Service-Orientation in Computing Curriculum

Yinong Chen and W. T. Tsai
School of Computing, Informatics, and Decision Systems Engineering
Arizona State University, Tempe, AZ 85287-8809, U.S.A.

***Abstract* — Web software development and cloud computing based on Service-Oriented Architecture (SOA) and Service-Oriented Computing (SOC) represent the modern software engineering theories, practices, and technologies. As an architecture-driven computing paradigm, SOA and SOC are mature and are becoming the major paradigm for software development. SOA and SOC should be taught in all computer science and computer engineering programs. We do not suggest using SOC to replace the currently taught Object-Oriented Computing (OOC) paradigm. As SOC is based on OOC, we suggest to teaching SOC as the continuation and extension of OOC. At Arizona State University, SOA and SOC paradigm is incorporated into our Computing Science and Software Engineering programs since 2006. This paper presents the topics of the related courses and the open resources created for the courses, which are available for public accesses, including textbooks, lecture presentation slides, tests and assignments, software tools, a repository of sample services and applications, and tutorials of using tools and the cloud computing environment for service hosting and deployment.**

Keywords – Computing paradigm, service-oriented architecture, service repository, computing curriculum, computer science education

I. INTRODUCTION

Software development has evolved for several generations from imperative, procedural, object-oriented, to distributed object-oriented computing paradigms [1][2]. As the emergence of Service-Oriented Architecture (SOA) and Service-Oriented Computing (SOC) [3][4][5], software development is shifting from distributed object-oriented development, represented by for example, CORBA (Common Object Request Broker Architecture) [5] developed by OMG (Object Management Group) and Distributed Component Object Model (DCOM) [7] developed by Microsoft, to service-oriented development (SOD). SOA, SOC, and SOD have been adopted and supported by all major computer companies. The related technologies have been standardized by OASIS, W3C, and ISO [8]. Government agencies also adopted SOC in their system development [9].

Before proceed further, let us first clarify the fundamental concepts used in the paper and in other literatures: SOA, SOC, SOD, and SaaS (Software-as-a-service) [3][4][5][8].

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 standard protocols [3][4]. 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.

SOC refers to the computing based on the SOA conceptual model [5]. However, SOC goes a step further to include not only the concepts and principles, but also the methods, algorithms, coding, and evaluation, which form a large part of the software development process.

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 [8]. SOD also involves using the current technologies and tools to effectively produce operational software.

SaaS is a special kind of software that runs on the top of a cloud platform. The simplest SaaS is simply a Web service. However, SaaS often refers to a Web application that uses Multi-Tenancy Architecture (MTA) with a scalable scheduler. SaaS takes advantages of the computing resources provided by the cloud platform, such as platform-as-a-service (PaaS) and infrastructure-as-a-service (IaaS). Well-known PaaS include GAE (Google App Engine), EC2, and Azure. Similarly, Salesforce.com is a well-known SaaS with MTA. A MTA SaaS uses one code base but can serve hundreds of thousand tenants, and each tenant has it own users. Thus, a MTA SaaS with a single code

978-1-4673-0412-2/11/$26.00 ©2011 IEEE

base can serve millions of users with many different customizations by the tenants.

In OOD, the software is often developed on desktop computers and distributed through executable files, the deployment is not a major issue. In SOD, on the other hand, software can be developed on desktop and then deployed to a server, or directly developed and hosted on a cloud computing environment. Hosting and deployment are major issues in the development process, which makes the teaching topics on SOD important, in additional to the topics on SOA and SOC.

In Computer Science and Computer System Engineering programs at the Arizona State University, particularly in the Software Engineering concentration, SOA, SOC, and SOD are taught in a number of courses, including freshman course CSE101 (Introduction to Computer Science and Engineering), sophomore course CSE240 (Introduction to Programming Languages), senior course CSE445 (Distributed Software Development), and CSE494 (Software Integration and Engineering). CSE240 teaches different programming paradigms and uses a programming language in each paradigm to write application programs in the paradigm. SOC is introduced as the latest computing paradigm [2]. This paper will focus on CSE101, CSE445, and CSE494, where service orientation is the key topics of these courses. While teaching these courses, we collected and created abundance of resources, which are made available to all instructors and students around the world, including textbooks [2][8], syllabi [10][16][17], presentation slides, tests and assignments, a repository of sample services and applications for public accesses [21]. The courses also suggested the software tools and the cloud computing environment for service hosting and deployment and created tutorials of using the tools environments.

This paper is organized as follows. Section II presents the service-oriented robotics application development in the freshman course CSE101. Sections III and IV discuss the core topics of SOA, SOC, and SOD taught in CSE445 and CSE494, respectively. Section V presents the repository of resources created for the courses. Section VI outlines the textbook developed for the courses discussed in this paper. Section VII shows the enrollments history of the courses since offered. Section VIII concludes this paper.

II. SERVICE-ORIENTED ROBOTICS PROGRAMMING

CSE101 is a course required for all students in Computer Science program and Computer System Engineering program at Arizona State University. The course consists of one lecture hour and three lab hours each week for 15 weeks. The main tasks in the three lab hours are using service-oriented robotics development to learn the engineering design process. The development environment used is Microsoft Robotics Developer Studio (MRDS) and its Visual Programming Language (VPL) [11]. First released in 2006, MRDS and VPL laid an important milestone in service-oriented computing. VPL is architecture-driven and is a service-oriented programming language that allows students to develop services, deploy the services into a repository, and then use the services in the repository to develop workflow-based robotics applications.

As shown in Figure 1, MRDS is based on the .Net framework. VPL, C#, and Visual Basic can be used to program services as well as applications. The services can be added into MRDS's service repository. The Concurrency and Coordination Runtime (CCR) shown in Figure 1 (right) provides a concurrent message-oriented programming model for SOA application development, and it manages asynchronous operations, dealing with concurrency, exploiting parallel hardware and handling exceptions and failures. As a result, users do not need to deal with threading, locks, and semaphores.

Decentralized System Services (DSS) provides a service-hosting environment and a set of basic services facilitating tasks such as debugging, logging, monitoring, security, discovery, and data persistence. It also manages the interoperability between simulation engine and CCR, and allows application compositing using remote services. The DSS runtime is built on top of CCR and standard communication protocols. Instead of using SOAP over HTTP, DSS runtime uses REST (Representational State Transfer) over HTTP, which can better deal with the real-time requirement in the robotics applications.

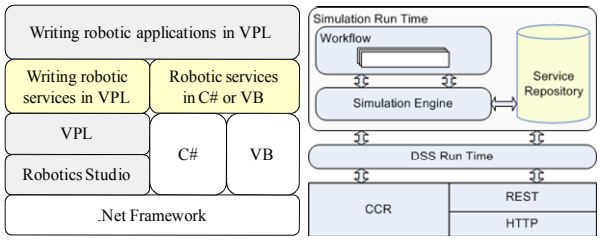

Figure 1. MRDS and its development environment

Selecting a good development environment is important in a freshman class. Designing laboratories and a good design project is even more critical. We piloted the first offer of the service-oriented robotics development course in Fall 2006, two months after Microsoft released the Robotics Studio and VPL in June 2006, thanks to our prior research on service-oriented robotics application sponsored by Microsoft Embedded Systems Research Grant starting in 2003. The course evolved to the form of today after five years' offerings. The course consists of one lecture hour and three laboratory hours. Student takes the lecture in the

morning, read the lab manual and take a pre-lab quiz (online) before attending the lab session in the afternoon. A number of resources have been developed to support the course, including syllabus [10], the lab manual [11], videos of sample robotics projects [10], a repository of services and sample applications [21]. Figure 2 shows a simulated robot in a maze developed for the design project. Student must design an autonomous maze navigation algorithm, such as a distance-based greedy algorithm or a wall-following algorithm, and write VPL program to traverse the maze. The program must be written for the simulated robot and maze first. Then, the student must reconfigure the program to control the real NXT robot to navigate a real maze.

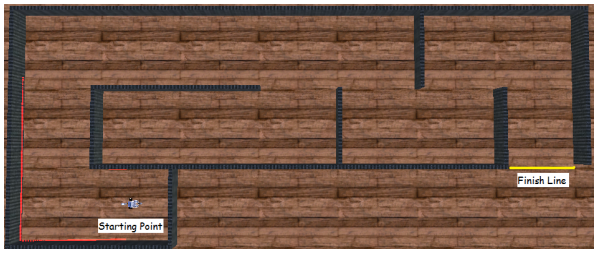

Figure 2. A simulated maze for supporting lab exercise

The service-oriented robotics development in CSE101 includes the following laboratories:

- Using basic activities;
- Developing activities and services representing logic gates;
- Using the gates (activities and services) to develop an adder and an ALU (Arithmetic and Logic Unit);
- Event-driven programming of sensors;
- Construction of effectors and programming of actuators for ball capture (treasure hunting);
- Programming of a sumo robots;
- Programming finite state machines, programming stateless and stateful vending machines;
- Programming maze navigation robot based on finite state machine description of different algorithms.

In the formal laboratories, students learn VPL language and using VPL to program different components and applications. A design project is given as a homework outside the lab sessions. Students will work in a group of three or four to solve a number challenges, including autonomous maze navigation, autonomous sumo robots, and a remote-control ball game and treasure hunting. Based on the engineering design process learned in the course lectures and the experience in the labs and design project, students will give a presentation and write a final report of their design project.

As the materials are easy to learn, exciting, and educational, we have proposed to teach the service-oriented robotics development as a part of the high school computing course. The proposal was funded by the U.S. Department of Education's FIPSE program (2007 to 2010) [12], which leads the implementation of the course in Coronado High School [13][14] and the summer Robotics Camps for high school students [15].

III. DISTRIBUTED SOFTWARE DEVELOPMENT

In the Computer Science (CS) program at Arizona State University, there is a Software Engineering (SE) concentration. Students enrolled in SE concentration are required to take the following four courses, in addition to the required CS courses:

- Software Analysis and Design
- Software Quality Assurance and Testing
- CSE445: Distributed Software Development
- CSE494: Software Integration and Engineering

CSE445 and CSE494 form a sequence and are designed to teach the latest software engineering techniques in service-oriented computing, as well as the latest technologies supporting the development of service-oriented software. The topics of CSE445 will be discussed in this section and the topics of CSE494 will be presented in the next section.

CSE445 is the first class in our computer science program to discuss distributed software development in depth. The objectives and outcomes of a course include [16]:

1. To develop an understanding of the software engineering of programs using concurrency and synchronization, with the following outcomes:
   a. Students can identify the application, advantages, and disadvantages of concurrency, threads, and synchronization.
   b. Students can apply design principles for concurrency and synchronization.
   c. 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:
   a. Students can recognize alternative distributed computing paradigms and technologies;
   b. Students can identify the phases and deliverables of the software lifecycle in the development of distributed software;
   c. Students can create the required deliverables in the development of distributed software in each phase of a software lifecycle;
   d. 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:

a. Students understand the role of service publication and service directories;
b. Students can identify available services in service registries;
c. 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:
   a. Students can develop distributed programs using the current technology and standards;
   b. Students can use the current framework to develop programs and Web applications using graphical user interfaces, remote services, and workflow.

CSE445 is designed to achieve the objectives and outcomes of the course. The course contents are organized in six units:

1. Introduction to Distributed Computing and Service-Oriented Computing: This unit gives an overview of the area and the main topics to be discussed in the course. It covers distributed computing paradigm, distributed software architecture, design patterns, concepts of service-oriented architecture, service-oriented computing, and service-oriented software development.
2. Distributed Computing with Multithreading: This unit covers the fundamental issues in distributed computing, including critical operations, synchronization, resource locking versus unbreakable operations, semaphore, events and event coordination, and event-driven distributed computing in Web environment. As a part of our parallel computing initiative [22], we also cover the performance issues of multithreading and distributed computing under the multi-core architecture support [23].
3. Essentials in Service-Oriented Software Development: This unit covers the essential components and gives a quick start of developing service-oriented software, including development environments, service-oriented computing standards and interfaces, developing services as service providers, understanding service brokers, discovering and publishing services in service brokers, and developing service clients that consuming services.
4. XML Data Representation and Processing: This unit discusses XML and related technologies supporting service-oriented computing, including XML fundamentals, XML data processing in SAX, DOM, and XPath models, XML type definition and schema, XML validation, and XML Stylesheet language.
5. Web Application and Web Data Management: This unit elaborates the development of Web applications using service orientation. It covers the models of Web applications, structure of Web applications, state management in Web applications, including view state, session state, cookies, application state, and file system. Database and caching support to Web application state management are discussed in the next course, as a tradeoff between the materials in the sequence of two courses. This unit also covers dynamic graphics generation to leverage the presentation of Web applications at the programming level.
6. Dependability of Service-Oriented Software: Dependability, including reliability and security, is a more important issue in Web applications than that in desktop applications. In addition to present essential issues in dependability design of Web applications, this unit also designs and implements the security mechanisms that safeguard the Web applications.

For all the major concepts and principles in the course, working examples are given to make sure that students not only understand the concepts, but also know how to apply the principles to implement operational software. Figure 3 shows an example used in Unit 5, which illustrates the development of an online bookstore, consisting of the catalog page for buyers, add-to-catalog page for sellers, and the shopping cart service that keeps track of the books selected by a buyer.

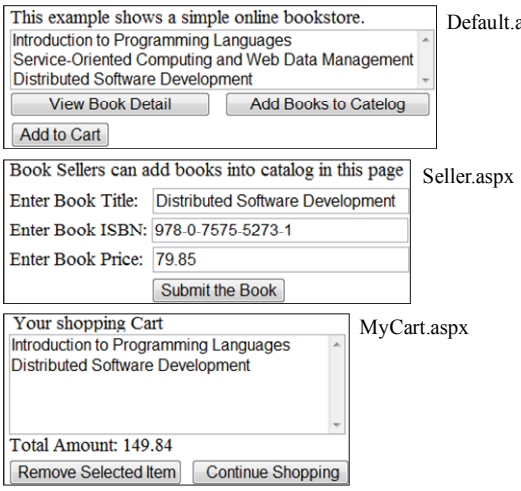

Figure 3. Developing an online bookstore

For each unit, students will complete a development project to enforce the contents covered in the lectures. These projects are:

1. Getting started with service-oriented software development, in which students will follow the tutorials in the appendix of the text to develop their first software.
2. Developing an e-commerce software using event-driven multithreading programming. In this project,

students will write a chicken-farm thread that supplies chickens at dynamic prices and multiple stores that buy chickens from the farm. Whenever a chicken price cut event occurs, the stores will be alerted. The stores will order chickens based on the price and stock.

3. Developing and deploying services. In this project, a group of students will work together. Each student will develop certain number of services, and the students in a group must coordinate to develop services that can be used in the same application. The services must be deployed into a cloud server.
4. XML processing. In this project, students will create XML files, XML schema files, and style sheets. They will write program to process the XML files, check the XML files against their schema, and convert XML files into different formats according to the style files.
5. Creating Web application. Students will work in a group to create Web applications. They use the services they create in project 3 and use public services discovered in public service directories and repositories. The applications must be deployed into a cloud server.
6. Adding security into the Web application. Students must extend their Web applications with security features, such as user registration and access control.
7. Graduate students in CSE598 section of the course will participate in a course working on service-oriented computing. They write a research paper on given topics and peer review each other research papers. As we have between 30 to 50 graduate students in each semester, the course workshop is running like a real workshop. Students learn research, writing, reviewing, and the complete cycle of publishing a paper. The papers are actually published online and linked to the course Web site [16].

Figure 4 illustrates the topics and the relationship among the topics in SOA, SOC, and SOD covered in CSE445 and the CSE494 courses. As the domain is new, no existing textbooks adequately cover these materials. There are books that cover the concepts and principles well, but they do not discuss the development of operational software. There are books that focus on service-oriented software development. Those books are mostly platform-based and do not associate the concepts and principles to the software in development. We have developed a textbook to facilitate this course, which consists of detailed explanation of concepts, using examples to illustrate each key concept, and using case studies to link multiple concepts together and to provide working examples. Part I of the text [8] is dedicated to teach this course. Assignments and projects are given at the end of each chapter.

Students taking CSE445 are expected to have good programming background in an object-oriented programming language such as C++, Java, or C#, and basic software engineering background. All development tasks are language-based either in Java or in C#, including multithreading software development, Web service development, and Web application development. In contrast, the next course, CSE494/598, will focus more on the software composition and integration using existing services and components.

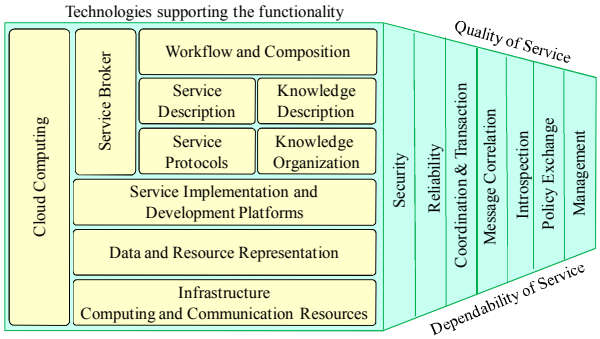

Figure 4. Organization of the SOC-enabling technologies

IV. SOFTWARE INTEGRATION AND ENGINEERING

CSE494 (Software Integration and Engineering) course is built on the basic concepts and principles discussed in CSE445, yet they do not rely on the detail of CSE445. If both courses are offered, CSE445 should be offered before CSE494. However, with a review of the basic concepts and preparation in XML, this course can be taught independent of CSE445. CSE494 emphasizes software composition and integration using existing services and components. The approach is based on higher-level of data management and application building techniques. The objectives and outcomes of the course are [17]

1. To understand software architecture and software process
   a. Students understand the requirement and specification process in problem solving.
   b. Students understand software life cycle and process management
   c. 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
   a. Students can apply software architecture to guide software development in the problem solving process.
   b. Students understand interface requirement of software services

c. Students can compose software based on interfaces of services and components
d. Students can develop software system using different composition methods and tools

3. To understand and apply data and information integration in software development
   a. Students can compose software systems using different data resources in different data formats.
   b. Students can integrate application logic with different databases.
   c. Students can apply the entire software life cycle to develop working software systems.

CSE494 is designed to achieve the objectives and outcomes of course. The course materials are organized in seven units.

1. Service-Oriented Application Architecture: This unit reviews service-oriented computing concepts, and it particularly focuses on the application's architecture using existing services and dynamic architecture that allows reconfiguration and re-composition of services used in the application.
2. Advanced Services and Architecture-Driven Application Development: The unit covers the development of self-hosting services without using a Web server and RESTful services. It also covers the application development using these services.
3. Enterprise Software Development and Integration: This unit uses BPEL (Business Process Execution Language), Workflow Foundation, message-based integration (e.g., Enterprise Service Bus), mashup composition, and other development environments to develop enterprise software. It focuses on software integration using existing components and using workflow, instead of using the traditional programming languages, to compose software.
4. Service-Oriented and Event-Driven Robotics Applications: This unit uses another service-oriented and workflow-used composition language, Visual Programming Language (VPL), for application composition. VPL is specifically designed for event-driven software development in robotics applications. Robot as a Service (RaaS) and Cyber-Physical Devices are also covered, which extend cloud computing from virtual space to the physical world [18][19].
5. Interfacing Service-Oriented Software with Databases: This unit discusses the interfaces between service-oriented software and databases, including relational databases and XML databases. LINQ (Language Integrated Query) is used to handle the queries to relational databases, XML databases, and objects in a uniform way. Caching and recommendation based on social data are also discussed.
6. Ontology and Semantic Web: This unit studies the basic resource representation and the ontology languages for describing resource, property, and structure that form the ontology. It covers ontology languages RDF, RDFS, and OWL.
7. Cloud Computing and Software as a Service: Cloud computing offers an infrastructure that enables rapid delivery of computing resources as a utility in a dynamic, scalable, and virtualized manner. This unit covers the basic concepts, principles, enabling technologies behind cloud computing and software as a service. Multiple cloud computing environments and their application are discussed in detail.

Cloud computing is a key unit in the course, as it is the latest technology that all the major computing companies and governments are supporting and utilizing. In the U.S. Federal Cloud Computing Strategy, published in 2011, Vivek Kundra, the U.S. Chief Information Officer, planned to spend $20billion of the entire government's $80 billion IT budget in cloud computing [20].

Another highlight of the course is of teaching the workflow-based software development, which turns the dream of generating executable directly from the flowchart into reality. A keynote on this topic was given in JICSIT 2011 / ITAIC 2011 [24].

Similar to CSE445/598, a project is given at the end of each unit. These projects include a survey on cloud computing, RESTful service development, Web applications consuming RESTful services and other types of services, workflow-based integration, BPEL-based integration, database and ontology development. Figure 5 shows typical example of developing an image verification application. In this example, RESTful services are used for generating a random string and then generating the image string. Such an application is found in most registration pages to differentiate a human user from a programmed user.

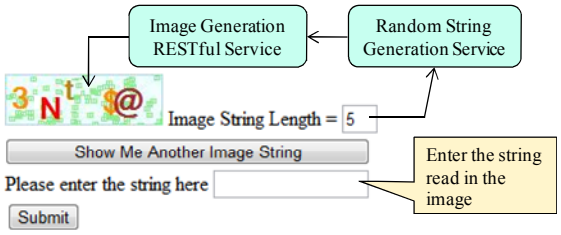

Figure 5. Application integration using existing services

As the domain is new, no existing textbooks adequately cover these materials. We have covered these materials in the same textbook used for CSE445 [8]. Part II of the textbook is dedicated to CSE494. Assignments and projects are given at the end of each

chapter. Notice that the Part III, Appendices of the same text, is used for CSE101. Thus, the same text is used for multiple service-oriented computing courses that we discussed in the paper. Textbook is further discussed in Section VI.

V. SOFTWARE DESIGN WITH SAAS

CSE564 (Software design) is a core course in software engineering for senior and beginning graduate course that cover various software design techniques. The syllabus was written so that the course can be easily updated with new materials. In the last two years, this course has covered SaaS with the following topics:

Introduction to cloud computing: This introduces SaaS, PaaS, cloud computing such as GAE (GFS and BigTable), Dynamo, and MapReduce.

Introduction to SaaS: This introduces SaaS, SaaS maturity models, various SaaS structures such as CCSOA [25], GSE [26], UCSOA [27], EasySaaS [32].

Customization: This introduces tenant customization design and processes, variability points, MTA, and OIC model of customization [28].

Database design and partitioning: This introduces the database design, data partitioning, and data in SaaS.

Ontology and SaaS design: This introduces ontology, its applications in SaaS; how to use ontology to support SaaS.

Service composition and recommendation: This introduces service composition and recommendation in SaaS such as the Grapevine model.

SaaS modeling, code generation and resource management: This introduces a model-based approach to construct SaaS include modeling, model-based code generation, and intelligent resource management.

Recovery, Fault-tolerance, Recovery, and Scheduling: SaaS and PaaS fault tolerance and techniques, redundancy management [31], scalable MTA scheduling.

Scalability and Security: This introduces scalablity issues and metrics, ways to evaluate scalablity and security mechanisms such as filtering and isolation.

Automated SaaS Testing: This introduces automated testing in SaaS, automated testing is important as SaaS integrates software development, execution, and testing together in an integrated system.

Sample SaaS Implementations: This introduces several simplified SaaS implementations for students to exercise and experiment. The sample systems illustrate various features of EasySaaS.

Note that the topics include both SaaS application and SaaS infrastructure development. SaaS application development includes using PaaS, or PaaS with a SaaS infrastructure to develop SaaS applications. SaaS infrastructure development involves constructing a design framework for tenant customization and a runtime to manage SaaS application execution. SaaS infrastructure can be developed on top of a PaaS or a database management system. For example, one step of SaaS application development is tenant customization process and it involves database schema, composing SaaS applications using services and GUIs. SaaS infrastructure design involves database design, partitioning, and matching algorithms, and recommendation algorithms, and a runtime monitoring, provisioning, and recovery algorithms to manage SaaS application execution.

As a part of regular course, CSE564 also covers traditional software design topics such as design-for-change, design patterns, object-oriented frameworks, SOD, traditional software architecture such as the Blackboard architecture. But 60% of the course addresses SaaS.

As most of SaaS contents are new and thus students will learn from research papers, PPT, white papers, and public videos such as Google I/O, even though the textbook contains a chapter on these related topics.

Furthermore, the SaaS contents change significantly as the field advances. In 2010, students were taught these concepts, but in 2011, students in CSE564 construct a small SaaS called MiniSaaS using GAE using MTA. Next year, students will experiment the MiniSaaS for resource management, scheduling, and fault tolerance. In addition to the MiniSaaS, students are also exposed to several small SaaS design so that they are comfortable to design one code base for a variety of applications. Students have reacted positively to these MTA SaaS samples. We plan to make the MiniSaaS open source so that other universities can also exercise and experiment the SaaS application and SaaS infrastructure development.

MiniSaaS is essentially a simplified version of EasySaaS, and it has GUI, workflow, service, and data customization, MTA. It can also have its own runtime infrastructure. Instead of developing SaaS application by developing the corresponding code, SaaS application can be done by the service-oriented manner, i.e., publishing, discovery, composition, deployment, and management. Instead of having its own runtime infrastructure, it can be run on top of a commercial PaaS such as GAE.

VI. ASU REPOSITORY OF SERVICES AND APPLICATIONS

The purpose of SOC is to improve the quality and productivity of software and application development. It can succeed only if a large repository of services is available. Ideally, all services developed by all corporations and by individuals will be open to the

public in public directories and repositories. The current situations are:

- Private Services: Many corporations, for example, IBM and SAP, keep their repositories private for internal use only. These services will not be available for education purposes.
- Paid Services: Many corporations, for example, Amazon Web Services, offer commercial services and subscription and payments are required. It is obviously correct and necessary for the application holders to pay for the services they use, in order to reflect the value of the services and the entire service-oriented computing paradigm, as well as maintain the service agreement between the service providers and service clients. However, such services are not useful for education purposes, as we cannot ask students to use these services to build their course projects, although many of our students paid for the services in order to develop better assignment projects.
- Free public services: There are a number of service directories where free public services are listed, including Xmethods.net, Webservicex.net, and remotemethods.com. Google and Microsoft offer free services and APIs in a number of areas in search and map services.

Free public services are great resources for education purposes, which are the main sources our students use to develop their applications. There are several problems with the free public services.

- The number of services and the range of services are limited.
- The performance of some of the services is not adequate. The services are too slow to use (frequent timeout when many students are accessing). The situation occurs particularly before an assignment is due and a large number of students are accessing the free services.
- The availability, reliability, and maintainability are not warranted. Services are often offline or be removed without notice. Service interfaces and implementations can be modified too.

To reduce the possible problems, we have developed a repository of our own at:

http://venus.eas.asu.edu/WSRepository/repository.html

This repository complements the free public services in several ways. We develop services according the need of the course and its assignments. The source code of the services and applications are open and explained as examples in the text [8]. The services and applications include simple function services that illustrate the development process, for example, encryption and decryption services, access control services, random number guessing game services, random string (strong password) generation services, dynamic image generation services, random string image (image verifier) service, caching services, shopping cart services, messaging buffer services, and mortgage application/approval services. The services are implemented in multiple formats, including ASP .Net services, Windows Communication Foundation services, RESTful services, and Work Flow services. All the services are free and open to the public. We maintain the server to keep the high availability and reliability of the services.

We also developed a service directory that lists services offered by other service directories and repositories using a service crawler that discovers available services online. The service directory can be accessed through a service engine at: http://venus.eas.asu.edu/sse/. We also offered a registration page for anyone to list their services into the service directory. The registration is at:

http://venus.eas.asu.edu/sse/ServiceRegister.aspx

VII. THE TEXTBOOK

As we discussed in the previous sections, a textbook is necessary to support these new courses and is developed that summarizes the main contents of the four courses in SOA, SOC, SOD, SaaS [8]. Three editions have been published, and the third edition of the book has the following chapters, which are organized in three parts for the three courses. The cover page of the book in Figure 6 shows the key topics of the text, where SOC, SOC, SOD, and SaaS concepts and principles are the center of all the related technologies.

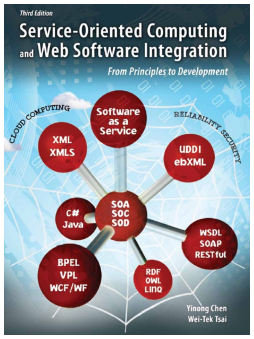

Figure 6. SOC, SOC, SOD and SaaS

The reasons for frequently updating are twofold. First, this is a new domain and we have been learning the materials and learning the feedback from our students. Second, domain is emerging and new methods and technologies are occurring rapidly. After the third editions of intensive revision of the text, we believe we have a set of stable materials.

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 Data Representation and Processing
Chapter 5	Web Application and Data 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	Service-Oriented and Event-Driven Robotics Applications
Chapter 10	Interfacing Service-Oriented Software with Databases
Chapter 11	Ontology and Semantic Web
Chapter 12	Service-Oriented Application Architecture
Chapter 13	A Mini Walkthrough of Service-Oriented Software Development
Chapter 14	Cloud Computing and Software as a Service
Part III	**Appendix: Tutorials on Service-Oriented Software Development**
Appendix A	Web Application Development
Appendix B	Service-Oriented Robotics Applications
Appendix C	ASU Repository of Services and Applications

Part I of the text is used for CSE445/598. Part II is used for CSE494/598. Appendix B on Service-oriented Robotics Applications in Part III is used for CSE101/FESE100. Chapter 9 in Part II can also be covered after Appendix B. CSE564 is largely based on Chapter 14, the Cloud Computing, with many supplementary materials.

The text is available publisher's site [8] and at http://www.amazon.com/.

VIII. Enrollment and Evaluation

In 2004, computer science related majors started to decline in enrollment. CSE101 was designed to improve the first programming experience and to remedy the decline in students' interest in computer science studies. CSE101 uses service-oriented robotics application development to achieve its goal and was first offered in Fall 2006. Table 1 list the enrollment of the class. As CSE101 is the first class Computer Science (CS) and Computer System Engineering (CSE) students are required to take in our school, fall semesters have significantly more students than in spring semesters, since most new students enter the school in fall semesters. To see the growth of students, we listed enrollments in fall and spring semester, respectively. In the first two years, CSE101 is offered to CSE students as an experiment. The class received encouraging feedbacks and the course was then required for both CSE and CS students. As can be seen that the enrollment of the course has constantly increased in spring and in fall semesters, respectively. As all CS and CSE students are required to take CSE101, the enrollment number of the course largely reflect the enrollment number of the programs.

Table 1. CSE101 enrollment since Fall 2006

Year	Semester	No of sections	Enroll-ment total	Programs required
2006	Fall	2	73	CSE
2007	Fall	2	75	CSE
2008	Fall	3	117	CS, CSE
2009	Fall	3	112	CS, CSE
2010	Fall	5	220	CS, CSE
2011	Fall	8	320	CS, CSE, IE
2007	Spring	1	26	CSE
2008	Spring	1	24	CSE
2009	Spring	2	74	CS, CSE
2010	Spring	3	66	CS, CSE
2011	Spring	3	105	CS, CSE

Figure 7 plots the data, which visualize the increase of the enrollment in both spring and in fall semesters.

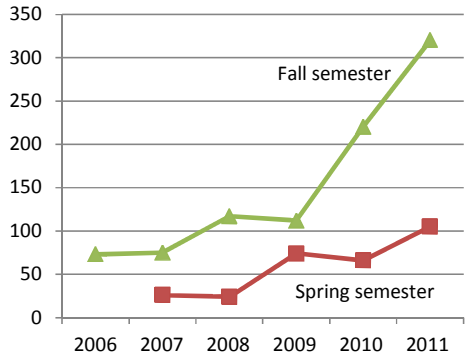

Figure 7. CSE101 enrollment 2006 to 2011

From Fall 2011, another program, Industrial Engineering (IE), is added to the required list of CSE101. The course is renamed to FSE100, which is required for all students in ASU Ira A. Fulton Schools of Engineering (FSE). In additional to CS, CSE, and

IE, FSE also include the many all other engineering schools, such as aerospace engineering, bioengineering, chemical engineering, civil engineering, electrical and electronic engineering, environmental engineering, mechanic engineering, and material engineering.

At the same time in 2006, the authors were tasked to redevelop CSE445 to reflect the new development paradigm in distributed software development. CSE445 was named Distributed Computing with Java and CORBA. The course was renamed Distributed Software Development and the contents are changed to base on service-oriented computing, as discussed in the previous section. CSE445 is required for Software Engineering concentration, which is a part of CS program. There are about 50 students are enrolled in SE concentration. CS and CSE students can take CSE445 as a credit elective. CSE445 is also listed as CSE598 as an elective of graduate students. The enrollment numbers of CSE445/598 are given in Table 2.

Table 2. CSE445/598 enrollments since Fall 2006

Year	Semester	445 enrollment	598 enrollment	Enrollment total
2006	Fall	25	14	39
2007	Spring	16	16	32
2007	Fall	24	21	45
2008	Spring	39	8	47
2008	Fall	35	23	58
2009	Spring	38	13	51
2009	Fall	33	10	45
2010	Spring	38	22	60
2010	Fall	42	34	76
2011	Spring	50	20	70
2011	Fall	30	52	82

Figure 8 plots the data, which visualize the enrollment in of CSE445 and CSE598. Both sections show significant increase from 2006 to 2011. The combined enrollment has increased from 39 in Fall 2006 to 82 in Fall 2011.

Course evaluation is done at the end of the course by all students. Table 3 shows the average scores of CSE445/598 (Distributed Software Development) since Fall 2006. The scores are out of 5.0, where 5.0 is very good, 4.0 is good, 3.0 is fair, and 2.0 is poor.

Table 3. CSE445/598 student evaluation scores

Year	Semester	445 score	598 score
2006	Fall	3.69	4.37
2007	Spring	3.99	4.13
2007	Fall	4.03	4.33
2008	Fall	4.52	4.81
2009	Spring	4.22	4.37
2010	Spring	4.44	4.46
2010	Fall	4.56	4.63
2011	Spring	4.49	4.52

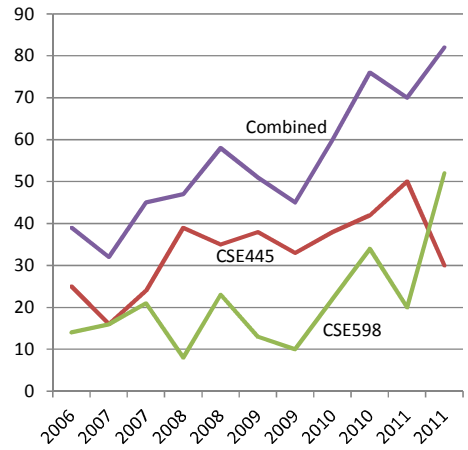

Figure 8. CSE445/598 enrollment 2006 to 2011

In additional to numerical scores, ASU course evaluation also asks students answer three questions in writing. Below is a complete set of written feedbacks from CSE445 students in Spring 2010, without any addition or subtraction.

What did you like most about this course?

- The instructor is one of the best instructors I've had at ASU and this course is one of the most important courses I've taken at ASU
- real world technology applied today
- The new material/subject area
- I like the instructor very much! Actually I wish I can take more courses of his.
- Having an open book and open note quiz every class helped my motivation to attend class and helped me understand topics better.
- This course used a combination of daily lecture exercises, chapter quizzes, programming projects, and exams to test and reinforce the course material.
- I felt like this was a very successful way to keep students involved, and rewards consistent effort and engagement in the course.
- It's interesting and applicable in most of what I want to do.
- learning how to create a website and the services that they can provide
- Useful concepts learned by doing homework projects
- It one of the most relevant courses I've ever taken at ASU. It has real applications, and the projects/
- Homework are right on the money. This is stuff people can use.
- The topic of the class is a very interesting topic
- Very hands-on.
- Interesting subject matter, lots of application and programming.
- Learning Service Oriented Computing and the hands on experience you get with SOC. I also liked working with Visual Studio 2008. Before this class, there was very

little use of full featured developmental tools. I think 95% of everything I have done was on a plain text editor.
- I like that we learned about web based technologies which are very relevant in todays job market.
- Chen's courses are always very well put together. He inspires practical application by having us do it ourselves.

What did you like least about this course?

- It would have been nice to have all online content available
- Always a power point, never really ventured outside of that.
- Linked assignments 3 and 5. I don't like having assignments that build off each other because if you don't get the first one working correctly, you can't really do the second. I know this was an optional linkage but the group I am in didn't agree to start from scratch on something else.
- Nothing. I thoroughly enjoyed this course, and wish more of the courses I took at ASU were similar to this one.
- Very difficult time understanding his English.
- The in class quizzes,
- Lecture exercise EVERY DAY
- Sometimes he goes too much into specifics of how to open Visual Studio, how to add a Service
- Reference, etc. He should have tutorials for that if students find it helpful, but the class time should
- be spent on the more complicated aspects of what he is teaching.
- Lectures were boring. I often felt like I couldn't connect in-class examples to homework.
- Heavy focus on a single technology in a single programming language. The focus is kind of necessary since we have people who need that help, but in reality the concepts are language agnostic.
- Some of the instruction on how to do thing is VS2008 did not work as they were supposed to and you have to find other ways to get things to work correctly on your own.
- Everything was Microsoft... understood because it is supported by the school.

Comments

- Awesome course!
- Dr. Chen did an amazing job with this course.
- Team projects are not for everybody, aka. me.
- Unfortunately due to our capstone professor and the unnecessary amount of time that class consumed I was unable to spend an appropriate amount of time in this class. Due to those circumstances I don't feel like i got as much out of this class as i should have.
- Add some discussion or more examples to liven up the lectures.
- Very good for such a new class. Once the class gets a little more mature, it will be even better.

Based on the feedbacks from students, a follow up class of CSE445/598 class has been designed and pilot-taught in summer 2010 and summer 2011, with 20 students enrolled in both sections. The new course CSE494/598 has been approved as a required course for the Software Engineering Concentration from Spring 2012, which has the same status as CSE445/598. We expect the course to have similar number of enrollment.

IX. CONCLUSIONS

The paper presented four courses in SOA and SOC in the curriculum of computer science, computer system engineering, and software engineering at the Arizona State University. Two of the courses have been included in the curriculum since 2006, the third was added in 2010, and the fourth is a regular course updated with new contents. In addition to outlining the course contents, a few examples are shown in the paper to add certain detail of the types of materials covered in the courses. For all the major concepts and topics in the courses, working examples are given to make sure that students not only understand the concepts and principles, but also know how to implement the concepts and principles in operational software. Web deployment of software into a Web and cloud computing environments is another emphasis of CSE445/598 and CSE494/598, as it is not covered in any other object-oriented software development courses. The increase in the enrollment of the courses as well as the feedbacks from students show that these courses improved our curriculum and are well received by students. Resources are created and made available for other universities to adopt.

X. ACKNOWLEDGEMENT

This paper is based projects sponsored by U.S. National Science, Foundation project DUE 0942453, and the European Regional Development Fund and the Government of Romania under the grant no. 181 of 18.06.2010.

REFERENCES

[1] Carlo Ghezzi, Mehdi Jazayeri, Programming Language Concepts, 3rd Edition, John Wiley & Sons, New York, 1998.

[2] Yinong Chen, W.T. Tsai, Introduction to programming languages: Programming in C, C++, Scheme, Prolog, C#, and SOA, 2nd Ed., Kendall Hunt Publishing, 2006.

[3] D. Krafzig, K. Banke, and D. Slama, Enterprise SOA: Service-Oriented Architecture Best Practices, Prentice Hall, PTR, 2005

[4] Th. Erl, Service-Oriented Architecture: Concepts, Technology, and Design, Prentice Hall, August 2005.

[5] M. P. Singh, M. N. Huhns, Service-Oriented Computing, John Wiley & Sons, Atrium, 2005.

[6] Douglas C. Schmidt, Distributed Object Computing with CORBA Middleware, http://www.cs.wustl.edu/~schmidt/corba.html

[7] Thuan L. Thai, Learning DCOM, O'Reilly Media, 1999.

[8] Yinong Chen and W.T. Tsai, Service-Oriented Computing and Web Software Integration, 3rd Edition, Kendall Hunt Publishing, 2011, http://www.kendallhunt.com/Author.aspx?id=925

[9] R. Paul, "DoD Towards Software Services," Proc. of IEEE International Workshop on Object-oriented Real-time Dependable Systems (WORDS 05), February 2005, pp. 3–6.

[10] Yinong Chen, "CSE101 Resources", http://www.public.asu.edu/~ychen10/teaching/cse101/

[11] Yinong Chen, "CSE101 Laboratories", January 2011, http://www.public.asu.edu/~ychen10/teaching/cse101/labs.pdf

[12] W. T. Tsai, Yinong Chen, Yann-Hang Lee, James Collofello, Gary Bitter, "Preparing high school teachers for service-oriented computing", the U.S. Department of Education, Fund for the Improvement of Post-Secondary Education (FIPSE), Project ID: P116B060433.

[13] W. T. Tsai, Yinong Chen, Calvin Cheng, and Xin Sun, Gary Bitter and Mary White, "An Introductory Course on Service-Oriented Computing for High Schools", Journal of Information Technology Education, Volume 7, pages 323-346.

[14] W. T. Tsai, X. Sun, Y. Chen, Q. Huang, G. Bitter, and M. White, "Teaching Service-Oriented Computing and STEM Topics via Robotic Games," Proc. of IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC), 2008, pp. 131–137.

[15] Yinong Chen, ASU Summer Robotics Camps, http://venus.eas.asu.edu/roboticscamp/

[16] ASU SCIDSE Syllabus, "CSE445/598 Software Integration and Engineering", http://www.public.asu.edu/~ychen10/teaching/cse445/

[17] ASU SCIDSE Syllabus, "Software Integration and Engineering", http:// www.public.asu.edu/~ychen10/teaching/cse494sie/

[18] Yinong Chen, Zhihui Du, and Marcos Garcia-Acosta, M., "Robot as a Service in Cloud Computing", In Proceedings of the Fifth IEEE International Symposium on Service Oriented System Engineering (SOSE), Nanjing, June 4-5, 2010, pp.151-158.

[19] Yinong Chen, "Assuring Mobile Physical Services for the New Generation of Networks", Panel Paper, in 10th International Workshop on Assurance in Distributed Systems and Networks (ADSN), Hiroshima, March 2011, pp.585-588.

[20] Vivek Kundra, Federal Cloud Computing Strategy , February 8, 2011, http://ctovision.com/wp-content/uploads/2011/02/Federal-Cloud-Computing-Strategy1.pdf

[21] Yinong Chen and W.T. Tsai, ASU Repository of Web Services and Web Applications, http://venus.eas.asu.edu/WSRepository/repository.html

[22] Intel - Arizona State University Collaboration in Parallel Computation, http://engineering.asu.edu/cidse/Curriculum

[23] Yinong Chen, Performance of Multithreading with Many-core Support, Intel Academic Community Courseware, http://software.intel.com/en-us/courseware/course/view.php?id=594

[24] Yinong Chen, A Dream of Software Engineering: Service Orientation and Cloud Computing, JICSIT 2011 /ITAIC 2011 Keynote, http://www.public.asu.edu/~ychen10/activities/ jicsit11/ChenKeynote11.pdf

[25] W. T. Tsai, B. Xiao, Y. Chen, and R. A. Paul, "Consumer-Centric Service-Oriented Architecture: A New Approach," in Proceedings of SEUS-WCCIA' 06. Washington, DC, USA, IEEE Computer Society, 2006, pp. 175–180.

[26] W. T. Tsai, B. Xiao, R. Paul, Q. Huang, and Y. Chen, "Global Software Enterprise: A New Software Constructing Architecture," in Proceedings of CEC-EEE '06. Washington, DC, USA: IEEE Computer Society, 2006, p. 55.

[27] M. Chang, J. He, W. T. Tsai, B. Xiao, and Y. Chen, "UCSOA, User-Centric Service-Oriented Architecture," in Proceedings of ICEBE '06, Washington, DC, USA, 2006, pp. 248–255.

[28] W. Tsai, Q. Shao, and W. Li, "OIC: Ontology-based Intelligent Customization Framework for SaaS," in Proceedings of SOCA. IEEE, 2010, pp. 1–8.

[29] W. Tsai, Y. Huang, and X. Bai, "Grapevine Model for Template Recommendation and Generation in SaaS Applications," Arizona State University, Tempe, AZ, USA, 2011.

[30] W. Tsai, G. Qi, and X. Bai, "AgileSaaS: An agile SaaS Development Framework," Arizona State University, Tempe, AZ, USA, 2011.

[31] W. Tsai, Q. Shao, Y. Huang, and X. Bai, "Towards a Scalable and Robust Multi-Tenancy SaaS," in Proceedings of the Second Asia-Pacific Symposium on Internetware. ACM, 2010, p. 8.

[32] W. T. Tsai, Y. Huang, and Q. Shao, "EasySaaS: A SaaS Development Framework," Proc. of IEEE SOCA, 2011.