

Engineering Scalable, Elastic, and Cost-Efficient Cloud Computing Applications

Steffen Becker • Gunnar Brataas • Sebastian Lehrig
Editors

Engineering Scalable, Elastic, and Cost-Efficient Cloud Computing Applications

The CloudScale Method

Editors

Steffen Becker
Reliable Software Systems Group
University of Stuttgart
Stuttgart, Germany

Gunnar Brataas
Software Engineering, Safety & Security
SINTEF Digital
Trondheim, Norway

Sebastian Lehrig
IBM Research
Dublin, Ireland

ISBN 978-3-319-54285-0 ISBN 978-3-319-54286-7 (eBook)
DOI 10.1007/978-3-319-54286-7

Library of Congress Control Number: 2017937065

© Springer International Publishing AG 2017

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Foreword

The digitization of the world around us also impacts IT systems themselves. The increased connectivity of everything and everybody with IT services creates challenges in software design. Systems are becoming increasingly distributed, open, adaptive, dynamic, and mobile.

As a consequence for software quality engineering, run-time factors are increasingly determining the quality of a system. Formerly, assumptions on the run-time context could be taken as static and hence be modeled at design time. Such models were then used in software quality analysis. Nowadays, the context and the system itself are more dynamic. The context changes, and the system, being adaptive, also reflects such changes.

This means, we have no fixed moment during design when system properties are fixed and can be determined by a static analysis. For software quality, this means, instead of proven design-time properties, we need to shift to a means to deal with this dynamicity at run-time. For example, instead of proven security, we talk about highly resilient systems; instead of performance, we talk about scalability.

However, all such measures to increase the resilience of scalability need to be built into the software at design time. Hence, techniques that are able to evaluate at design time the effects of such run-time measures and run-time system properties are of interest. While this may sound impossible, the approach shown in this book demonstrates the feasibility and applicability of this design-time evaluation of quality properties depending on run-time factors in the domain of performance engineering. This book is therefore overdue. After the advent of cloud-based systems in the beginning of the century, this trend of cloud computing is even enforced by the ubiquitous use of several mobile devices, which all need to work with consistent data. Due to the mobility of the devices and the highly fluctuating number of users, the workload of the cloud drastically varies.

This underpins the relevance of the methods presented in this excellent book by outstanding researchers in this field of scalability modeling and analysis. There is no book before this that enlightens us so much about this shift from performance

to scalability! I hope you can read it with the same gain as I did and with the same joy—of seeing Palladio being used and advanced!

Karlsruhe, KIT and FZI
Germany
December 2016

Professor Dr. Ralf Reussner
Chair Software Design and Quality, KIT
Executive Director FZI

Preface

Berlin, 2017: The start-up company SmartService has built an application allowing users to manage subscriptions to business services, magazines, etc. and to send out cancellations automatically and on time. The service is implemented based on a cloud computing environment and uses various third-party services, for example, a service to convert images of contracts uploaded by users into PDF. So far, SmartService has assumed that using a cloud platform will allow it to scale its application as needed without further actions. With this application, SmartService has found a promising niche. The application has spread rapidly, showing a perfect growth of a hockey stick-shaped curve.

Unfortunately, the rising number of users leads to ever-increasing infrastructure costs and, especially during peak loads, to high end-to-end response times, which result in customer losses. The application faces some severe scalability issues. If they cannot fix the problem soon, the whole start-up will be at risk. SmartService needs to address the problem quickly.

Did you ever wonder how to engineer cloud computing services that are scalable, elastic, and cost-efficient—just like the above fictional scenario about the SmartService company?

In this book we describe a detailed method—the CloudScale method—for ensuring that services running on the cloud achieve exactly these properties, ideally by design. With the CloudScale method, software architects can analyze both existing and planned IT services. The method allows to answer questions like:

- With an increasing number of users, can my service still deliver an acceptable quality of service?
- What if each user uses the service more intensively, can my service still handle it with an acceptable quality of service?
- What if the number of users suddenly increases, will my service still handle it?
- Will my service be cost-efficient?

Continuing SmartService’s scenario, the CloudScale method allows them to analyze the scalability problem in detail and identify scalability anti-patterns and bottlenecks within its application. Using the method, SmartService quickly realizes that its scalability problems are caused by conservative handling of data in the realization of the application. SmartService uses CloudScale’s scalability know-how and applies the CloudScale method to find the best scalable architecture. Equipped with this knowledge, the company swiftly restructures its application and continues its path of growth successfully. In the future, CloudScale’s method and tools will allow SmartService to avoid any such critical scalability problems right from the beginning.

When we, the CloudScale EU project,¹ began our work about 4 years ago, we pursued the vision to provide assistance to stakeholders involved in engineering scalable, elastic, and cost-efficient cloud computing services. The “Berlin, 2017” scenario about the SmartService company describes our ideas pretty well. You might still consider such a scenario purely fictional. But, in fact, it is realistic!

For example, the initial design of SAP’s BusinessByDesign system had severe scalability issues.² As a consequence, SAP suffered from significant financial losses and had to reimplement large parts of the system before it was ready to be released. Another recent famous example is the US government’s healthcare system.³ The system broke down under the load caused by users trying to gather information about their health insurance during the first weeks. Officials had not tested the system with the load the system finally had to face. Also online games regularly face scalability issues during the week of a new game or add-on release. For example, in World of Warcraft, Blizzard regularly faces issues on new game releases despite its extensive experience in this business domain.⁴

Motivated by such scenarios, we began our work on the CloudScale method. You, the reader of this book, have the final outcome of our efforts in your hands: This book gives you an overview of the problems involved in engineering scalable, elastic, and cost-efficient cloud computing services and describes the CloudScale method—a description of rescue tools and the required steps to exploit these tools.

This book is meant for all stakeholders interested in achieving our vision: managers, product owners, software architects, developers, testers, operational personnel, etc. With this book, they can both see the overall picture and drill into issues of particular interest.

¹www.cloudscale-project.eu.

²www.v3.co.uk/v3-uk/news/1970547/sap-update-business-bydesign-plans.

³www.informationweek.com/healthcare/policy-and-regulation/why-healthcaregov-failed/d/d-id/1112064.

⁴www.ibtimes.com/battlenet-servers-down-world-warcraft-hearthstone-overwatch-players-around-world-2383254.

This book presents results developed by a motivated group of researchers from different companies, originating from various countries. Not all of them participated in writing this book. However, this book would not exist without their contributions. We thank all CloudScale members for this effort.

Chemnitz, Germany
Trondheim, Norway
Dublin, Ireland
December 2016

Steffen Becker
Gunnar Brataas
Sebastian Lehrig

Acknowledgments

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007–2013) under grant number 317704 (CloudScale) and the Norwegian Research Council under grant number 256669 (ScrumScale). Richard Sanders and Gregor Pipan contributed with proofreading.

Gregor Pipan

XLAB d.o.o.

Pot za Brdom 100, 1000 Ljubljana, Slovenia

e-mail: gregor.pipan@xlab.si

Richard Torbjørn Sanders

SINTEF Digital

Strindvegen 4, 7034 Trondheim, Norway

e-mail: Richard.Sanders@sintef.no

Contents

Part I Introduction and Overview

1	Introduction	3
	Steffen Becker, Gunnar Brataas, Mariano Cecowski, Darko Huljenić, Sebastian Lehrig, and Ivana Stupar	
1.1	Getting It Right	4
1.2	Software in the Cloud Computing Era	5
1.3	Some Useful Definitions to Characterize Services	7
1.3.1	Operations	7
1.3.2	Service-Level Objectives	8
1.3.3	Workload	8
1.3.4	Capacity	9
1.4	Quality Properties of Services	9
1.4.1	Scalability	10
1.4.2	Elasticity	10
1.4.3	Cost-Efficiency	11
1.5	Consequences of Scalability, Elasticity, and Cost-Efficiency Issues	12
1.6	Causes of Scalability, Elasticity, and Cost-Efficiency Issues	13
1.7	How Should You Manage Scalability, Elasticity, and Cost-Efficiency?	13
1.8	Reactive Scalability, Elasticity, and Cost-Efficiency Management	14
1.8.1	Immediate Temporal Solutions	14
1.8.2	Long-Term Solutions	15
1.9	Proactive Scalability, Elasticity, and Cost-Efficiency Management	16
1.10	The CloudScale Method	17
1.11	What Does It Cost?	18

1.12	What Do You Need?	19
1.13	Conclusion	20
	References	21
2	CloudScale Method Quick View	23
	Gunnar Brataas, Steffen Becker, Mariano Cecowski, Darko Huljenić, Sebastian Lehrig, and Ivana Stupar	
2.1	Process Steps of the CloudScale Method	24
2.2	Running Example	26
2.3	Identify Service-Level Objectives, Critical Use Cases, and Key Scenarios	27
2.3.1	Service-Level Objectives	27
2.3.2	Critical Use Cases	28
2.3.3	Key Scenarios	28
2.4	Identify Scalability, Elasticity, and Cost-Efficiency Requirements	29
2.4.1	Scalability Requirements	30
2.4.2	Elasticity Requirements	30
2.4.3	Cost-Efficiency Requirements	31
2.5	Specify ScaleDL Model	32
2.6	Use Analyzer	32
2.6.1	Scalability Analysis	32
2.6.2	Elasticity Analysis	33
2.6.3	Cost-Efficiency Analysis	34
2.7	Use Spotters	34
2.8	Realize, Deploy, and Operate System	35
2.9	Cloud Computing HowTos	35
2.10	Cloud Computing HowNotTos	37
2.11	The CloudScale Method in the Unified Process	41
2.11.1	Unified Processes	41
2.11.2	Relating the CloudScale Method	42
2.12	Conclusion	43
	References	43

Part II Modeling Cloud Computing Applications

3	Cloud Computing Applications	47
	Mariano Cecowski, Steffen Becker, and Sebastian Lehrig	
3.1	Introduction	48
3.2	Web Applications	49
3.3	Cloud Computing Characteristics	51
3.4	From Web to Cloud Computing Applications	52
3.5	Requirements of Cloud Computing Applications	53
3.6	Modeling Cloud Computing Applications	54
3.6.1	Common View Types for Applications	54
3.6.2	Cloud-Specific View Types for Applications	55

3.7	CloudStore Running Example	56
3.8	Modeling Hints	58
3.9	Conclusion	59
	References	60
4	ScaleDL	61
	Gunnar Brataas, Steffen Becker, Mariano Cecowski, Vito Čuček, and Sebastian Lehrig	
4.1	Introduction	62
4.2	Overview Model	62
4.2.1	Concepts of Overview Model	63
4.2.2	Example of Overview Model	64
4.2.3	Tool Support for Overview Model	65
4.3	Usage Evolution	65
4.3.1	Concepts for Usage Evolution	66
4.3.2	Example of Usage Evolution	67
4.3.3	Tool Support for Usage Evolution	69
4.4	Architectural Templates	69
4.4.1	Concepts of Architectural Templates	70
4.4.2	Example for Architectural Templates	70
4.4.3	Catalog of Architectural Templates	72
4.4.4	Tool Support for Architectural Templates	73
4.5	The Extended Palladio Component Model	73
4.5.1	Concepts of the Extended Palladio Component Model	74
4.5.2	Example for the Extended Palladio Component Model	76
4.5.3	Tool Support for the Extended Palladio Component Model	78
4.6	Conclusion	81
	References	81

Part III The CloudScale Method for Software Architects

5	The CloudScale Method	85
	Gunnar Brataas and Steffen Becker	
5.1	Introduction	86
5.2	Granularity	86
5.3	Method Notation	88
5.4	Roles in the Method	89
5.5	Method Steps	90
5.6	Identify Service-Level Objectives, Critical Use Cases, and Key Scenarios	92
5.7	Identify Scalability, Elasticity, and Cost-Efficiency Requirements	94
5.8	Use-Case I: Analyzing a Modeled System	96
5.9	Use-Case II: Analyzing and Migrating an Implemented System	97
5.10	Realize, Deploy, and Operate	98

5.11	Conclusion	99
	References	99
6	Analyzing a Modeled System	101
	Sebastian Lehrig, Gunnar Brataas, Mariano Cecowski, and Vito Čuček	
6.1	Introduction	102
6.2	Step I: Specify ScaleDL Model	102
6.2.1	Determine Granularity	104
6.2.2	Specify Usage Evolution	106
6.2.3	Specify Overview Model and Generate Extended Palladio Component Model	109
6.2.4	Complete Extended Palladio Component Model	111
6.2.5	Summary for the Specification of ScaleDL Models	115
6.3	Step II: Use Analyzer	115
6.3.1	Set Configuration Parameters	117
6.3.2	Run Analyzer and Assess Requirements	118
6.4	Analyzer Running Example	119
6.4.1	Step I: Specifying a CloudStore Model via ScaleDL	119
6.4.2	Step II: Using the Analyzer with the CloudStore Model ...	123
6.5	Conclusion	127
	References	128
7	Analyzing and Migrating an Implemented System	131
	Steffen Becker and Sebastian Lehrig	
7.1	Introduction	132
7.2	Spotting HowNotTos	133
7.3	Statically Detecting HowNotTos	136
7.4	Dynamically Detecting HowNotTos	138
7.5	Resolving HowNotTos with HowTos	140
7.6	Spotter Running Example	141
7.6.1	Static Spotter	141
7.6.2	Dynamic Spotter	143
7.7	Conclusion	145
	References	146
Part IV Making the CloudScale Method Happen		
8	The CloudScale Method for Managers	149
	Steffen Becker, Gunnar Brataas, Mariano Cecowski, Darko Huljenić, Sebastian Lehrig, and Ivana Stupar	
8.1	Introduction	150
8.2	Key Considerations	150
8.3	Relation to Other Engineering Methods	152
8.4	Pros and Cons of the CloudScale Method	154
8.4.1	Critical Success Factors for Method Adoption and Use ...	154

8.4.2	Organizational Issues	156
8.4.3	Costs	156
8.4.4	Covering the Cost of the CloudScale Method Adoption ...	157
8.4.5	Risks	158
8.4.6	Critical Factors for Successful Projects	158
8.5	A Pilot Project	159
8.6	Setting Up the CloudScale Environment	161
8.7	Complementing Tools	162
8.8	Following the CloudScale Method for the Pilot Project	163
8.9	Conclusion	164
	References	165
9	Case Studies	167
	Darko Huljenić, Ivana Stupar, and Mariano Cecowski	
9.1	Case Study: Electronic Health Record	167
9.1.1	Electronic Health Record	168
9.1.2	Applying the CloudScale Method and Tools to Electronic Health Record	170
9.1.3	Discussion of the Electronic Health Record Case	176
9.2	Case Study: Kantega's Flyt CMS	177
9.2.1	Flyt CMS	177
9.2.2	Applying the CloudScale Method and Tools to Flyt CMS	179
9.2.3	Discussion of the Flyt CMS Case	180
9.3	Additional Case Studies for the CloudScale Method	181
9.4	Conclusion	182
	References	183
	Glossary	185
	Index	187

Contributors

Steffen Becker University of Stuttgart, Stuttgart, Germany

Gunnar Brataas SINTEF Digital, Trondheim, Norway

Mariano Cecowski XLAB d.o.o., Ljubljana, Slovenia

Vito Čuček XLAB d.o.o., Ljubljana, Slovenia

Darko Huljenić Ericsson Nikola Tesla, Zagreb, Croatia

Sebastian Lehrig IBM Research, Dublin, Ireland

Ivana Stupar Ericsson Nikola Tesla, Zagreb, Croatia

Engineering Scalable, Elastic, and Cost-Efficient Cloud
Computing Applications

The CloudScale Method

Becker, S.; Brataas, G.; Lehrig, S. (Eds.)

2017, XIX, 190 p. 54 illus., 31 illus. in color., Hardcover

ISBN: 978-3-319-54285-0