

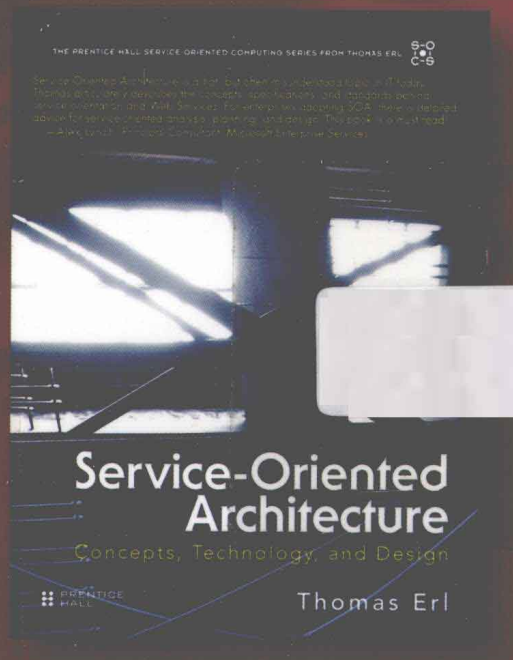
服 务 计 算 技 术 丛 书

PEARSON

SOA概念、技术与设计

(英文版)

(美) Thomas Erl 著



Service-Oriented Architecture:
Concepts, Technology, and Design



科学出版社

服务计算技术丛书

SOA 概念、技术与设计

(英文版)

Service-Oriented Architecture:
Concepts, Technology, and Design

〔美〕 Thomas Erl 著

科学出版社

北 京

图字: 01-2012-0463

内 容 简 介

本书是一本关于面向服务架构(SOA)、Web Service 和面向服务技术的综合教程,系统介绍了 SOA 的概念、技术和设计。书中用 125 个研究实例和 300 多幅插图介绍了建立 SOA 平台所需要的最重要的几个方面:目标、障碍、概念、技术、标准、交付策略、分析和设计的流程等。本书旨在帮助读者充分理解现代 SOA 的组成,以及成功实现 SOA 的步骤。全书通俗易懂,用比喻的方式描述复杂的概念,便于读者理解。

本书可供 SOA 领域的软件架构师、高级软件工程师、分析师、应用科研人员等参考学习。

Original edition, entitled SERVICE-ORIENTED ARCHITECTURE: CONCEPTS, TECHNOLOGY, AND DESIGN, 1E, 9780131858589 by ERL, THOMAS, published by Pearson Education, Inc, publishing as Prentice Hall, Copyright © 2006 Pearson Education, Inc.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

China edition published by PEARSON EDUCATION ASIA LTD., and CHINA SCIENCE PUBLISHING & MEDIA LTD. (SCIENCE PRESS) Copyright © 2012.

Authorized for sale and distribution in the People's Republic of China exclusively (except Taiwan, Hong Kong SAR and Macau SAR). 本版本仅可在中国地区(不包括香港、澳门和台湾)销售发行。

本书封面贴有 Pearson Education (培生教育出版集团)激光防伪标签。无标签者不得销售。

图书在版编目(CIP)数据

SOA 概念、技术与设计 = Service-Oriented Architecture Concepts, Technology, and Design : 英文/(美) 艾尔 (Erl, T.) 著. —北京: 科学出版社, 2012

(服务计算技术丛书)

ISBN 978-7-03-033642-2

I. ①S… II. ①艾… III. ①互联网络—网络服务器—英文 IV. ①TP368.5

中国版本图书馆 CIP 数据核字(2012)第 030653 号

责任编辑: 朱雪玲 / 责任印制: 赵 博 / 封面设计: 迷底书装

科 学 出 版 社 出 版

北京东黄城根北街16号

邮政编码: 100717

<http://www.sciencep.com>

双 青 印 刷 厂 印 刷

科学出版社发行 各地新华书店经销

*

2012 年 3 月第 一 版 开本: B5(720×1000)

2012 年 3 月第一次印刷 印张: 49

字数: 782 000

定价: 115.00 元

(如有印装质量问题, 我社负责调换)

Preface

Authoring this book involved nearly a year of writing, research, and staying on top of a subject matter that is constantly expanding its reach and importance. Although the majority of the chapters focus on service-oriented architecture from a vendor-neutral perspective, achieving an accurate representation of this perspective required that I spend a great deal of time evaluating SOA support in all primary vendor platforms. As part of this research stage I spoke with more than a hundred senior IT professionals, either through interviews or through my work as an awards judge evaluating platform submissions.

One of the most interesting facets of this project has been in defining service-orientation within the context of Web services. While studying the individual parts of what constitutes service-orientation as a paradigm, I came to realize just how many of its roots lie in past innovations. Yet at the same time, it is distinct, blending traditional and new concepts in support of a unique architectural model.

Despite its apparent “newness,” SOA, on a fundamental level, is based on a very old and established school of thought. Service-orientation, as a means of separating things into independent and logical units, is a very common concept. As I progressed through these chapters, I began to notice this more often in everyday life. Items, people, organizations we come into contact with either offer some form of service or participate in performing a service. Once applied to technology architecture, though, service-orientation is concerned with a specific part of our service-oriented world: business automation.

Competitive business climates demand that corporations minimize redundant effort and maximize the expediency with which strategic goals can be achieved. Inefficient organizations that consistently waste resources are bound to fall behind. The manner in

which an organization automates its business is a critical factor in determining the level of efficiency at which it operates and, ultimately, the extent of success it attains in its ventures.

This is what makes SOA so valuable. By shaping automation logic through service-orientation, existing investments can be leveraged, business intelligence can be accurately expressed, and inherent automation agility can be achieved. When coupled with the Web services technology platform, SOA offers a significant and real benefit potential that can transform the technology and outlook of an organization. My goal for this book is to help you explore, understand, and realize this potential.

Acknowledgments

While writing this book I was blessed with a strong team of technical reviewers and superior editorial, production, and marketing professionals. My thanks to all of you for your tireless efforts. A special thanks to my family for their patience and unwavering support.

Contents

Preface

xix

Chapter 1

Introduction	1
1.1 Why this book is important	2
1.1.1 The false SOA	2
1.1.2 The ideal SOA	3
1.1.3 The real SOA	4
1.2 Objectives of this book	4
1.2.1 Understanding SOA, service-orientation, and Web services	5
1.2.2 Learning how to build SOA with Web services	5
1.3 Who this book is for	6
1.4 What this book does not cover	6
1.5 How this book is organized	7
1.5.1 Part I: SOA and Web Services Fundamentals	8
1.5.2 Part II: SOA and WS-* Extensions	10
1.5.3 Part III: SOA and Service-Oriented	13
1.5.4 Part IV: Building SOA (Planning and Analysis)	14
1.5.5 Part V: Building SOA (Technology and Design)	16
1.5.6 Conventions	19
1.6 Additional information	19
1.6.1 The XML & Web Services Integration Framework (XWIF)	19
1.6.2 www.serviceoriented.ws	20
1.6.3 Contact the Author	20

Chapter 2

Case Studies	21
2.1 How case studies are used	22
2.1.1 Style characteristics	22
2.1.2 Relationship to abstract content	22
2.1.3 Code samples	23
2.2 Case #1 background: RailCo Ltd.	23
2.2.1 History	23
2.2.2 Technical infrastructure	23
2.2.3 Automation solutions	24
2.2.4 Business goals and obstacles	24
2.3 Case #2 background: Transit Line Systems Inc.	25
2.3.1 History	26
2.3.2 Technical infrastructure	26
2.3.3 Automation solutions	27
2.3.4 Business goals and obstacles	27

Part I

SOA and Web Services Fundamentals	29
--	-----------

Chapter 3

Introducing SOA	31
3.1 Fundamental SOA	32
3.1.1 A service-oriented analogy	32
3.1.2 How services encapsulate logic	33
3.1.3 How services relate	35
3.1.4 How services communicate	35
3.1.5 How services are designed	36
3.1.6 How services are built	37
3.1.7 Primitive SOA	38
3.2 Common characteristics of contemporary SOA	40
3.2.1 Contemporary SOA is at the core of the service-oriented computing platform	41
3.2.2 Contemporary SOA increases quality of service	42

3.2.3	Contemporary SOA is fundamentally autonomous	42
3.2.4	Contemporary SOA is based on open standards	43
3.2.5	Contemporary SOA supports vendor diversity	43
3.2.6	Contemporary SOA promotes discovery	44
3.2.7	Contemporary SOA fosters intrinsic interoperability	45
3.2.8	Contemporary SOA promotes federation	45
3.2.9	Contemporary SOA promotes architectural composability	46
3.2.10	Contemporary SOA fosters inherent reusability	47
3.2.11	Contemporary SOA emphasizes extensibility	48
3.2.12	Contemporary SOA supports a service-oriented business modeling paradigm	48
3.2.13	Contemporary SOA implements layers of abstraction	49
3.2.14	Contemporary SOA promotes loose coupling throughout the enterprise	50
3.2.15	Contemporary SOA promotes organizational agility	51
3.2.16	Contemporary SOA is a building block	52
3.2.17	Contemporary SOA is an evolution	53
3.2.18	Contemporary SOA is still maturing	53
3.2.19	Contemporary SOA is an achievable ideal	53
3.2.20	Defining SOA	54
3.2.21	Separating concrete characteristics	55
3.3	Common misperceptions about SOA	56
3.3.1	“An application that uses Web services is service-oriented.”	56
3.3.2	“SOA is just a marketing term used to re-brand Web services.”	57
3.3.3	“SOA is just a marketing term used to re-brand distributed computing with Web services.”	57
3.3.4	“SOA simplifies distributed computing.”	57
3.3.5	“An application with Web services that uses WS-* extensions is service-oriented.”	58
3.3.6	“If you understand Web services you won’t have a problem building SOA.”	58
3.3.7	“Once you go SOA, everything becomes interoperable.”	59
3.4	Common tangible benefits of SOA	59
3.4.1	Improved integration (and intrinsic interoperability)	60
3.4.2	Inherent reuse	60
3.4.3	Streamlined architectures and solutions	61
3.4.4	Leveraging the legacy investment	61
3.4.5	Establishing standardized XML data representation	62

- 3.4.6 Focused investment on communications infrastructure 63
 - 3.4.7 “Best-of-breed” alternatives 63
 - 3.4.8 Organizational agility 63
- 3.5 Common pitfalls of adopting SOA 64
 - 3.5.1 Building service-oriented architectures like traditional distributed architectures 65
 - 3.5.2 Not standardizing SOA 65
 - 3.5.3 Not creating a transition plan 66
 - 3.5.4 Not starting with an XML foundation architecture 67
 - 3.5.5 Not understanding SOA performance requirements 67
 - 3.5.6 Not understanding Web services security 68
 - 3.5.7 Not keeping in touch with product platforms and standards development 69

Chapter 4

- The Evolution of SOA 71
 - 4.1 An SOA timeline (from XML to Web services to SOA) 72
 - 4.1.1 XML: a brief history 72
 - 4.1.2 Web services: a brief history 73
 - 4.1.3 SOA: a brief history 74
 - 4.1.4 How SOA is re-shaping XML and Web services 76
 - 4.2 The continuing evolution of SOA (standards organizations and contributing vendors) 78
 - 4.2.1 “Standards” vs. “Specifications” vs. “Extensions” 78
 - 4.2.2 Standards organizations that contribute to SOA 79
 - 4.2.3 Major vendors that contribute to SOA 82
 - 4.3 The roots of SOA (comparing SOA to past architectures) 86
 - 4.3.1 What is architecture? 86
 - 4.3.2 SOA vs. client-server architecture 88
 - 4.3.3 SOA vs. distributed Internet architecture 95
 - 4.3.4 SOA vs. hybrid Web service architecture 104
 - 4.3.5 Service-orientation and object-orientation (Part I) 107

Chapter 5

Web Services and Primitive SOA	109
5.1 The Web services framework	111
5.2 Services (as Web services)	112
5.2.1 Service roles	114
5.2.2 Service models	126
5.3 Service descriptions (with WSDL)	131
5.3.1 Service endpoints and service descriptions	133
5.3.2 Abstract description	134
5.3.3 Concrete description	135
5.3.4 Metadata and service contracts	136
5.3.5 Semantic descriptions	137
5.3.6 Service description advertisement and discovery	138
5.4 Messaging (with SOAP)	142
5.4.1 Messages	143
5.4.2 Nodes	149
5.4.3 Message paths	152

Part II

SOA and WS-* Extensions	155
What is “WS-*”?	157

Chapter 6

Web Services and Contemporary SOA	
(Part I: Activity Management and Composition)	159
6.1 Message exchange patterns	162
6.1.1 Primitive MEPs	163
6.1.2 MEPs and SOAP	169
6.1.3 MEPs and WSDL	169
6.1.4 MEPs and SOA	171
6.2 Service activity	172
6.2.1 Primitive and complex service activities	174
6.2.2 Service activities and SOA	175

6.3	Coordination	177
6.3.1	Coordinator composition	179
6.3.2	Coordination types and coordination protocols	180
6.3.3	Coordination contexts and coordination participants	180
6.3.5	The activation and registration process	181
6.3.5	The completion process	182
6.3.6	Coordination and SOA	183
6.4	Atomic transactions	186
6.4.1	ACID transactions	187
6.4.2	Atomic transaction protocols	188
6.4.3	The atomic transaction coordinator	188
6.4.4	The atomic transaction process	189
6.4.5	Atomic transactions and SOA	191
6.5	Business activities	193
6.5.1	Business activity protocols	194
6.5.2	The business activity coordinator	195
6.5.3	Business activity states	195
6.5.4	Business activities and atomic transactions	196
6.5.5	Business activities and SOA	197
6.6	Orchestration	200
6.6.1	Business protocols and process definition	203
6.6.2	Process services and partner services	203
6.6.3	Basic activities and structured activities	204
6.6.4	Sequences, flows, and links	204
6.6.5	Orchestrations and activities	205
6.6.6	Orchestration and coordination	205
6.6.7	Orchestration and SOA	205
6.7	Choreography	208
6.7.1	Collaboration	209
6.7.2	Roles and participants	210
6.7.3	Relationships and channels	210
6.7.4	Interactions and work units	210
6.7.5	Reusability, composability, and modularity	210
6.7.6	Orchestrations and choreographies	211
6.7.7	Choreography and SOA	212

Chapter 7

Web Services and Contemporary SOA

(Part II: Advanced Messaging, Metadata, and Security)	217
7.1 Addressing	220
7.1.1 Endpoint references	222
7.1.2 Message information headers	223
7.1.3 Addressing and transport protocol independence	225
7.1.4 Addressing and SOA	225
7.2 Reliable messaging	228
7.2.1 RM Source, RM Destination, Application Source, and Application Destination	230
7.2.2 Sequences	230
7.2.3 Acknowledgements	231
7.2.4 Delivery assurances	233
7.2.5 Reliable messaging and addressing	235
7.2.6 Reliable messaging and SOA	235
7.3 Correlation	238
7.3.1 Correlation in abstract	239
7.3.2 Correlation in MEPs and activities	239
7.3.3 Correlation in coordination	240
7.3.4 Correlation in orchestration	240
7.3.5 Correlation in addressing	240
7.3.6 Correlation in reliable messaging	240
7.3.7 Correlation and SOA	241
7.4 Policies	242
7.4.1 The WS-Policy framework	243
7.4.2 Policy assertions and policy alternatives	244
7.4.3 Policy assertion types and policy vocabularies	245
7.4.4 Policy subjects and policy scopes	245
7.4.5 Policy expressions and policy attachments	245
7.4.6 What you really need to know	245
7.4.7 Policies in coordination	246
7.4.8 Policies in orchestration and choreography	246
7.4.9 Policies in reliable messaging	246
7.4.10 Policies and SOA	246

7.5	Metadata exchange	248
7.5.1	The WS-MetadataExchange specification	249
7.5.2	Get Metadata request and response messages	250
7.5.3	Get request and response messages	251
7.5.4	Selective retrieval of metadata	252
7.5.5	Metadata exchange and service description discovery	252
7.5.6	Metadata exchange and version control	253
7.5.7	Metadata exchange and SOA	254
7.6	Security	257
7.6.1	Identification, authentication, and authorization	259
7.6.2	Single sign-on	260
7.6.3	Confidentiality and integrity	261
7.6.4	Transport-level security and message-level security	262
7.6.5	Encryption and digital signatures	263
7.6.6	Security and SOA	265
7.7	Notification and eventing	266
7.7.1	Publish-and-subscribe in abstract	267
7.7.2	One concept, two specifications	268
7.7.3	The WS-Notification Framework	268
7.7.4	The WS-Eventing specification	271
7.7.5	WS-Notification and WS-Eventing	274
7.7.6	Notification, eventing, and SOA	274

Part III

SOA and Service-Orientation 277

Chapter 8

Principles of Service-Orientation 279

8.1	Service-orientation and the enterprise	280
8.2	Anatomy of a service-oriented architecture	284
8.2.1	Logical components of the Web services framework	284
8.2.2	Logical components of automation logic	285
8.2.3	Components of an SOA	288
8.2.4	How components in an SOA inter-relate	289

8.3	Common principles of service-orientation	290
8.3.1	Services are reusable	292
8.3.2	Services share a formal contract	295
8.3.3	Services are loosely coupled	297
8.3.4	Services abstract underlying logic	298
8.3.5	Services are composable	301
8.3.6	Services are autonomous	303
8.3.7	Services are stateless	307
8.3.8	Services are discoverable	309
8.4	How service-orientation principles inter-relate	311
8.4.1	Service reusability	312
8.4.2	Service contract	313
8.4.3	Service loose coupling	315
8.4.4	Service abstraction	316
8.4.5	Service composability	317
8.4.6	Service autonomy	318
8.4.7	Service statelessness	319
8.4.8	Service discoverability	320
8.5	Service-orientation and object-orientation (Part II)	321
8.6	Native Web service support for service-orientation principles	324

Chapter 9

Service Layers	327
9.1 Service-orientation and contemporary SOA	328
9.1.1 Mapping the origins and supporting sources of concrete SOA characteristics	329
9.1.2 Unsupported SOA characteristics	332
9.2 Service layer abstraction	333
9.2.1 Problems solved by layering services	334
9.3 Application service layer	337
9.4 Business service layer	341
9.5 Orchestration service layer	344
9.6 Agnostic services	346
9.7 Service layer configuration scenarios	347
9.7.1 Scenario #1: Hybrid application services only	348
9.7.2 Scenario #2: Hybrid and utility application services	349

9.7.3	Scenario #3: Task-centric business services and utility application services	349
9.7.4	Scenario #4: Task-centric business services, entity-centric business services, and utility application services	350
9.7.5	Scenario #5: Process services, hybrid application services, and utility application services	350
9.7.6	Scenario #6: Process services, task-centric business services, and utility application services	351
9.7.7	Scenario #7: Process services, task-centric business services, entity-centric business services, and utility application services	352
9.7.8	Scenario #8: Process services, entity-centric business services, and utility application services	352

Part IV

Building SOA (Planning and Analysis)	355
---	------------

Chapter 10

SOA Delivery Strategies	357
--------------------------------	------------

10.1	SOA delivery lifecycle phases	358
10.1.1	Basic phases of the SOA delivery lifecycle	358
10.1.2	Service-oriented analysis	359
10.1.3	Service-oriented design	359
10.1.4	Service development	360
10.1.5	Service testing	360
10.1.6	Service deployment	361
10.1.7	Service administration	361
10.1.8	SOA delivery strategies	362
10.2	The top-down strategy	363
10.2.1	Process	363
10.2.2	Pros and cons	365
10.3	The bottom-up strategy	366
10.3.1	Process	367
10.3.2	Pros and cons	368
10.4	The agile strategy	370
10.4.1	Process	370
10.4.2	Pros and cons	373

Chapter 11

Service-Oriented Analysis (Part I: Introduction)	375
11.1 Introduction to service-oriented analysis	377
11.1.1 Objectives of service-oriented analysis	377
11.1.2 The service-oriented analysis process	377
11.2 Benefits of a business-centric SOA	382
11.2.1 Business services build agility into business models	383
11.2.2 Business services prepare a process for orchestration	384
11.2.3 Business services enable reuse	384
11.2.4 Only business services can realize the service-oriented enterprise	385
11.3 Deriving business services	386
11.3.1 Sources from which business services can be derived	387
11.3.2 Types of derived business services	392
11.3.3 Business services and orchestration	395

Chapter 12

Service-Oriented Analysis (Part II: Service Modeling)	397
12.1 Service modeling (a step-by-step process)	398
12.1.1 “Services” versus “Service Candidates”	398
12.1.2 Process description	399
12.2 Service modeling guidelines	416
12.2.1 Take into account potential cross-process reusability of logic being encapsulated (task-centric business service candidates)	416
12.2.2 Consider potential intra-process reusability of logic being encapsulated (task-centric business service candidates)	417
12.2.3 Factor in process-related dependencies (task-centric business service candidates)	417
12.2.4 Model for cross-application reuse (application service candidates)	418
12.2.5 Speculate on further decomposition requirements	418
12.2.6 Identify logical units of work with explicit boundaries	419
12.2.7 Prevent logic boundary creep	419
12.2.8 Emulate process services when not using orchestration (task-centric business service candidates)	420

- 12.2.9 Target a balanced model 421
- 12.2.10 Classify service modeling logic 422
- 12.2.11 Allocate appropriate modeling resources 422
- 12.2.12 Create and publish business service modeling standards 422
- 12.3 Classifying service model logic 423
 - 12.3.1 The SOE model 424
 - 12.3.2 The enterprise business model 426
 - 12.3.3 “Building Blocks” versus “Service Models” 426
 - 12.3.4 Basic modeling building blocks 426
- 12.4 Contrasting service modeling approaches (an example) 430

Part V

- Building SOA (Technology and Design) 445

Chapter 13

- Service-Oriented Design (Part I: Introduction) 447

- 13.1 Introduction to service-oriented design 448
 - 13.1.1 Objectives of service-oriented design 448
 - 13.1.2 “Design standards” versus “Industry standards” 449
 - 13.1.3 The service-oriented design process 449
 - 13.1.4 Prerequisites 451
- 13.2 WSDL-related XML Schema language basics 453
 - 13.2.1 The schema element 454
 - 13.2.2 The element element 455
 - 13.2.3 The complexType and simpleType elements 455
 - 13.2.4 The import and include elements 456
 - 13.2.5 Other important elements 456
- 13.3 WSDL language basics 457
 - 13.3.1 The definitions element 458
 - 13.3.2 The types element 459
 - 13.3.3 The message and part elements 461
 - 13.3.4 The portType, interface, and operation elements 462
 - 13.3.5 The input and output elements (when used with operation) 462