Cloud Computing Patterns

Christoph Fehling • Frank Leymann •
Ralph Retter • Walter Schupeck •
Peter Arbitter

Cloud Computing Patterns

Fundamentals to Design, Build, and Manage Cloud Applications

Springer

Christoph Fehling
University of Stuttgart
Stuttgart
Germany

Frank Leymann
University of Stuttgart
Stuttgart
Germany

Ralph Retter
T-Systems International GmbH
Frankfurt
Germany

Walter Schupeck
Daimler AG
Stuttgart
Germany

Peter Arbitter
Microsoft Deutschland GmbH
Unterschleißheim
Germany

Additional material to this book can be downloaded from http://extra.springer.com.

ISBN 978-3-7091-1567-1 ISBN 978-3-7091-1568-8 (eBook)
DOI 10.1007/978-3-7091-1568-8
Springer Wien Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013955000

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

To Carina and my parents, Elisabeth and Horst. Thank you for your love and support.

– Christoph

To the woman of my dreams!

– Frank

To Anni and Emil.

– Ralph

To my grandsons Leonardo and Max Viktor.

– Walter

To Annika, Paulina, Nils, and my lovely wife Martina.

– Peter

Foreword by Gregor Hohpe

Over the recent years, you may have observed a strategy, dare I say "pattern," for software engineering books: take a popular buzzword and append the word "patterns" to suggest that this particular title contains a substantial treatment of the subject matter and is organized as a collection of easy to digest chapters, which promise to provide solutions to recurring problems. For the pattern community, this is a definite sign of success, and not one that is due to large marketing budgets but rather many successful titles that deliver on this very promise. A recent search on the "world's largest bookstore" in the category "Books – Software Development" yielded no less than 499 titles containing the word "patterns." How does one pick out the gems from such a vast collection of solutions to recurring problems?

First, good patterns books present a coherent pattern language, not just a collection of patterns, in line with Christopher Alexander's seminal title *A Pattern Language*. Structuring a pile of patterns into a sequence of chapters is laudable but does not make a language. Rather, a pattern language covers a specific domain and usually offers multiple patterns for common design challenges and offers a clear road map through the patterns.

Many influential software patterns books have also adopted a strong visual language. I may be biased because *Enterprise Integration Patterns* was one of the first titles to expand the notion of a pattern *sketch* to a visually coherent iconic language, but because software architecture and specifically software patterns aim to communicate complex design ideas in an easy-to-grasp way, a visual language is a strong asset.

Almost all significant software patterns books are not simply the result of the authors' deep insight or determination but have been a true community effort. For example, the Pattern Languages of Programming (PLoP) conference, which has been running since 1994, has been the birthplace of many great patterns books. While it is certainly not a requirement for pattern authors to workshop their papers at PLoP, the claim that patterns are harvested rather than invented points to the importance of community participation. This also means that good patterns books don't usually pop out of nowhere but are the pinnacle of a lengthy process of stepwise refinement based on community feedback.

Last but not least, most successful patterns books strike a delicate balance between academic rigor and real-world applications. The academic world brings depth of thinking and a clear structure, while the industry contributes the required validation and real-world examples.

If you are wondering why the foreword to *Cloud Computing Patterns* muses about the pattern movement as a whole, I invite you to revisit after reading this title. I hope you will agree that I was in fact describing the essential properties of *cloud computing patterns*! I have not much left to say but congratulate the authors on the successful delivery of a relevant, well-structured, and community-validated software patterns title. Because I am sure I won't have to convince you that cloud computing is one of the most relevant software engineering topics of the last decade.

– Gregor Hohpe, coauthor of *Enterprise Integration Patterns* (Hohpe, G., Woolf B.: Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions. Addison-Wesley. http://www.eaipatterns.com/ (2003)).

Gregor Hohpe

Foreword by Robert Hanmer

The forecast for computing is full of clouds. The computer press and the public media are abuzz with "cloud-this" and "cloud-that," but what does it all mean? In many industries, service providers are looking to the cloud to help simplify their hardware management and increase their flexibility and adaptability to change and to provide new and innovative services, but much of what's written about clouds is about the nuts and bolts and offerings of specific cloud providers. The big picture of how to build useful and robust applications to support business needs on top of clouds is missing. Specific case studies are useful, but they can be limiting because they also are tied to specifics. What's needed is a resource to understand the ins and outs of clouds and how they can be used from a top-down perspective. This book is that resource.

I first met the authors at the 2011 Pattern Languages of Programming Conference in Portland, OR, USA. They submitted a paper to the conference that was about early work that ultimately led to this book. Because a paper about clouds was timely, it was one of the most popular papers at the conference. Everyone wanted to learn more about clouds, and all were intrigued by the range of cloud solutions studied to understand the patterns.

Patterns in the style of Christopher Alexander and the *Design Patterns* book [2] (aka the Gang of Four - GOF patterns) are based on real-life experiences and proven practices, not unproven theories. The successes, failures, and "aha's" of real-life solutions were captured in these patterns which were mined from projects across several industries. The authors aren't trying to sell clouds and they aren't just patting themselves on the back because they've built something nice with clouds. They explain clouds based on what they've seen work in practice. The topics range from the basics of on-demand self-service, broad network access, pay per use, and rapid elasticity to complex topics like structuring multi-tier applications and how to best handle issues like fault tolerance and performance.

Patterns explain not just what a cloud is or what you must do to effectively use a cloud but also explain the trade-offs involved in using the cloud. The patterns tie the cloud definitions back to the NIST Definitions of Cloud Computing [3] to thoroughly cover the topic. I agree with the authors that the pattern form is a good way to explain clouds, because patterns explain the design decisions and trade-offs you must be aware of to build an effective cloud solution. The authors' experiences help you to know why things are the way they are as well as highlight how to work around or benefit from that rationale.

You'll find seven chapters in this book. Each chapter contains many patterns, which describe a solution to a problem with an explanation of why that solution is the right one. The chapters build from the basic fundamentals of the NIST definition of a cloud all the way to cloud application shapes and sizes. In between, the authors cover the basics of the different kinds of cloud offerings that you'll find and the basics of designing your solution's cloud architecture and how to manage your cloud. The last chapter talks about problems and pitfalls to avoid, for example, the mismatch between the requirements imposed from the bottom up by the cloud providers and the requirements pressing downward from the business analysts.

I especially like the author's treatment of non-functional requirements. They don't ignore the issue of fault tolerance and robustness which would give you a view of only the sunniest days ahead. This book points out that the rapid elasticity of the cloud is not a guarantee of high reliability by itself but that further design principles like elasticity and resiliency management are needed to ensure that the application achieves the required level of reliability and availability.

I think that this book will help us all better utilize the cloud. And it's a good example of using patterns to explain a complex topic. The scope of computing-related patterns is expanded from the range of technical domains like fault-tolerant software and small memory systems to enterprise applications and then all the way towards structuring complete business processes and environments like we see here with cloud computing.

Cloud-this and cloud-that are explained in this book, which will help you prepare for whatever the cloudy future of computing delivers.

– Robert S. Hanmer, author of *Patterns for Fault Tolerant Software* (Hanmer, R.: Patterns for Fault Tolerant Software. Wiley, Chichester (2007)).

Robert Hanmer

Preface

Which is the right cloud computing provider for my company?
How do I use the cloud computing products of a certain provider and how do they compare to the products I already have in place? We use an infrastructure as a service platform already. What is different now?
How does cloud computing impact existing applications if they are moved to the cloud?
How can I use cloud computing to reduce provisioning and setup times, so my users can access servers more quickly?

During the past years of our collaborative research and consulting on cloud computing, we often encountered these questions or very similar ones. Often, these questions were driven by the need to rapidly produce results in big enterprise cloud initiatives that were founded with the ambitious goals to reduce the cost of IT tremendously and make provisioning of IT resources faster. We also encountered a general trend that cloud computing initiatives are often driven *bottom-up*, starting with infrastructure automation activities. These initiatives, therefore, tend to focus on the IT infrastructure and how to change its setup, deployment, and management and leave application architecture standards for the enterprise untouched. Results are company-internal infrastructure-as-a-service offerings that can provision virtual servers in a fast and efficient manner via a self-service portal. In this book, we describe how cloud computing changed IT infrastructure provisioning and management as well, but mainly we focus on the following questions:

How does cloud computing change the architecture of applications using it efficiently? How can I use cloud infrastructure and platform offerings to efficiently and rapidly design, build, and manage applications to support the changing needs of my business?

Many products found in the cloud computing market display similar functionality that customers may use to build their cloud applications. The discussions of IT architects and developers are often hindered by different product names, provider-specific terminologies, the different distributions of common functionality among products, as well as the fact that some products are merely *cloud-washed*, and thus their name is extended with the term "cloud" to make them more attractive. In our research, we, therefore, analyzed cloud products as well as applications built using different cloud vendors and cloud computing technologies. The goal of this analysis has been to extract common behavior and common components of cloud products as well as the common architecture principles involved to build cloud applications. The result is the book you hold in your hands.

The reason for abstracting from existing cloud providers and cloud applications is to capture provider-independent and sustainable knowledge abstracted from concrete products. This abstract description covers how to build cloud applications, how different cloud products behave, and when to choose a particular form of a cloud offering given a concrete usage scenario.

What You Will Learn and What This Book Is About and Not About

In this book, we build upon our experiences and approach cloud computing principles with a *top-down* mind-set. The goal of this book is not to show you the benefits of cloud infrastructure automation technology over traditional data centers but to show you how cloud principles can be supported on the business and application layers. This book should give you nuggets of advice in the form of patterns that allow you to better understand how you can support cloud properties on the application level and how to select appropriate cloud infrastructure and platform offerings to make your life easier. Furthermore, this book gives IT architects and developers a common vocabulary when discussing about cloud products and architectures of custom cloud applications without focusing on a concrete cloud provider.

During our consulting and research experiences, we often missed the abstraction of cloud definitions from concrete provider offerings and vendor products. Thus, we chose to use a pattern-based approach to describe cloud offerings vendor neutrally in our technical report (Fehling, C., Leymann, F., Mietzner, R., Schupeck, W.: A collection of patterns for cloud types, cloud service models, and cloud-based application architectures. Technical report, University of Stuttgart) in 2011. As this approach proved very popular wherever we presented and discussed it, we decided to revise and extend this approach and the resulting pattern catalogue into what you will find in this book. In order to not lose track with reality, we describe the essential properties of a pattern and then add concrete providers and vendor offerings where we are aware of them in "known uses" sections of the pattern. These sections are by no means exhaustive; they should give you an initial idea on providers and vendors offering a product that exhibits the properties described in the pattern.

If you take this book as a catalogue of patterns that will help you to design, build, and manage *cloud-native* applications as well as select suitable cloud infrastructure and platform offerings, it will provide you with the most value. We wish you as much motivation and satisfaction reading about and applying the patterns in the book as we had collecting and presenting them at various conferences and events. For further information, please also refer to:

http://www.cloudcomputingpatterns.org
Germany

Christoph Fehling
Frank Leymann
Ralph Retter
Walter Schupeck
Peter Arbitter

Trademarks

Acknowledgments

This book would not have been possible without the support of many persons with whom we discussed architectural styles, the individual cloud computing patterns, and cloud computing in general. Even though these discussions may not have been related to the writing of this book directly, they were essential to our understanding of cloud computing, architectural patterns, their creation, and their use.

For a first feedback to our research results, we would like to thank all participants of the writers' workshop held at the 2011 Conference on Pattern Languages of Programs (PLoP). Ernst Oberortner provided excellent feedback on our first patterns and Ralph E. Johnson was a great workshop leader and teacher. Their feedback helped us to present the first book draft at the 2012 Chili Conference on Pattern Languages of Programs (ChiliPLoP), where the book was discussed in detail. Especially, we would like to thank Richard P. Gabriel, Joseph W. Yoder, and Anthony Kwiatkowski for the detailed discussions. For continuous reviews of this book during its writing and significant help, we would also like to thank

Olaf Zimmermann

Robert Hanmer

This book was not only influenced by persons who helped us write and improve it but also by our colleagues and friends with whom we enjoyed doing research in the area of cloud computing. We would like to thank David Schumm, Daniel Schleicher, Oliver Kopp, Tammo van Lessen, Tolga Dalman, Thilo Ewald, and Jochen Rütschlin for the time we worked together. Finally, this book was not only influenced by our work but also by our personal life. We would like to thank our families and friends for supporting us during this time and patiently enduring "book-writing moods," especially Carina Traut, Sven Tschersich, and Anni Retter as well as little Emil.

Contents

1 Introduction 1
1.1 Essential Cloud Computing Properties 3
1.2 Essential Cloud Application Properties 5
1.3 Use of Patterns for Cloud Computing 7
1.4 Pattern Format Used in This Book 9
1.5 Overview of This Book 11
1.6 How to Read This Book. 13

2 Cloud Computing Fundamentals 21
2.1 Overview of Fundamental Cloud Computing Patterns 22
2.2 Application Workloads 23
2.2.1 Static Workload 26
2.2.2 Periodic Workload 29
2.2.3 Once-in-a-Lifetime Workload 33
2.2.4 Unpredictable Workload 36
2.2.5 Continuously Changing Workload 40
2.3 Cloud Service Models 42
2.3.1 Infrastructure as a Service (IaaS) 45
2.3.2 Platform as a Service (PaaS) 49
2.3.3 Software as a Service (SaaS) 55
2.4 Cloud Deployment Models 60
2.4.1 Public Cloud 62
2.4.2 Private Cloud 66
2.4.3 Community Cloud 71
2.4.4 Hybrid Cloud 75

3 Cloud Offering Patterns 79
3.1 Overview of Cloud Offering Patterns 80
3.2 Impact of Cloud Computing Properties on Offering Behavior 81
3.3 Cloud Environments 86
3.3.1 Elastic Infrastructure 87
3.3.2 Elastic Platform 91
3.3.3 Node-Based Availability 95
3.3.4 Environment-Based Availability 98

3.4 Processing Offerings 100
3.4.1 Hypervisor 101
3.4.2 Execution Environment 104
3.4.3 Map Reduce 106
3.5 Storage Offerings 109
3.5.1 Block Storage 110
3.5.2 Blob Storage 112
3.5.3 Relational Database 115
3.5.4 Key-Value Storage 119
3.5.5 Strict Consistency 123
3.5.6 Eventual Consistency 126
3.6 Communication Offerings 131
3.6.1 Virtual Networking 132
3.6.2 Message-Oriented Middleware 136
3.6.3 Exactly-Once Delivery 141
3.6.4 At-Least-Once Delivery 144
3.6.5 Transaction-Based Delivery 146
3.6.6 Timeout-Based Delivery 149

4 Cloud Application Architecture Patterns 151
4.1 Overview of Cloud Application Architecture Patterns 152
4.2 Fundamental Cloud Architectures 155
4.2.1 Loose Coupling 156
4.2.2 Distributed Application 160
4.3 Cloud Application Components 166
4.3.1 Stateful Component 168
4.3.2 Stateless Component 171
4.3.3 User Interface Component 175
4.3.4 Processing Component 180
4.3.5 Batch Processing Component 185
4.3.6 Data Access Component 188
4.3.7 Data Abstractor 194
4.3.8 Idempotent Processor 197
4.3.9 Transaction-Based Processor 201
4.3.10 Timeout-Based Message Processor 204
4.3.11 Multi-Component Image 206
4.4 Multi-Tenancy 208
4.4.1 Shared Component 210
4.4.2 Tenant-Isolated Component 214
4.4.3 Dedicated Component 218
4.5 Cloud Integration 221
4.5.1 Restricted Data Access Component 222
4.5.2 Message Mover 225
4.5.3 Application Component Proxy 228

4.5.4 Compliant Data Replication 231
4.5.5 Integration Provider 234

5 Cloud Application Management Patterns 239
5.1 Overview of Application Management Patterns 240
5.2 Management Components 242
5.2.1 Provider Adapter 243
5.2.2 Managed Configuration 247
5.2.3 Elasticity Manager 250
5.2.4 Elastic Load Balancer 254
5.2.5 Elastic Queue 257
5.2.6 Watchdog 260
5.3 Management Processes 264
5.3.1 Elasticity Management Process 267
5.3.2 Feature Flag Management Process 271
5.3.3 Update Transition Process 275
5.3.4 Standby Pooling Process 279
5.3.5 Resiliency Management Process 283

6 Composite Cloud Application Patterns 287
6.1 Overview of Cloud Application Patterns 288
6.2 Native Cloud Applications 289
6.2.1 Two-Tier Cloud Application 290
6.2.2 Three-Tier Cloud Application 294
6.2.3 Content Distribution Network 300
6.3 Hybrid Cloud Applications 303
6.3.1 Hybrid User Interface 304
6.3.2 Hybrid Processing 308
6.3.3 Hybrid Data 311
6.3.4 Hybrid Backup 314
6.3.5 Hybrid Backend 317
6.3.6 Hybrid Application Functions 320
6.3.7 Hybrid Multimedia Web Application 323
6.3.8 Hybrid Development Environment 326

7 Impact of Cloud Computing Properties 331
7.1 Cloud Computing Properties on Levels of the Application Stack . . . 332
7.1.1 Downwards-Propagation of Requirements 334
7.1.2 Upwards-Propagation of Properties 335
7.1.3 Meet-in-the-Middle for Cloud Properties
and Requirements 335
7.2 Impact of Core Cloud Properties 336
7.2.1 Pay-Per-Use 336
7.2.2 Rapid Elasticity 337
7.2.3 Homogenization 340
7.2.4 Resource Sharing/Multi-Tenancy 342

7.3 Impact of Other Common Cloud Offering Properties 345
7.3.1 Environment-Based Availability 346
7.3.2 Eventual Consistency 349
7.3.3 At-Least-Once Messaging 351

References 353

Index 363

List of Figures

Fig. 1.1 Mapping of chapters to a cloud reference application 12
Fig. 2.1 Pattern map of cloud computing fundamentals 21
Fig. 2.2 Exemplary resource provisioning 24
Fig. 2.3 Exemplary static workload 27
Fig. 2.4 Exemplary periodic workload 30
Fig. 2.5 Exemplary once-in-a-lifetime workload 34
Fig. 2.6 Exemplary unpredictable workload 37
Fig. 2.7 Exemplary continuously changing workload 41
Fig. 2.8 Application stack and associated cloud service models 43
Fig. 2.9 Infrastructure as a service in the application stack 46
Fig. 2.10 Platform as a service in the application stack 50
Fig. 2.11 Software as a service in the application stack 56
Fig. 2.12 Level of elasticity and pay-per-use of different cloud deployment types 61
Fig. 2.13 Public cloud 63
Fig. 2.14 Private cloud, outsourced private cloud, and virtual private cloud 67
Fig. 2.15 Community cloud, outsourced community cloud, and virtual community cloud 72
Fig. 2.16 Hybrid cloud in the cloud scope 76
Fig. 3.1 Pattern map of cloud offerings 79
Fig. 3.2 Eventual consistency scenario 84
Fig. 3.3 Components of an elastic infrastructure 88
Fig. 3.4 Components of an elastic platform 92
Fig. 3.5 Exemplary node-based availability 96
Fig. 3.6 Exemplary environment-based availability assurances 99
Fig. 3.7 Hypervisor types using virtualization and para-virtualization 102
Fig. 3.8 Execution environment in an application stack 105
Fig. 3.9 Elastic map reduce using key-value storage 107
Fig. 3.10 Images of a block storage being mapped to virtual drives 111
Fig. 3.11 Blob storage accessed via HTTP 113
Fig. 3.12 Exemplary relational storage 116
Fig. 3.13 Exemplary read operation on a key-value storage 120
Fig. 3.14 Exemplary strict consistent replicas 124

Fig. 3.15 Exemplary eventual consistent replicas 127
Fig. 3.16 Self-service interface for configuration of virtual networking 133
Fig. 3.17 Exemplary VLAN setup 134
Fig. 3.18 Firewall setup for an example application 134
Fig. 3.19 VPN from a corporate network to an IaaS provider 135
Fig. 3.20 Message-oriented middleware and related patterns 137
Fig. 3.21 Message filter used to guarantee exactly-once delivery 142
Fig. 3.22 Communication between a sender and a receiver to ensure at-least-once delivery 145
Fig. 3.23 Operations of the transactional reception of a message 147
Fig. 3.24 Operations of timeout-based reception of a message 150
Fig. 4.1 Map of the cloud application architecture patterns 151
Fig. 4.2 Realization of loose coupling through an intermediary 158
Fig. 4.3 Exemplary decomposition into three tiers 162
Fig. 4.4 Process-based decomposition 162
Fig. 4.5 Pipes-and-filters-based decomposition 162
Fig. 4.6 Stateful application components 169
Fig. 4.7 Stateless application components 172
Fig. 4.8 User interface components in a common setup 176
Fig. 4.9 Integration of different portlets into a portal 177
Fig. 4.10 Processing component in a standard setup 181
Fig. 4.11 Exemplary pipes-and-filters video processing application 182
Fig. 4.12 Exemplary process-based video processing application 183
Fig. 4.13 Batch processing component in a standard setup 186
Fig. 4.14 Data access components integrating stateful components and storage offerings residing in two clouds 189
Fig. 4.15 Data access component interface and data structure 191
Fig. 4.16 Exemplary data abstractions 195
Fig. 4.17 Idempotent component for messaging (*left*) and storage offerings (*right*) 199
Fig. 4.18 Operations of the transactional processing of a message (*left*) and data (*right*) 202
Fig. 4.19 Operations of the timeout-based processing of a message 205
Fig. 4.20 Two application components managed as one multi-component image 207
Fig. 4.21 Clients accessing a shared component 211
Fig. 4.22 Clients accessing a tenant-isolated component 216
Fig. 4.23 Table-based tenant isolation (*left*) and row-based tenant isolation (*right*) 216
Fig. 4.24 Clients accessing dedicated components 219
Fig. 4.25 A data access component and a restricted data access component providing data to a secure and an insecure environment 223
Fig. 4.26 Message mover integrating queues of two environments 226
Fig. 4.27 Application component proxy bridging two environments 229

Fig. 4.28 Compliant data replication between a secure cloud and an insecure cloud 232
Fig. 4.29 Integration provider integrating two private environments 235
Fig. 5.1 Map of the cloud application management patterns 239
Fig. 5.2 Abstract management architecture 241
Fig. 5.3 Exemplary provider adapter component accessed synchronously 244
Fig. 5.4 Exemplary provider adapter component accessed asynchronously 245
Fig. 5.5 Polling (*left*) and pushing (*right*) of managed configurations 248
Fig. 5.6 Elasticity manager interacting with an elastic platform or elastic infrastructure 251
Fig. 5.7 Elastic load balancer interacting with an elastic platform or elastic infrastructure 255
Fig. 5.8 Elastic queue interacting with an elastic platform or an elastic infrastructure 259
Fig. 5.9 Watchdog supervising application components hosted on IaaS and PaaS 261
Fig. 5.10 Elasticity management flow 268
Fig. 5.11 Feature flags management process 273
Fig. 5.12 Update transition process for application components accessed through load balancers and queues 277
Fig. 5.13 Decommissioning executed by the standby pooling process and interacting components 280
Fig. 5.14 Provisioning executed by the standby pooling process and interacting components 281
Fig. 5.15 Resiliency process handled by the watchdog 285
Fig. 6.1 Map of composite patterns for cloud computing 287
Fig. 6.2 Exemplary architecture of a two-tier cloud application 291
Fig. 6.3 Exemplary architecture of a three-tier cloud application 295
Fig. 6.4 Content distribution from a storage offering to two cloud environments 301
Fig. 6.5 Hybrid user interface in an elastic cloud and a static data center 305
Fig. 6.6 Hybrid processing in a static data center and an elastic cloud 309
Fig. 6.7 Hybrid data residing in a static and an elastic environment 312
Fig. 6.8 Hybrid backup using an elastic cloud to archive data 315
Fig. 6.9 Hybrid backend using a static data center and an elastic cloud 318
Fig. 6.10 Hybrid UI, processing, and data deployment 321
Fig. 6.11 Hybrid multimedia web application residing in an elastic cloud and a static data center 324
Fig. 6.12 Hybrid development environment and its integration with a production environment 327
Fig. 7.1 Propagation of requirements in the application stack 333

Fig. 7.2 Workloads and elasticity requirements 337
Fig. 7.3 Elasticity on different levels of the stack 338
Fig. 7.4 Resource sharing/multi-tenancy on different levels in the stack 343
Fig. 7.5 Mitigation of environment-based availability on higher levels of the application stack 346