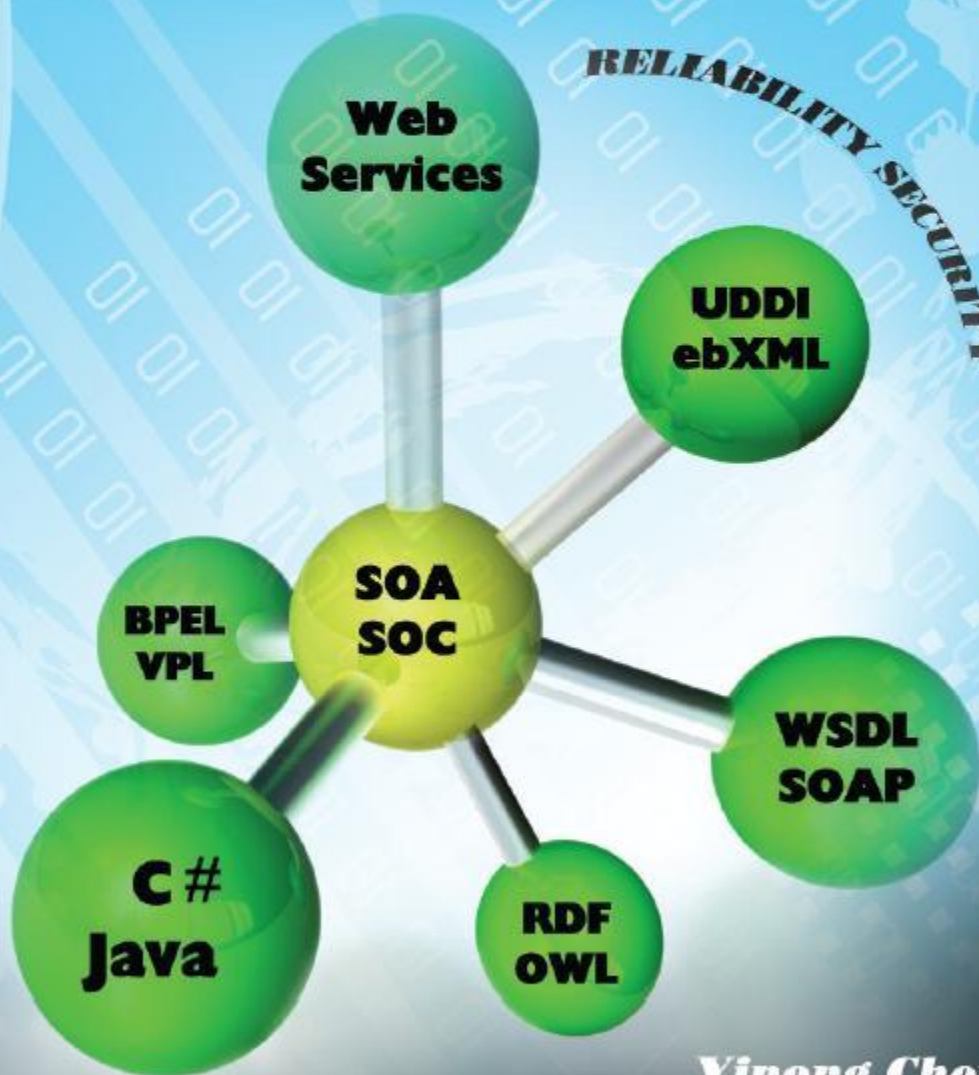


Distributed Service-Oriented **Software Development**



*Yinong Chen
Wei-Tek Tsai*

DISTRIBUTED SERVICE-ORIENTED SOFTWARE DEVELOPMENT

YINONG CHEN AND WEI-TEK TSAI

ARIZONA STATE UNIVERSITY



KENDALL/HUNT PUBLISHING COMPANY
4050 Westmark Drive Dubuque, Iowa 52002

Copyright © 2008 by Kendall/Hunt Publishing Company

ISBN 978-0-7575-5273-1

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the copyright owner.

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

Table of Contents

Preface	ix
Chapter 1	Introduction to Distributed Service-Oriented Computing	1
1.1	Computer Architecture and Computing Paradigms	1
1.1.1	Computer Architecture	1
1.1.2	Software Architecture	2
1.1.3	Computing Paradigms	3
1.2	Distributed Computing and Distributed Software Architecture	5
1.2.1	Distributed Computing	6
1.2.2	N-Tier Architecture	7
1.2.3	Distributed Object Architecture	9
1.3	Service-Oriented Architecture and Computing	11
1.3.1	Basic Concepts and Terminologies	11
1.3.2	Service-Oriented Computing	15
1.3.3	Object-Oriented Computing versus Service-Oriented Computing	17
1.3.4	Service-Oriented Enterprise	18
1.3.5	Service-Oriented System Engineering	20
1.4	Service-Oriented Software Development and Applications	22
1.4.1	Traditional Software Development Processes	22
1.4.2	Service-Oriented Software Development	22
1.4.3	Applications of Service-Oriented Computing	25
1.5	Discussions	27
1.6	Exercises and Projects	31
Chapter 2	Distributed Computing with Multithreading	39
2.1	Introduction to C# and .Net	39
2.1.1	Getting started with C# and .Net	40
2.1.2	Comparison between C++ and C#	42
2.1.3	Namespaces and the using Directive	44
2.1.4	The Queue Example in C#	46
2.1.5	Class and Object in C#	48
2.1.6	Parameters: Passing by Reference with ref & out	51
2.1.7	Base Class and Base Calling Class Constructor	52
2.1.8	Constructor, Destructor, and Garbage Collection	53
2.1.9	Pointers in C#	53
2.1.10	C# Unified Type System	54
2.2	Memory Management and Garbage Collection	56
2.2.1	Static Variables and Static Methods	57
2.2.2	Runtime Stack for Local Variables	57
2.2.3	Heap for Dynamic Memory Allocation	60
2.2.4	Scope and Garbage Collection	60
2.3	General Issues in Multitasking and Multithreading	61
2.3.1	Basic Requirements	61
2.3.2	Critical Operations and Synchronization	62

2.3.3	Deadlock and Deadlock Resolving	64
2.3.4	Order of Execution	66
2.3.5	Operating System Support for Multitasking and Multithreading	66
2.4	Multithreading in Java	68
2.4.1	Creating and Starting Threads	68
2.4.2	Thread Synchronization	72
2.4.3	Synchronized Method.....	73
2.4.4	Synchronized Statements	78
2.5	Multithreading in C#	79
2.5.1	Thread Classes and Properties.....	79
2.5.2	Monitor	80
2.5.3	Reader and Writer Locks.....	90
2.5.4	Mutexes	95
2.5.5	Semaphore	95
2.5.6	Coordination Event.....	96
2.6	Exceptions Handling and Event-Driven Programming	99
2.6.1	Exception Handling	99
2.6.2	Event-Driven Programming	104
2.7	Discussions	109
2.8	Exercises and Projects	111
Chapter 3	Getting Started with Service-Oriented Software Development	125
3.1	Overview of Service-Oriented Software Development Environments	125
3.2	Service Provider: Creating and Hosting Services	127
3.2.1	Using ASP .Net to Create Web Services.....	128
3.2.2	Program Your Services in C#.....	129
3.2.3	Testing Your Web Services.....	130
3.2.4	Hosting Your Web Services as a Service Provider	131
3.3	Service Brokers: Publishing and Discovering Services	133
3.3.1	An Ideal Service Broker with all Desired Features.....	134
3.3.2	UDDI Service Registry.....	137
3.3.3	ebXML Service Registry and Repository	145
3.3.4	Ad Hoc Registry Lists	148
3.4	Service Requesters: Building Applications Using Services.....	148
3.4.1	Creating a Web Application Project in ASP.Net	148
3.4.2	Creating GUI and Composing an Application Based on Remote Web Services.....	150
3.5	Java-Based Web Service Development.....	158
3.5.1	Web Application Building Using AJAX programming.....	158
3.5.2	Java-Based Web Service Development and Hosting	160
3.6	Discussions	162
3.7	Exercises and Projects	163
Chapter 4	XML and Related Technologies	169
4.1	XML	169
4.1.1	XML versus HTML.....	170
4.1.2	XML Syntax	171
4.1.3	XML Namespaces	174

4.2	XML Processing.....	175
4.2.1	DOM: Document Object Model.....	176
4.2.2	SAX: Simple API for XML.....	178
4.2.3	XML Processing in Java.....	180
4.3	XPath	182
4.4	XML Type Definition Languages	185
4.4.1	XML Document Type Definition (DTD).....	185
4.4.2	XML Schema.....	188
4.4.3	Namespace.....	190
4.4.4	Validation	192
4.5	Extensible Stylesheet Language.....	194
4.6	Discussions	199
4.7	Exercises and Projects	201
Chapter 5	Composition Languages for Service-Oriented Software Development.....	207
5.1	SOAP	208
5.1.1	SOAP Format	208
5.1.2	SOAP Over HTTP.....	210
5.1.3	Connecting Endpoint and Proxy	211
5.2	WSDL: Web Service Description Language.....	213
5.2.1	Elements of WSDL Documents	213
5.2.2	WSDL Document Example.....	214
5.3	BPEL	216
5.3.1	Overview of Composition Languages.....	216
5.3.2	BPEL Activities and Constructs.....	218
5.3.3	BPEL Process.....	218
5.3.4	WSDL Interface Definition of BPEL Process.....	221
5.3.5	BPEL Process	223
5.3.6	An Example Invoking Real Web Services	226
5.3.7	Stateless versus Stateful Web Services	233
5.3.8	BizTalk's Singleton Object Approach	233
5.3.9	BPEL's Correlation Approach	234
5.4	Frameworks Supporting BPEL Composition.....	237
5.4.1	Oracle SOA Suite	238
5.4.2	ActiveBPEL.....	238
5.4.3	BizTalk	240
5.5	WSFL: Web Services Flow Language.....	241
5.5.1	Overview of WSFL	241
5.5.2	Global Model.....	242
5.5.3	Flow Model	243
5.6	Service-Oriented Computing in Robotics Applications.....	245
5.6.1	Service-Oriented Robotics Applications.....	245
5.6.2	Event-Driven Robotics Applications.....	246
5.6.3	Developing Service-Oriented Applications in VPL.....	248
5.6.4	Developing Service-Oriented Robotics Applications	254
5.7	Other Composition Languages.....	260
5.7.1	OWL-S.....	260
5.7.2	SCA/SDO	261

5.7.2	Workflow Foundation	262
5.8	Discussions	264
5.8	Exercises and Projects	265
Chapter 6	Dependability of Service-Oriented Software	271
6.1	Basic Concepts	271
6.1.1	Dependability	271
6.1.2	Dependability Attributes and Quality of Service	273
6.1.3	Security Issues in SOA Software	273
6.2	Security Design in Web Applications	275
6.2.1	IIS and Windows-Based Security Mechanisms	275
6.2.2	Structure of Web Application and Security Management	277
6.2.3	Forms-Based Security	280
6.3	Windows Communication Foundation	286
6.3.1	A Comprehensive Service-Oriented Software Development Environment	286
6.3.2	WCF Service Endpoints	287
6.3.3	WS-Security	290
6.3.4	WS-Reliability	291
6.3.5	Transactions.....	293
6.4	Discussions	295
6.5	Exercises and Projects	297
Chapter 7	Database and Ontology in Distributed Service-Oriented Software.....	303
7.1	Databases in Service-Oriented Software	303
7.2	Relational Databases in Service-Oriented Software.....	305
7.2.1	Interface between Database and Software.....	305
7.2.2	SQL Database in ADO .Net	307
7.2.3	DataAdapter and DataSet in ADO .Net.....	311
7.3	XML-Based Database and Query Language XQuery	314
7.3.1	Expressing Queries	315
7.3.2	Transforming XML Document	317
7.3.3	XQuery Discussions	319
7.4	Ontology Languages RDF and RDF Schema.....	319
7.4.1	Semantic Web and Ontology.....	319
7.4.2	RDF	320
7.4.3	RDF Schema.....	322
7.4.4	Reasoning and Verification in Ontology.....	329
7.5	OWL: Web Ontology Language	331
7.5.1	From RDF to OWL	331
7.5.2	The OWL Class and Property	332
7.5.3	Boolean Combinations of Classes.....	333
7.5.4	Property Restrictions	333
7.5.5	Synopsis of OWL Lite, DL, and Full	334
7.7	Ontology Development Environments	336
7.8	Discussions	337
7.9	Exercises and Projects	339

Chapter 8	Service-Oriented Application Architecture	345
8.1	Introduction	345
8.2	Application Architectures	347
8.2.1	Dynamic Architecture via Dynamic Composition	349
8.2.2	Dynamic Re-Composition	350
8.2.3	Lifecycle Management Embedded in Operation Infrastructure	351
8.3	Examples of Service-Oriented Application Architectures	353
8.3.1	IBM WebSphere Architecture	353
8.3.2	Enterprise Service Bus	355
8.3.3	FERA Community Project	356
8.3.4	SAP NetWeaver	357
8.3.6	Service-Oriented Enterprise Model	358
8.3.7	User-Centric Service Oriented Architecture	361
8.4	Discussions	361
8.5	Exercises and Projects	363
Chapter 9	A Mini Walkthrough of Service-Oriented Software Development.....	369
9.1.	Introduction	369
9.2	Sample Domain Model	373
9.2.1	Ontology Systems.....	374
9.2.2	Published Services.....	378
9.2.3	Published Workflows	381
9.2.4	Shipping Domain Collaboration Templates	383
9.3	Specific Requirements for a Project	384
9.4	A Worked Example	387
9.5	Discussions	395
Appendix	Tutorials on Component-Based and Service-Oriented Software Development.....	399
A.1	Component-Based Movie and Game Programming.....	399
A.1.1	Developing a Game in an Engineering Process	400
A.1.2	Basic Programming Concepts in Alice	401
A.1.3	Graphic Programming	403
A.1.4	Online Tutorials and Examples	405
A.2	Web Application Composition	406
A.2.1	Design of Graphical User Interface.....	406
A.2.2	Discovering Web services Available Online	412
A.2.3	Access Web Services in Your Program: Cinema Service	414
A.2.4	Access Web services in Your Program: Weather Forecasting Service.....	419
A.2.5	Access Web Services in Your Program: USZip Service.....	422
A.3	Service-Oriented Robotics Applications	423
A.3.1	Getting Started with Microsoft Robotics Studio and VPL Programming	424
A.3.2	Programming Conditions in VPL.....	426
A.3.3	Programming Loop in VPL.....	427
A.3.4	Programming a Robot in a Simulation Environment	428
A.3.5	Deploying the Program to a Real Robot	434
A.3.6	Programming the Arm of the Robot.....	435

A.3.7	Autonomous Robot in an Obstacle Course	438
A.3.8	Autonomous Robot Exploring a Maze	443
A.4	Exercises and Projects	451
References	455
Index	467

Preface

Software development has evolved for several generations from imperative, procedural, object-oriented, to distributed object-oriented paradigms. As the emergence of service-oriented computing, distributed software development is shifting from *distributed object-oriented development*, represented by CORBA (Common Object Request Broker Architecture) developed by OMG (Object Management Group) and Distributed Component Object Model (DCOM) developed by Microsoft, to *distributed service-oriented development*. Service-oriented computing and service-oriented software development have been adopted and supported by all major computer companies, including BEA, Google, HP, IBM, Intel, Microsoft, Oracle, SAP, and Sun Microsystems, and their technologies have been standardized by OASIS, W3C, and ISO.

Before we start to introduce what this book is about, let us first clarify three fundamental concepts: service-oriented architecture, service-oriented computing, and service-oriented software development.

Service-Oriented Architecture (SOA) is a distributed software architecture, which considers a software system consisting of a collection of loosely coupled services that communicate with each other through standard interfaces and protocols. These services are platform independent. Services can be published in public or private directories or repositories for software developers to compose their applications. As a software architecture, SOA is a conceptual model that concerns the organization and interfacing among the software components (services). It does not concern the development of operational software.

Service-Oriented Computing (SOC) refers to the computing paradigm that is based on the SOA conceptual model. However, SOC goes a step further to include not only the concepts and principles, but also the methods, algorithms, coding, and evaluation, which are a large part of the software development process.

Service-Oriented Development (SOD) concerns the entire software development cycle based on SOA concepts and SOC paradigm, including requirement, specification, architecture design, composition, service discovery, service implementation, testing, evaluation, deployment, and maintenance. SOD also involves using the current technologies and tools to effectively produce operational software.

We use “Distributed Service-Oriented Software Development” as the title of the book to compare with the widely used “Distributed Object-Oriented Software Development” approach, and to emphasize the fact that service-oriented software development is distributed naturally. Not only is the software under development distributed in different computers in different

locations, but also the development process is distributed, in the sense that the application builders, service brokers, and service providers are developers working independently in different locations, but following the same interfaces and standards. Furthermore, we have a chapter (Chapter Two) to discuss distributed computing in general and how SOA, SOC, and SOD fit into the framework of general distributed computing.

Recently, many SOA, SOC, and SOD books have been published in response to the growing requirements in these areas. These books fall into one of the three categories:

- (1) high-level concepts and principles in SOA;
- (2) one of the aspects of the SOC, such as BPEL, Ontology, or XML;
- (3) SOD using a specific platform, such as Visual Studio .Net, Oracle SOA Suite, Java EE, or WebSphere. Most of these books are written by developers, and are largely focused on the language, platforms, and tools.

Different from the existing books, this book takes a balanced approach to teach all three topics of SOA, SOC, and SOD in one course, and covers a large portion of each topic in depth. The main concern of the book is to teach the SOA/SOC concepts, principles, and methods.

However, concepts, principles, and methods are not only explained in text and diagram, but also demonstrated in working code.
--

We believe that students can better understand concepts, principles, and methods, if they see a piece of working code that implements them.

We also introduce the cutting-edge technologies and tools that can be applied to develop operational software with reasonable size and functionality, such as an operational online bookstore, trading site, or a robotics program manipulating a real robot to traverse a maze with artificial intelligence. Such software can never be developed in a course assignment without the latest development tools and without using the services and components made available by professional service providers. Many exercises and at least one large project are given at the end of each chapter of the book for students to practice SOA and SOC concepts and to develop operational software.

This book covers SOA, SOC, and SOD topics in breadth and depth.
--

The book is based on the materials taught by the authors in CSE445/598 (Distributed Software Development) course in Computer Science and Engineering at Arizona State University every semester since Fall 2006. The CSE445 session is for seniors and the CSE598 session is for graduate students. The CSE598 also has an online session that is taught to students in the executive master's program in engineering. Many of these students are on the side of software project management. A part of advanced materials of the text was also taught in CSE 565 (Software Verification, Validation, and Testing). The objectives and outcomes of a course based on the text can include:

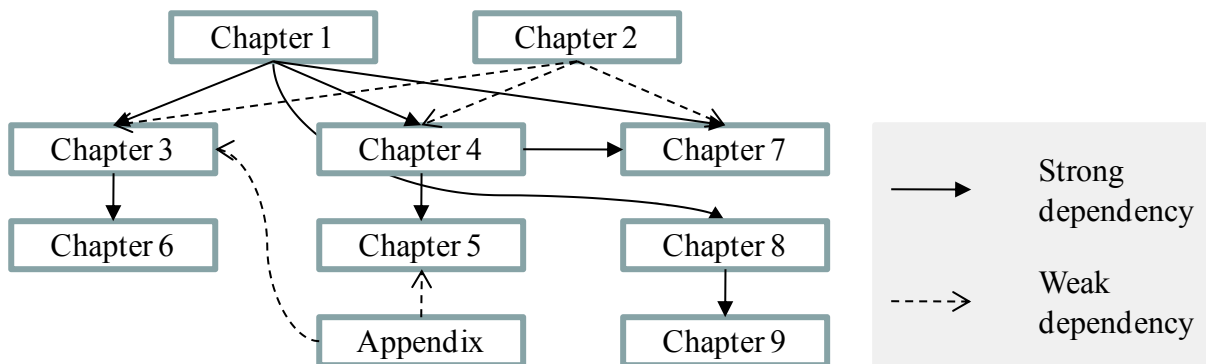
1. To develop an understanding of the software engineering of programs using concurrency and synchronization, with the following outcomes:
 - * Students can identify the application, advantages, and disadvantages of concurrency, threads, and synchronization.
 - * Students can apply design principles for concurrency and synchronization.
 - * Students can design and write programs demonstrating the use of concurrency, threads, and synchronization.
2. To develop an understanding of the development of distributed software, with the following outcomes:
 - * Students can recognize alternative distributed computing paradigms and technologies;
 - * Students can identify the phases and deliverables of the software lifecycle in the development of distributed software;
 - * Students can create the required deliverables in the development of distributed software in each phase of a software lifecycle;
 - * Students understand the security and reliability attributes of distributed applications.
3. To develop an ability to design and publish services as building blocks of service-oriented applications, with the following outcomes:
 - * Students understand the role of service publication and service directories;
 - * Students can identify available services in service registries;
 - * Students can design services in a programming language and publish services for the public to use.
4. To build skills in using a current technology for developing distributed systems and applications, with the following outcomes:
 - * Students can develop distributed programs using the current technology and standards;
 - * Students can use the current framework to develop programs and web applications using graphical user interfaces, remote services, and workflow.

This book is not for an introductory course in programming. Its main audiences are the seniors and graduate students in computer science and engineering, or software engineers with programming background. The readers are expected to be fluent in one of the object-oriented programming languages such as C++, C#, and Java. Furthermore, students are expected to have understood basic software engineering principles.

The book consists of nine chapters and an appendix. Each chapter is a unit that can be taught in six to nine lecture hours, depending on the level of the detail the instructor wants to cover. They are

- Chapter 1 Introduction to Distributed Service-Oriented Computing
- Chapter 2 Distributed Computing with Multithreading
- Chapter 3 Getting Started with Service-Oriented Software Development
- Chapter 4 XML and Related Technologies
- Chapter 5 Composition Languages for Service-Oriented Software Development
- Chapter 6 Dependability of Service-Oriented Software
- Chapter 7 Database and Ontology in Distributed Service-Oriented Software
- Chapter 8 Service-Oriented Application Architecture
- Chapter 9 A Mini Walkthrough of Service-Oriented Software Development
- Appendix Tutorials on Component-Based and Service-Oriented Software Development

The dependency among the chapters is illustrated in the diagram below. Based on the dependency, a subset of the chapters can be selected to satisfy a set of course requirements.



This book is not intended to be a research monograph, but an undergraduate text for teaching senior and graduate students on SOA, SOC, and SOD. However, research students and working professionals may still find this book useful, because of its comprehensive and in-depth discussions of the state-of-the-art contents, cutting-edge technologies, and professional development tools. The book is based not only on the teaching experiences of the authors in these areas, but also on the understanding and expertise that the authors have accumulated in their research in these areas.

As SOA, SOC, and SOD are new and dynamic, the technologies and tools are evolving rapidly. Some of the materials may need to be updated soon after the print of the book. It is our intention to cover the latest concepts and technologies, and we must cut in at some point in this process. We have put more emphasis on the SOA and SOC concepts, principles, and methods, which are relative stable compared to the SOD technologies and tools. We started to teach from the material of the book in Fall 2006. Large part of the development examples are initially based on .Net 2005. Now .Net 2008 is released. With little or no revision, we are able to test or convert all the examples into .Net 2008 before the print of the book. We expect the examples to work for the new editions of the tools in the future.

The tutorials in the appendix of the book are an important addition to the book. They provide full detail of Web application development discussed in Chapter Three and the robotics software development discussed in Chapter Five. On the other hand, the tutorials can be taught independently of the main text to students with no programming experience. In fact, the contents of the tutorials have been taught in a service-oriented computing course for high school students.

We like to thank many of our sponsors, supporters and colleagues in this project including Prof. Xiaoying Bai of Tsinghua University, Prof. Gary Bitter of Arizona State University, Prof. Farokh Bastani of University of Texas at Dallas, Prof. Kuo-Ming Chao of Coventry University, Dr. Shuyuan Chen of SAP, Dr. J. Y. Chung of IBM, Prof. Zhihui Du of Tsinghua University, Dr. K. W. Hwang of IBM, Prof. Kane Kim of University of California at Irvine, Prof. Y. H. Lee of Arizona State University, Prof. Yisheng Li of Fudan University, Prof. K. J. Lin of University of California at Irvine, and Dr. Raymond Paul of DoD OSD NII, Dr. Mary White of Arizona State University, Prof. S. S. Yau, Arizona State University, Prof. I-Ling Yen of University of Texas at Dallas. They contributed to our understanding of the materials. We also acknowledge the generous support from U.S. Department of Education and U.S. Department of Defense. Without their support, this book will not be possible. We also thank the teaching assistants and research assistants at Arizona State University involving Zhibin Cao, Calvin Cheng, Sandy Chow, Jay Elston, Qian Huang, Sheng Liu, Zheng Liu, Wu Li, Xin Sun, Jingjing Xu, Xinyu Zhou, and Peide Zhong. They validated many of the examples and assignments used in the book. Finally, we would like to thank our families for their support and understanding of taking on such a project while carrying out a full research and teaching load at the university.

Note for Instructors

All the homework assignments have been classroom-tested at Arizona State University. Furthermore, all the code presented in this book has been developed and tested. Contact the authors if you are interested in obtaining more materials in this book. This book also has a corresponding website at <http://asusrl.eas.asu.edu/share/services/book/> where you can download resources related to this book. Instructor-only resources can be obtained by directly contacting the authors at {yinong, wtsai}@asu.edu.

Yinong Chen

Wei-Tek Tsai

May 2008