NINTH EDITION

SERVICE-ORIENTED COMPUTING AND SYSTEM INTEGRATION

SOFTWARE, IOT, BIG DATA, AND AI AS SERVICES

YINONG CHEN AND GENNARO DE LUCA

ARIZONA STATE UNIVERSITY

Kendall Hunt
publishing company

Table of Contents

Preface

Service-Oriented Computing (SOC), web software development, cloud computing, big data processing, and artificial intelligence represent the modern software engineering theories, practices, and technologies, which have reshaped the world in all aspects. The amount of data is not the key. The relationship among all types of data and the meaning behind the data are the key. Efficiently finding the connections of all related data and using these connections to make intelligent decisions become possible after the maturity of these cutting-edge theories, practices, and technologies. The goals of the book are to introduce and exercise these cutting-edge theories, practices, and technologies through lectures and assignments based on the lectures.

The text takes a comprehensive and coherent approach to studying the latest service-oriented architecture, distributed computing paradigm, distributed software development, and system integration technologies. The objectives are to learn the concepts, principles, methods, development frameworks, and their applications. These learning objectives are supported by hands-on examples. In the service development part, we assume that students have good knowledge in object-oriented computing, such as C++, C#, Java, or Python. Students learn to build services through class definition, interface specification, the association between class methods and service operations, service deployment, and service hosting. In the system integration part, we assume that students have a basic understanding of software architecture through a general software engineering course. We take an architecture-driven approach to help students create the working solution step-by-step from their architecture design. The first step is to design the architecture, which includes the major components and the interconnection. The next step is to define the interfaces among the components using the standard data types. Finally, the behavior of each component is linked to remote services or local objects. The elaborated architecture is automatically translated into the executable.

The text consists of 13 chapters and 3 appendices. They are organized into three parts. Each part can be taught as a separate course, even though they are intrinsically related to the central goals and objectives of the book.

Part I: Distributed Service-Oriented Software Development and Web Data Management

Chapter 1	Introduction to Distributed Service-Oriented Computing
Chapter 2	Distributed Computing with Multithreading
Chapter 3	Essentials in Service-Oriented Software Development
Chapter 4	XML and Web Data Formats
Chapter 5	Web Application and State Management
Chapter 6	Dependability of Service-Oriented Software

Part II: Advanced Service-Oriented Computing and System Integration

Chapter 7	Advanced Services and Architecture-Driven Application Development

Chapter 8	Enterprise Software Development and Integration
Chapter 9	IoT, Robotics, and Device Integration via Visual Programming
Chapter 10	Interfacing Service-Oriented Software with Databases
Chapter 11	Big Data Processing and Cloud Computing
Chapter 12	Artificial Intelligence and Machine Learning
Chapter 13	Mobile Computing and Application Development

Part III: Appendices: Tutorials on Service-Oriented System Development

Appendix A	Web Application Development
Appendix B	Visual IoT/Robotics Programming Language Environment
Appendix C	ASU Repository of Services and Applications

Part I includes the first six chapters, which can be used for a distributed computing, service-oriented computing, or web software development course at the junior, senior, or graduate level of universities. This part emphasizes the computing paradigm, data representation, state management, and programming language-based SOC software development. It introduces fundamental concepts and principles, in addition to technologies and tools, which are not being taught in traditional software engineering courses.

Chapter 1 gives an overview and explains fundamental concepts of distributed software architecture, design patterns, distributed computing, service-oriented architecture, and enterprise software architecture. The connections and distinctions between object orientation and service orientation are discussed in detail.

Chapter 2 studies parallel computing in multithreading. It discusses threading, synchronization, coordination, event-driven programming, and performance of parallel computing under multicore computers.

Chapter 3 introduces the essential concepts and techniques in service-oriented architecture, particularly the three-party model of service-oriented software development: Service providers, service brokers, and service consumers. Service interfaces, service communication protocols, and service hosting are keys for supporting this new computing paradigm.

Chapter 4 discusses XML and related technologies, which are used almost everywhere in service-oriented software development. XML is used for representing data, interface, standards, protocols, and even the execution process definition. This chapter studies not only XML representations, but also XML processing and transforming.

Chapter 5 is a longer chapter and comprises application logic, data and state management, and presentation design. It involves application building based on architecture design using existing services and components, stateful web application development using different state management techniques, including view state, session state, application state, file management, and web caching. At the presentation layer, it discusses dynamic graphics generation, animation, and phone app development.

Chapter 6 deals with the dependability issues in web-based applications, including access control through Forms security, encryption and decryption applications, and Secure Sockets Layer in web communication. The chapter also discusses the reliability issues in web application design and particularly in web communication.

Part II includes the next seven chapters. These chapters are built on the basic concepts and principles discussed in Part I, yet they do not rely on the details of the first six chapters. This part emphasizes software and system composition and integration using existing services and components. It is based on

an architecture-driven approach, workflow, higher-level data management, and message-based integration. The material is good for an advanced software engineering, software integration, or system integration course at the senior or graduate level of universities.

Chapter 7 starts with reviewing service-oriented computing and service development covered in Part I. Then the chapter moves on to discuss more advanced service development that supports self-hosting and asynchronous communications. It also presents more detail in RESTful service development that has been briefly discussed in Part I, as well microservices. Finally, the chapter moves on to advanced web application development in HTML5, MVC, and .Net Core architecture development.

Chapter 8 starts with workflow-based software development and Workflow Foundation that supports architecture-driven software development. It uses examples and case studies to demonstrate software development by drawing the flowchart consisting of blocks of services and local components, adding inputs/outputs to the blocks, and then compiling the flowchart into executables. The chapter further discusses flowchart-based and architecture-driven software development by using other process languages and development environments. It first discusses BPEL (Business Process Execution Language) and BPEL-based development environments. Then the discussion is extended into message-based software integration and Enterprise Service Bus tools for integrating web contents.

Chapter 9 extends web-based computing to Internet of Things (IoT) and Robot as a Service (RaaS). As an example, robotics applications are studied in detail, using the service-oriented Visual IoT/Robotics Programming Language Environment (VIPLE) developed at Arizona State University. Full programming examples in VIPLE and hardware platform supported are discussed. Simulation environments are offered for teaching the chapter without using hardware. The simulators include garage door controller, maze navigation, traffic simulator, and autonomous driving environment.

Chapter 10 covers service-oriented database management, which focuses on the interface between service-oriented software and relational databases, XML databases, and LINQ (Language Integrated Query), and using LINQ to access object, relational, and XML databases.

Chapter 11 studies the cutting-edge topics in big data and cloud computing. It discusses major issues in big data, including big data infrastructure, big data management, big data analytics, and big data applications. Hadoop and VIPLE are used for illustrating automated data splitting and parallel computing. The relationship between big data and cloud computing is discussed. Finally, the chapter presents cloud computing and its main layers: Software as a Service, Platform as a Service, and Infrastructure as a Service. As examples, cloud platforms from Amazon Web Services, Google, IBM, Microsoft, and Oracle are discussed.

Chapter 12 presents the latest artificial intelligence, machine learning, and ontology theories and technologies. The latest generation of artificial intelligence is based on big data analysis and processing. This chapter presents its development, main concepts, and examples of developing machine-learning programs. Ontology is presented as a part of knowledge representation for big data processing and artificial intelligence applications.

Chapter 13 builds on the skills discussed in the previous chapters to introduce mobile computing and application development. It builds applications based on the MVC architecture and the advanced service development discussed in Chapter 7. It also links mobile app development to the Visual IoT/Robotics Programming Language Environment (VIPLE) introduced in Chapter 9. Phone simulators and physical phones (Android and iPhone) can be used to host the mobile apps developed in this chapter.

Part III consists of three appendices that supplement and support the main contents on web application development and IoT/robotics application development.

Appendices A and B contain tutorial-based materials that provide stepwise instructions, without missing pieces, to build working applications from scratch. These tutorials and exercises can help students to learn

concepts by examples. This part, in conjunction with parts of Chapter 3 and Chapter 9, can also be used for a freshman level course to introduce computing concepts through basic web application development and robotics programming.

Appendix C is the entrance to ASU Repository of Services and Applications. It lists and explains some of the deployed examples and URLs of SOAP services, RESTful services, web applications, and other resources used in this text. Free services found on the Internet come and go without any guarantee on quality of service. The repository provides a stable resource for teaching from this book without worrying about the availability and performance of the free services found on the Internet. ASU Repository of Services and Applications is open to the public and can be accessed at:

http://venus.sod.asu.edu/WSRepository/

Updates are carried out throughout the book in this new edition. A few major changes are carried out in the following parts.

Chapter 2 is updated to include a new section on asynchronous function call and performance evaluation between synchronous and asynchronous communications.

Chapter 3 is updated to shift the emphasis from WSDL services to RESTful services. A large part of the Chapter is rewritten to reflect the change of the current industrial practices.

Chapter 9 is also significantly updated to cover the new contents developed in the past two years. Autonomous driving in TORCS is updated. Traffic simulation and its programming are included in the book for the first time.

Chapter 13 was first included in the eighth edition by Dr. Gennaro De Luca. Major changes and updates are implemented in this edition.

Appendix A is updated to include a lis new services available.

Another important change in this edition is the further improvement of the ASU Repository of Services and Applications after ASU server was is transferred to an Amazon AWS cloud server. As a result, all the Web service and application URL addresses have been changed from the previous address "http://neptune.fulton.ad.asu.edu/WSRepository/" to the new https://venus.sod.asu.edu/WSRepository/ address. The new server should work in the same way as the previous server, and we expect the server to be more secure and to perform better.

The book embraces extensive contents. It can be used in multiple courses. At Arizona State University, we use the book as the text for two major required courses. The first course is CSE445/598 (Distributed Software Development), where the CSE445 session is for juniors and seniors, while the CSE598 session is for graduate students. The course started in Fall 2006, and first edition of the book was developed for this course in 2008. The first six chapters in Part I of this text are used for this course.

A second course CSE446/598 (Software Integration and Engineering) was piloted in 2010 and 2011, and the course became a regular course in 2012. The seven chapters in Part II of this text are used for this course.

Both CSE445 and CSE446 are required courses of the Software Engineering Concentration in the Computer Science program at Arizona State University.

The following table illustrates the lectures of CSE445/598 and CSE446/598. Each lecture is 75-min long and counts as 1.5 lecture hours. Each course is completed in about 44 lecture hours.

The first course focuses on distributed software development, including multithreading programming, event-driven programming, Web data representation, service development, and application building using programming languages as the composition language. Both C# and Java are used in the development. The course objectives and outcomes of ASU CSE445/598 are as follows:

1. To develop an understanding of the software engineering of programs using concurrency and synchronization.
    - The student can identify the application, advantages and disadvantages of concurrency, threads, and synchronization.
    - The student can apply design principles for concurrency and synchronization.
    - The student can design and write programs demonstrating the use of concurrency, threads, and synchronization.
2. To develop an understanding of the development of distributed software.
    - The student can recognize alternative distributed computing paradigms and technologies.
    - The student can identify the phases and deliverables of the software life cycle in the development of distributed software.
    - The student can create the required deliverables in the development of distributed software in each phase of a software life cycle.
    - The student understands 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.
    - The student understands the role of service publication and service directories
    - The student can identify available services in service registries.
    - The student can design services in a programming language and publish services for the public to use.
4. To build skills in using current technology for developing distributed systems and applications.
    - The student can develop distributed programs using the current technology and standards.
    - The student can use the current framework to develop programs and web applications using graphical user interfaces, remote services, and workflow.

The second course focuses on software and system integration using workflow languages and cutting-edge topics in software and system development. The course objectives and outcomes of ASU CSE446/598 are as follows:

1. To understand software architecture and software process.
    - Students understand the requirement and specification process in problem solving.
    - Students understand software life cycle and process management.
    - Students can identify advantages and disadvantages of software architectures and their trade-offs in different applications.
2. To understand and apply composition approach in software development.
    - Students can apply software architecture to guide software development in the problem-solving process.
    - Students understand interface requirements of software services.
    - Students can compose software based on interfaces of services and components.
    - Students can develop software systems using different composition methods and tools.
3. To understand and apply data and information integration in software development.
    - Students can compose software systems using different data resources in different data formats.
    - Students can integrate application logic with different databases.
    - Students can apply the entire software life cycle to develop working software systems.

CSE445/598 Lecture by Lecture Contents	CSE446/598 Lecture by Lecture Contents
L01 - 1. Intro to Architecture L02 - 1. Intro to Distributed Architecture L03 - 1. Intro to SOC Concepts L04 - 1. Intro to SOC Development L05 - 2. Multithreading concepts L06 - 2. Multithreading in Java and Python L07 - 2. Multithreading in C Sharp L08 - 2. Event-driven programming L09 - 2. Threading and Async Performance L10 - 3. SOC Service Development L11 - 3. SOC Hosting and Brokerage L12 - 3. SOC App Development in C# L13 - 3. SOC Service and App in Java L14 - 4. XML basics L15 - 4. XML processing L16 - 4. XML Types - Schema - HTML5 L17 - 4. XML Transformation L18 - 4. Other Web Data and Standards L19 - 5. Web Computing Models L20 - 5. Web App Architecture and Controls L21 - 5. Config-Global-DLL-Cookies L22 - 5. Session and File System L23 - 6. Security-Reliability Concepts L24 - 6. Forms Security L25 - 6. Data Encryption-Hash-Reliable Msg L26 - 6. Error Control and SSL L27 - 5. Dynamic Graphics - Animation L28 - 5. MVC Summary and Outlook	Unit 1-1 Introduction Unit 1-2 Self Hosting Service Unit 1-3 Advanced WCF Service Unit 1-4 RESTful Concepts Unit 1-5 RESTful Services Unit 2-1 Advanced Web App Architecture Unit 2-2 Enterprise Architecture and Process Unit 2-3 Workflow 1 Concepts Unit 2-4 Workflow 2 Case Studies Unit 3-1 BPEL 1 Process Unit 3-2 BPEL 2 Case Study and Frameworks Unit 3-3 Message-Based Integration Unit 3-4 WebCaching-Recommend Unit 4-1 Device Integration Unit 4-2 VIPLE Programming Unit 4-3 VIPLE FSM and Maze Unit 4-4 Mobile Computing and Phone App Unit 5-1 ADO Unit 5-2 LINQ to Object Unit 5-3 Lambda and LINQ to SQL Unit 5-4 LINQ to XML Unit 5-5 XML Database Unit 6-1 Big Data Concepts and Domains Unit 6-2 Big Data Processing Unit 6-3 AI and Machine Learning Unit 6-4 Ontology Unit 7-1 Cloud Computing Unit 7-2 Cloud Computing Case Studies

We recommend teaching the two courses in a sequence. However, the two courses can be taught independently without making one to be the prerequisite of the other. In this case, the basic concepts and principles from Part I, including those from a part of Chapter 1 and the first section of Chapter 4, should be reviewed or be assigned as reading materials for preparing the course using Part II. It is also sensible to choose a few topics from Part I and Part II to teach one course from the book. For example, Chapters 1, 3, 5, and 8 can form a good service-oriented computing course. Chapter 9 and Appendix B can be used for a computational thinking-based robotics course for students who do not have much programming language background.

Dr. De Luca has been using Chapters 2, 3, 4, 9, and 13 to teach the course IFT370 (Mobile Computing & Apps for IT) in the Information Technology department. The course covers the fundamental concepts for the development of cross-platform mobile applications. It covers programming paradigms (including object-oriented, functional, and service-oriented), program performance, design patterns, web services

(including REST, SOAP, and RPC), mobile software architectures， including three-layer, MVC, and MVVM, data storage and access, multithreading, and the basics of .NET MAUI in XAML and C#.

The authors would like to thank our colleagues at Arizona State University in preparing this book and related courses. Prof. Wei-Tek Tsai has taught CSE445/598. He was a coauthor of the book's first five editions. Dr. Janaka Balasooriya has been teaching CSE445/598 regularly and has constantly provided feedback and suggestions for improving the contents of the course and the book. Prof. Yann-Hang Lee contributed to the course contents related to IoT and robotics development. Prof. Hessam Sarjoughian and Prof. Stephen Yau contributed to the development of CSE445/598 and CSE446/598 courses. Many of our sponsors, colleagues, cooperators, and students have been involved in this project, including Prof. Farokh Bastani of University of Texas at Dallas, Dr. J. Y. Chung of IBM, Prof. Mei Hong of Beijing Institute of Technology, Prof. Yingxu Lai of Beijing University of Technology, Prof. Yinsheng Li of Fudan University, Prof. Zhongtao Li of University of Jinan, Prof. K. J. Lin of University of California at Irvine, Mr. John Oliver of Intel, Profs. Ruzhi Xu and Zhizheng Zhou of Qilu University of Technology, and 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 Intel, the US Department of Education, and the National Science Foundation. Without their support, the development of the contents of the book would not be possible. We also thank the teaching assistants and research assistants at Arizona State University, including Calvin Cheng, Jay Elston, Qian Huang, Adam Lew, Wu Li, Gavin Liu, Mengxue Liu, Sami Mian, Xin Sun, Jingjing Xu, Xinyu Zhou, Thomas Zelpha, and Peide Zhong. Finally, we would like to thank our families for their support and understanding in taking on such a project while carrying out a full research and teaching load at the university.

Note for Instructors

All the assignments and projects have been classroom-tested at Arizona State University for many years. 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. Instructor-only resources, such as complete presentation slides, assignments, and tests, can be obtained by directly contacting the authors at Yinong.Chen@asu.edu and Gennaro.Deluca@asu.edu.

Yinong Chen

Gennaro De Luca

December 2023

Part I
Distributed Service-Oriented Software Development and Web Data Management