing Pattern Validation	201
5.4 Multi-Cloud Cloud Blueprint and Multi-Cloud Deployer Pattern Validation.....	206
5.6 Image Build Pipeline Pattern Validation	212
5.7 Multi-Cloud Service Registry and Discovery	217
5.8 Multi-Cloud Container Orchestration Validation.....	220
5.9 Multi-Cloud Data Replication Pattern Validation.....	222
5.10 Conclusion.....	226

Chapter 6 - Summary of Contributions and Future Work.....	227
6.1 Summary of Main Contributions.....	227
6.2 Future Work	233
Appendix A – Major Cloud Provider Outages	234
References.....	241
Academia References	241
Industry and Web References	247

PREVIEW

List of Figures

Figure 1 - Infrastructure as a Service vs. Platform as a Service	2
Figure 2 - Infrastructure Software.....	5
Figure 3 - 2015 Major Cloud Provider Outage in Hours.....	7
Figure 4 - Multi-Cloud Hybrid Deployment Pattern.....	8
Figure 5 - Microsoft Azure Big Data Offerings Circa 2016.....	11
Figure 6 - Google Proprietary Big Data Offerings.....	12
Figure 7 - Time Series AWS Specific Pattern	13
Figure 8 - Vendor Neutral Time Series Pattern	15
Figure 9 - Multi-Cloud Control Plane Framework	67
Figure 10 - Multi-Cloud Cloud Blueprint Pattern.....	71
Figure 11 - Multi-Cloud Image Builder Pattern	73
Figure 12 - Multi-Cloud Deployer Pattern	75
Figure 13 - Fault Tolerance in Multi-Cloud Environment	76
Figure 14 - Routing in Multi-Cloud Environment.....	78
Figure 15 - Multi-Cloud Registry and API Pattern.....	80
Figure 16 - Telemetry, Logs and Failure Detection in Multi-Cloud Environment.....	81
Figure 17 - Multi-Cloud SLA Enforcer Rules Engine	83
Figure 18 - Multi-Cloud Data Replication.....	84
Figure 19 - Reactive Multi-Cloud Health Check and Load Balancing Pattern	85
Figure 20 - Advanced Failover Based on SLA Telemetry	87
Figure 21 - Exhausted Capacity Failover	88
Figure 22 - Continuous Data Replication Needed for Multi-Cloud Disaster Recovery ...	89
Figure 23 - Multi-Cloud Disaster Recovery Pattern	90
Figure 24 - Aggregate Billing in Multi-Cloud Environment.....	92
Figure 25 - Taking Advantage of Cloud Provider Price Discounts Dynamically	93
Figure 26 - Multi-Cloud Secrets Storage and Retrieval	95
Figure 27 - Multi-Cloud Security Policy Auditor Pattern	96

Figure 28 - Multi-Cloud Web Application / Service Pattern.....	98
Figure 29 - Multi-Cloud Internet of Things and Big Data Deployment	99
Figure 30 - Multi-Cloud Deployment and Orchestration with Containers.....	101
Figure 31 - Sequence Diagram of Obtaining Availability Zone Information and Placement of Virtual Machines	109
Figure 32 - Multi Availability Zone Deployment with Health Checking	112
Figure 33 - Multi Availability Zone Deployment with Load Balancers and Health Checking	113
Figure 34 - Same Geographical Region Deployment for Low Latency.....	114
Figure 35 - Multi-Cloud Deployment Example.....	115
Figure 36 - DNS Only Multi-Cloud Fault Tolerant Routing Pattern	117
Figure 37 - Multi-Cloud Fault Tolerant Routing Pattern with DNS and API Failover ..	118
Figure 38 - Multi-Cloud Routing API Failover Deployment	119
Figure 39 - Multi-Cloud DNS Implementation Example.....	120
Figure 40 - DNS and API Fail-over Implementation Example	121
Figure 41 - Multi-Cloud Cloud Blueprint Pattern.....	123
Figure 42 - Multi-Cloud Bootstrap Pattern Open Source Implementation Options	124
Figure 43 - Multi-Cloud Cloud Image Build Pattern	127
Figure 44 - Multi-Cloud Cloud Image Build Pattern Implementation with Open Source Software	128
Figure 45 - Multi-Cloud Node Deployer - Orchestrator Pattern.....	130
Figure 46 - Multi-Cloud Node Deployer - Orchestrator Pattern with Multi Availability Support.....	131
Figure 47 - Multi-Cloud Service Registry and Discovery API with Multi-Cloud Control Plane	134
Figure 48 - Multi-cloud Registry and API High Availability Deployment.....	135
Figure 49 - Multi-cloud Registry and API Implementation Example.....	136
Figure 50 - Multi-Cloud Telemetry and Log Aggregation Pattern.....	139
Figure 51 - Multi-Cloud Telemetry and Log Aggregation Pattern with Remote Probes	140
Figure 52 - Multi-Cloud Telemetry and SLA Enforcer Rules Engine.....	143
Figure 53 - Multi-Cloud Data Replication Deployment.....	147

Figure 54 - Reactive Multi-Cloud Health Check and Load Balancing Pattern	150
Figure 55 - Proactive Multi-Cloud Health Check and Load Balancing Pattern	152
Figure 56 - Proactive Multi-Cloud Health Check and Load Balancing Pattern with Multi-Cloud Control Plane	152
Figure 57 - Public Cloud Bursting Pattern	156
Figure 58 - Multi-Cloud Disaster Recovery Pattern	159
Figure 59 - Multi-Cloud Disaster Recovery Pattern with Multi-Cloud Control Plane ...	160
Figure 60 - Multi-Cloud Disaster Recovery Pattern Detailed	161
Figure 61 - Multi-cloud Aggregate Billing and Chargeback Pattern.....	164
Figure 62 - Multi-cloud Aggregate Billing and Chargeback Pattern with SLA Enforcer Rules Engine	165
Figure 63 - Cost Efficiency Discount Multi-Cloud Pattern.....	168
Figure 64 - Multi-Cloud Secret Storage and Retrieval – Secrets Vault Pattern	172
Figure 65 - Multi-Cloud Auditor Pattern.....	174
Figure 66 - Multi-Cloud Control Plane Framework.....	176
Figure 67 - Multi-Cloud Control Plane Framework Implemented with Open Source Components.....	178
Figure 68 - Conceptual Multi-Cloud Web Application / Service Pattern	180
Figure 69 - Multi-Cloud Web Application / Service Pattern Implementation	181
Figure 70 - Conceptual Multi-Cloud Internet of Things Event Stream Ingesting Pattern and Big Data Pipeline	183
Figure 71 - Multi-Cloud Internet of Things Event Stream Ingesting Pattern and Big Data Pipeline Implementation	184
Figure 72 - Containers vs. Virtual Machines	186
Figure 73 - Conceptual View of Container Orchestration Pattern for Multi-Cloud Deployments	187
Figure 74 - Container Orchestration Pattern for Multi-Cloud Deployments Implementation View	188
Figure 75 - Multi-Cloud Control Plane Framework Conceptual View.....	190
Figure 76 - Multi-Cloud Control Plane Framework Implementation View	191
Figure 77 - Multi-Cloud Cloud Fault Tolerant Routing Pattern Validation.....	201
Figure 78 - General DNS Failover Approach.....	204

Figure 79 - Primary and Secondary DNS Failure – API End Points are Used for Failover	205
Figure 80 - Multi-Cloud Cloud Blueprint and Multi-Cloud Deployer Pattern Validation - Conceptual	206
Figure 81 - Multi-Cloud Cloud Blueprint and Multi-Cloud Deployer Pattern Validation - Implementation View	207
Figure 82 - Multi-Cloud Cloud Blueprint and Multi-Cloud Deployer Pattern Sequence Diagram.....	210
Figure 83 - Validation of Multi-Cloud Image Pipeline Pattern	213
Figure 84 - Validation of Multi-Cloud Image Pipeline Pattern - Implementation View	214
Figure 85 - Multi-Cloud Service Registry and Discovery.....	217
Figure 86 - Multi-Cloud Service Registry and Discovery - Implementation View.....	218
Figure 87 - Multi-Cloud Container Orchestration Validation	221
Figure 88 -Multi-Cloud Data Replication Pattern Validation	223
Figure 89 - Multi-Cloud Deployment Framework Comprised of Multi-Cloud Patterns	227
Figure 90 - Multi-Cloud Deployment Framework Comprised of Multi-Cloud Patterns - Implementation View	228

Chapter 1 - Introduction

Most organizations are moving to the cloud or planning to move to the cloud. However, it quickly becomes clear that it's usually not a simple lift and load process moving from on-premises to the cloud.

So, what is one to do when they move to a cloud?

There are multiple stakeholders usually involved in this process:

- *Developers* just want to write code and deploy applications.
- *Operations and Support* professionals want to make sure that the solution is fault tolerant, not brittle and everyone can sleep well at night.
- *Security professionals* want to make sure there are no breaches and data is protected.
- *Business users* generally just want good SLA (Service Level Agreement) without any downtime for the cheapest price possible.

Some initial challenges and questions that are usually asked:

Which cloud provider should I choose?

Should I move to Amazon AWS, Azure, Google or someone else?

Is it just about a price? Should I go with the cheapest option?

What about data safety and security certifications?

My application or process needs a software stack i.e. J2EE, .NET etc. – how do I set this up in the cloud? How do I do it securely? How do I ensure there is no downtime/faults? More importantly, most organizations ask: I would like to move application X or process Y to the cloud – what is the best way of doing it?

Should I pick Infrastructure as a Service (IaaS) or Platform as a Service (PaaS)? Let's briefly illustrate the difference between the two:

Infrastructure as a Service vs. Platform as a Service

What is IaaS?

Infrastructure as a Service is an on-demand outsourced infrastructure service model allowing organizations to rent infrastructure and software (typically Operating System) in the vendor's data center.

What is PaaS?

Platform as a Service

is a software layer above Infrastructure as a Service (i.e. OpenStack) to allow developers to build applications and services without the need to worry about infrastructure setup or operating systems setup etc.

Items in **RED** need to be managed by the Tenant.

Item in **BLUE** are managed by the vendor

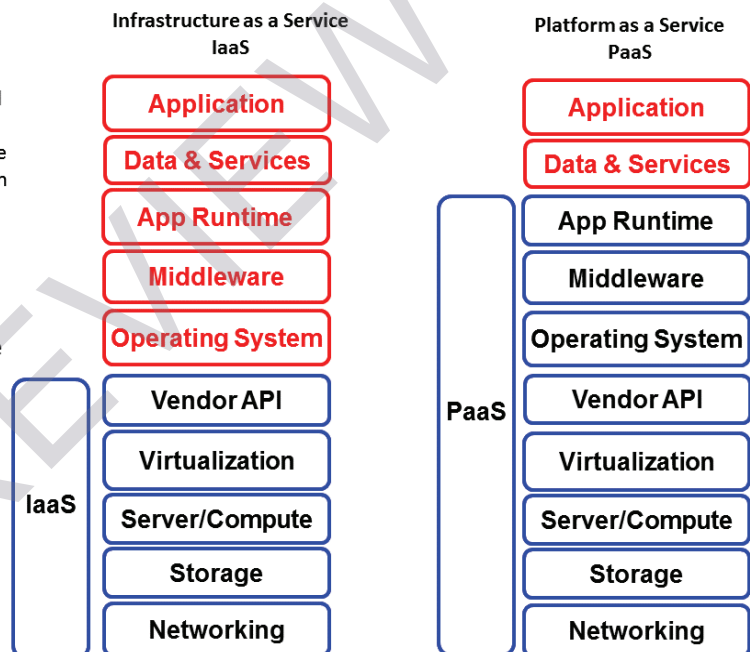
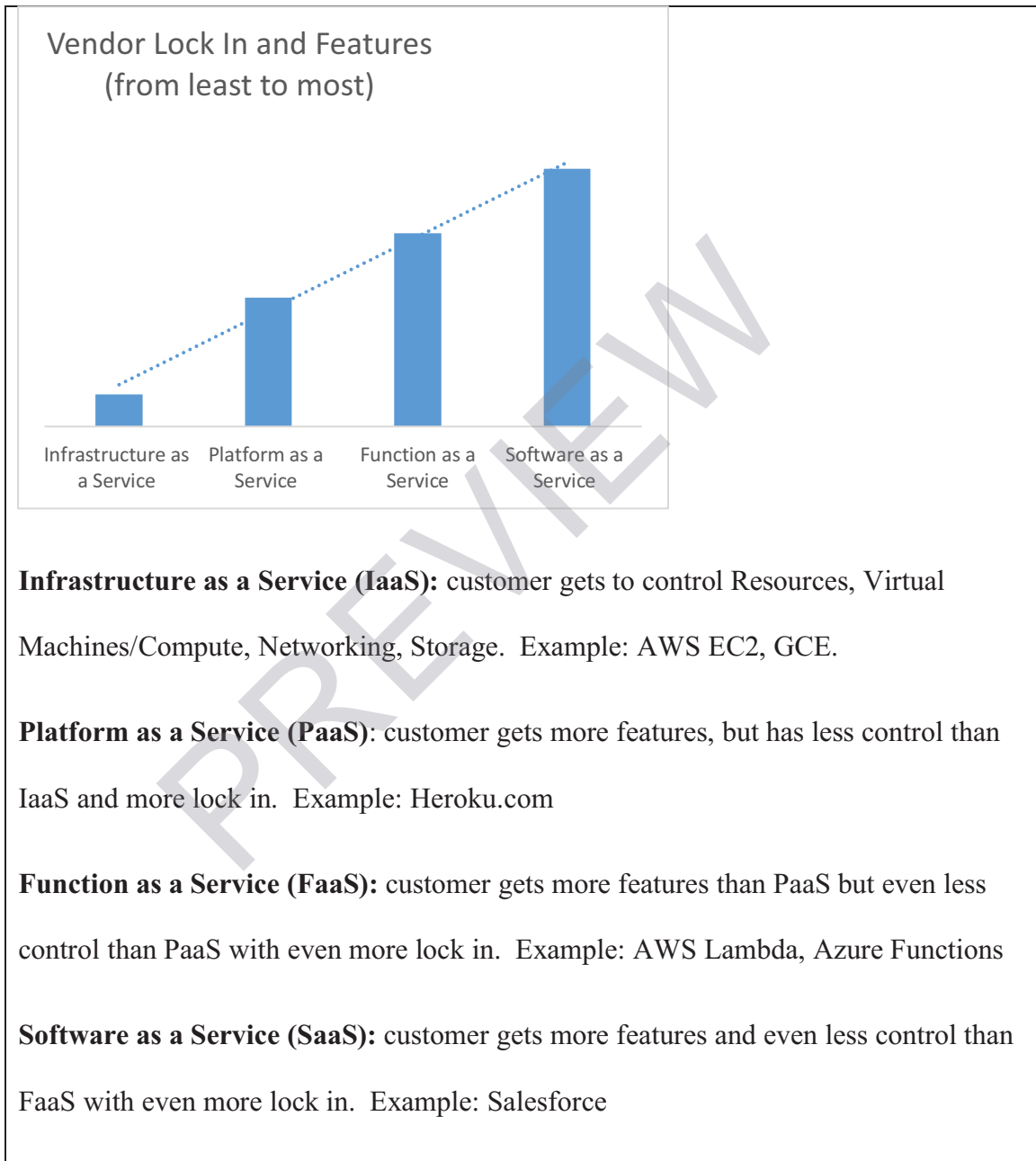


Figure 1 - Infrastructure as a Service vs. Platform as a Service

At this point, confused customers usually go to all cloud providers and every cloud provider promises that all of these will be addressed and they have excellent tools to help.

However, the reality is pretty grim. Even if a customer gets all of the questions answered and picks one cloud provider they end up locked in as most of these tools are proprietary and most cloud providers do not have or support standardized APIs.

The more features, automation and abstractions vendor provides usually results in more of lock-in for the customer. Here is high-level trend of vendor lock in of cloud offerings from least to most:



So is there a better way? It's usually very hard to quantify "better". However, one can argue the better way is to use a solution to a common problem which is generally defined as a pattern. In this research, we will provide patterns for common problems that organizations face when migrating to a cloud which ultimately will comprise multi-cloud framework. In addition, we will focus on building on lowest lock in layer which will be Infrastructure as a Service.

Originally most of the design patterns were software design patterns – i.e. Gang of Four Design Patterns. In this study, we will focus on infrastructure software deployment and design patterns.

What is a software infrastructure deployment pattern? It is a general reusable solution to a commonly occurring problem within a given context in software deployment. By infrastructure software we mean any software that is not application for example web service, application server software, database etc. This is also sometime referred to as software stack. Let us look at illustration of what Infrastructure Software or Stack is:

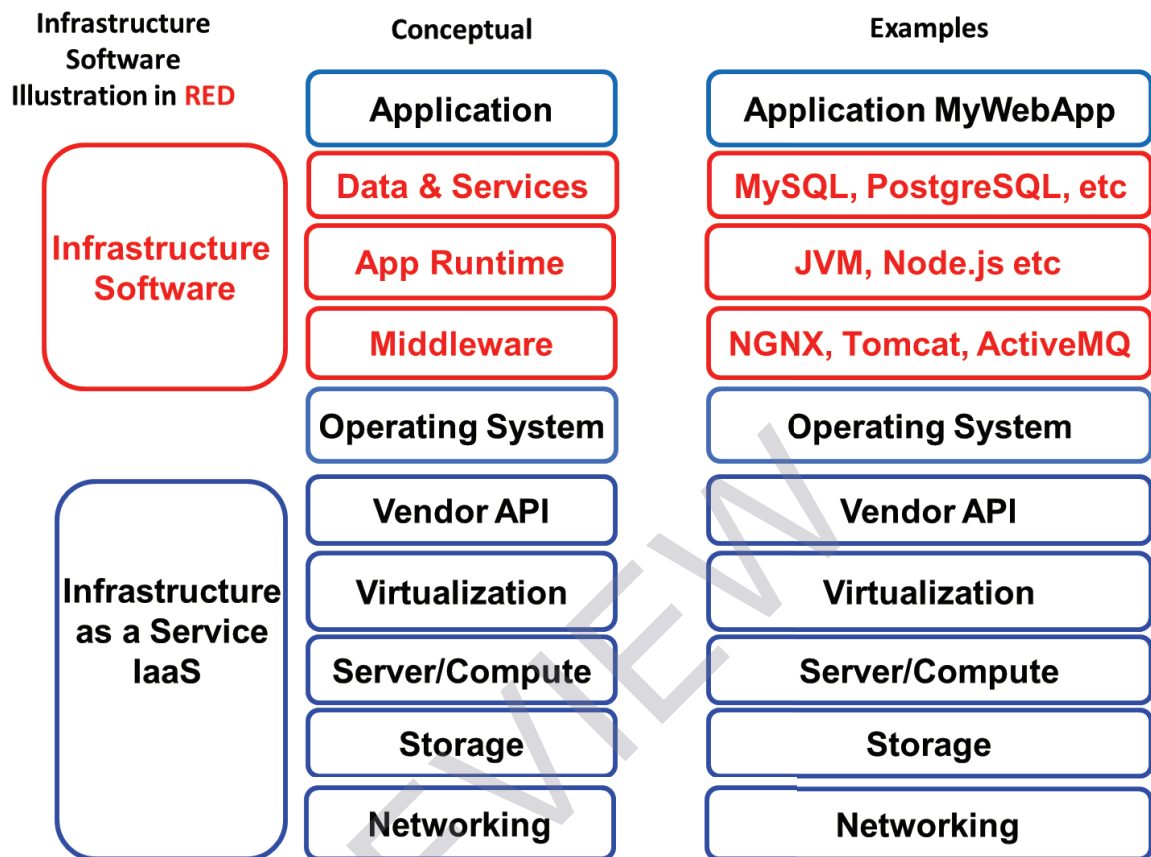


Figure 2 - Infrastructure Software

Applications need a specific software infrastructure stack to run on. Some of the common stacks can be classified as: Dynamic Web Application Stack, Media Streaming, Content Serving, Data Analytics Stack etc. These are examples of deployment stack patterns.

1.1 Multi-Cloud Deployment for Fault-Tolerance, Performance, Security and Cost Efficiency

What is Multi-cloud deployment? It is essentially the use of multiple cloud computing resources/services seamlessly as one coherent architecture. This might include Public and Private Cloud, as well as mix of SaaS, PaaS and IaaS computing resources.

Why does multi-cloud approach matter?

Fault Tolerance and High Availability

Every cloud provider experiences outages. These might be contained to one particular location, but in many cases those locations serve a very large group of clients for example U.S. Northeast or Europe. Many outages have demonstrated that high density deployments in one region with one cloud result in high impact outages to many customers. [Appendix A]. There are even services that track cloud outages in real time <https://cloudharmony.com/status>. (Valid by July 31, 2017)

Here the summary of total hourly impact of outages in 2015 by major cloud providers:

Major Cloud Provider	2015 Downtime in hours, rounded to first decimal
IBM Softlayer	17
Google Cloud Platform	11.5
Microsoft Azure	10.5
Amazon Web Services	2.5
Source:	http://www.networkworld.com/article/3020235/cloud-computing/and-the-cloud-provider-with-the-best-uptime-in-2015-is.html

Figure 3 - 2015 Major Cloud Provider Outage in Hours

This is just a one year example - please reference a more detailed list of major outages of cloud providers for past 3 years:

Appendix A – Major Cloud Provider Outages

Performance

Performance generally has to do with how fast computer systems can perform a task X but more precisely it has to do with following computer system quality attributes: response time, throughput, latency and scalability.

One of the core issues is that not every provider has the same performance profile and at any point of time it might change due to multi-tenant nature of the cloud and because

most of workloads and component are virtualized. For example, let's take hypervisor which runs multiple virtual machines. You will most likely be sharing a hypervisor with multiple tenants and this results in a noisy neighbor problem – where one tenant might negatively impact performance for another tenant sharing the hypervisor.

In addition will focus on ability to scale Up / Vertically within the same Virtual Machine Instance and Out / Creating new instances / components.

Locality

Not every cloud provider has presence in all countries and in many cases there are different regulations that might drive the need to be in specific country. Latency and proximity matters to many use cases. In many cases you might want to run some workloads on premises and some in the cloud which can be described as Hybrid Private/Public Cloud deployment pattern.

Here is a high-level example of Multi-Cloud or Hybrid Deployment Pattern:

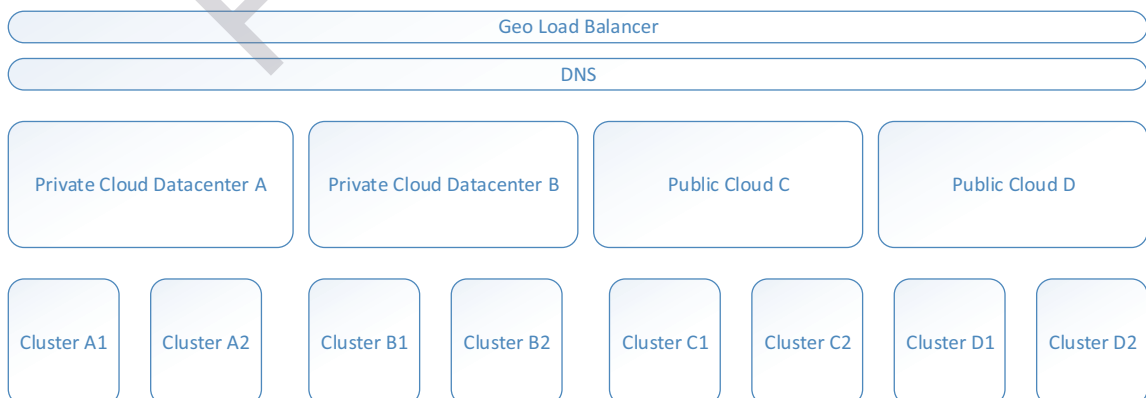


Figure 4 - Multi-Cloud Hybrid Deployment Pattern

Security

Another extremely important factor in any cloud deployment needs to be security. Cloud Providers have different services that solve this, but most of these are non-standard and proprietary which result in cloud provider lock in and customer confusion. One of the key areas we will focus on is how to protect secrets and data at rest. Security arguably should be part of every cloud pattern.

Cost Efficiency

Cloud providers change their prices quite frequently and generally the trend is to lower the price of services. In addition, there are discounted instances that cloud providers offer due to idling capacity. Multi-cloud approach allows you to shift workloads to leverage cheaper prices depending which cloud provider offers the better price. Furthermore, this body of work will demonstrate how this can be dynamically.

1.2 Current Solutions and Their Limitations

There are multiple cloud providers and some provide their own specific deployment patterns.

However, there are couple of problems with vendor provided patterns:

- 1). Patterns provided are usually vendor specific with specific APIs and software stacks that lock in the customer. The vendor generally has no incentive to support standards so the customer can leave at will to another provider. Furthermore, some providers such as AWS have stated that they will not support any partner that offers solution that runs on competitor's cloud platform. [36]
- 2). Generally, these proprietary services also cost more due to the fact that vendor can charge more for proprietary technology and convenience.
- 3). These patterns do not always take in account important factors such as system resilience properties i.e. fault tolerance.
- 4). Other patterns are very high level and theoretical and do not demonstrate the concrete software technology implementation in vendor neutral way i.e. with open source software which make these not very useful.

This leads to vendor lock in, poor deployment design and is prone to faults and poor service availability ultimately resulting in poor end user experience and data loss.

Let's take a look at Microsoft Azure Big Data offerings – all components with exception of HortonWorks Data Platform and HBase are proprietary:

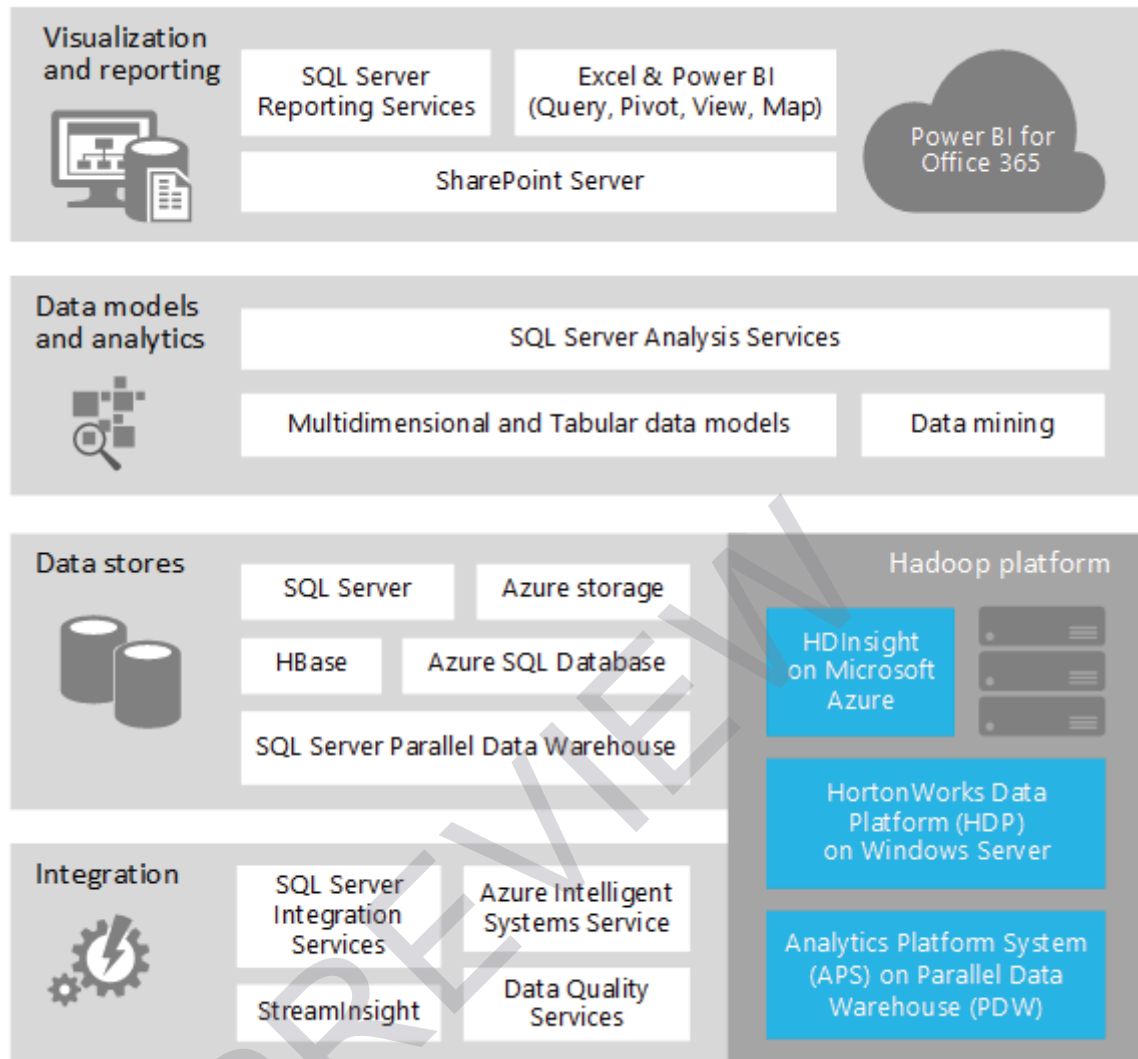


Figure 5 - Microsoft Azure Big Data Offerings Circa 2016

Source: <https://msdn.microsoft.com/en-us/library/dn749804.aspx>