

Introduction to Service-Oriented Computing

W.T. Tsai and Yinong Chen

Computer Science and Engineering Department, Arizona State University

Gary Bitter and Dorina Miron

Technology Based Learning and Research, College of Education, Arizona State University

Table of Contents

1. The Problem: Too Few Software Engineers and Too Many Programmers.....	1
2. What are SOA and SOC?.....	2
3. Who is using SOA and SOC?	2
4. Why are SOA and SOC research and education important?	3
5. Differences between object-oriented and service-oriented computing paradigms	3
6. The need to teach SOC as the first programming language.....	4
7. Feasibility of teaching the SOC paradigm in high schools and in university lower divisions	6
8. ASU SOA and SOC-based Research.....	8

1. The Problem: Too Few Software Engineers and Too Many Programmers

A serious problem currently experienced by most computer science and engineering programs in the U.S. is the significant decline in enrollment, due to the facts of offshoring labor-intensive programming jobs and the layoff of programmers in the past years. Perspective students and parents were not well educated on the difference between software engineers and programmers. The difference is: Software engineers define software specification to meet the requirements of applications, while programmers take a given specification and write a program to implement the specification. The fact is, computing industries are struggling to fill their vacancies with qualified people (Gates & Klawe, 2005; Lohr, 2005). This shortage is aggravated by the scarcity of software engineering skills. According to CNN Money Magazine's ranking list, published on April 12, 2006 [<http://money.cnn.com/magazines/moneymag/bestjobs/top50/index.html>], software engineer jobs top the list and are rated the best jobs in America, with 46.07% forecast growth rate in the next 10 years. Unfortunately, the current education system produces massive programmers (programming is taught from high schools to universities, in all majors) and very few software engineers (software engineering is taught in the upper divisions of CS/CSE departments of universities only), based on the current understanding that, to become a software engineer, one must be the best programmer first. The SOA paradigm will explicitly separate software engineering task from programming task, and thus create software engineers without first making them the best programmers.

2. What are SOA and SOC?

The Service-Oriented Computing (SOC) paradigm refers to the set of concepts, principles, and methods that represent computing in Service-Oriented Architecture (SOA) in which software applications are constructed based on independent component services with standard interfaces.

The main idea of SOC/SOA is to explicitly separate software engineering from programming, to emphasize on software engineering, and to deemphasize on programming. SOC separates software development into three independent parties: Application builders (by software engineers), service providers (by programmers), and service brokers (joint effort from standard organizations, computer industry, and government).

Service providers: They use a traditional a programming language such as Java, C++, or C# to write program components. All components will be wrapped with open standard interfaces, call services, or Web services if the services are available over the internet, so that application builders can simply use the services without further communication with the service providers. The same services can be used by many applications.

Service brokers: Allow services to be registered and published for public access. Help application builders to find services they need.

Application builders: Instead of constructing software from scratch using basic programming language constructs, the application builders represent the end users to specify the application logic in a *high-level specification language*, using standard services as components. Application builders are software engineers who have a good understand of software architecture and the application domain.

As the number and types of available services increase, the need for writing new services and the need for programmers will drop. On the other handle, as computer applications move into more and more domains, the need for application builders will increase. Using an analogy example, service providers are manufacturers of Lego pieces, while the application builders are kids (architects) who use Lego pieces to build millions of different Lego-based “products”. We do not need many people who can design different types of Lego pieces, but we need many architects to build different products.

3. Who is using SOA and SOC?

All major computer corporations, including BEA, IBM, Microsoft, Oracle, HP, SAP, Intel, Cisco, Juniper, SAP, and Sun Microsystems, have moved towards the SOC paradigm. Languages, protocols, and standards have been developed to support and to regulate SOC applications. Numerous frameworks, specification languages, and tools have been created.

Furthermore, government agencies, such as the U.S. Department of Defense (DoD) and NASA, have adopted SOC. In fact, *all* of DoD’s major IT initiatives in the past four years were based on the SOC paradigm, including the Army’s FCS, the Navy’s FORCEnet, the Air Force’s JBI, and the OSD’s NCES and GIG-ES (Paul, 2005).

In this ASU SOA Workshop, the three keynotes will be delivered by directors or researchers from DoD, IBM, and Microsoft.

SOC is also being adopted by major computer users, including banks (Web banking services), retailers (Web shopping services), airlines (Web booking services), travel agencies (Web composite services that link together the services from airlines, hotels, car rentals etc). Universities around the world (e.g., US, UK, Japan, Germany, China, and Singapore), including Arizona State University (ASU), have geared their computing research toward SOC-based modeling languages, automated code generation, service verification, and validation. ASU

developed the PSML-S (Process Specification and Modeling Language for Services), which supports graphic drag-and-drop specification of application logic (Tsai et al., 2005a). The specification written in PSML-S can be automatically translated into executable C# code. The PSML-S has been used for DoD, Intel, and other industry-sponsored projects.

The popularity of SOC is confirmed by industry observers and market research organizations: 1) The prestigious Gartner Group predicted that 45% of the U.S. companies will use SOA by 2006, and 80% by 2007 (Syntelligence 2005); 2) The IDC (www.idc.com) predicted that WS-related project will have \$7.1 billions by 2006 with an annual compound rate at 116% per year in the next few years according to the IDC report #27093; 3) The Radicati Group, which is more conservative, still predicted that the SOA market will reach \$6.2 billions by 2008 with an annual compound rate of 50% (Foster 2004). *These numbers all point to exponential growth and enormous popularity of SOA and SOC in the U.S. economy.*

4. Why are SOA and SOC research and education important?

While major computer companies and government agencies are moving towards the SOC paradigm, the U.S. education system is lagging behind, failing to adjust to the paradigm shift that has already taken place in the industries. In universities, SOC is currently taught either at the senior/graduate level or not at all. Most SOC courses are in the experimental stage or have a seminar format. Although there is a clear and growing demand for software engineers with SOC expertise, only a small fraction of students graduating with a degree in computer science and engineering have SOC skills.

In the year 2000, Microsoft launched a program designed to teach Microsoft software in schools. .Net was included in the program, but the curriculum focused on the use of tools (.Net to write Visual Basic or C# to write traditional programs). Our mission focuses primarily on the teaching of the new programming paradigm (SOC) and only secondarily addresses the use of advanced programming tools such as ASP.Net (SOC part of .Net) and IBM WebSphere.

Similarly, IBM also proposed SOC-based *Service Sciences* as a new discipline (www.research.ibm.com/ssme). The discipline has both management and engineering components (www.research.ibm.com/ssme). Two universities (University of California at Berkeley and NC State University) are starting such programs in 2006. Numerous researchers from Oxford, Stanford, Carnegie Mellon, UCLA, Northwestern, Harvard, Yale, Maryland are also active in this new discipline. IBM sponsors these educational initiatives and activities because the services sector employs 75% of U.S. labor force (www.research.ibm.com/ssme) and SOC is the best technology for the service sector.

In an article titled “Education Key to SOA Success”, LaPlante (2005) reported the following: “Our second survey inquired about the chief challenges of implementing an SOA. This one had a surprise in it: the No. 1 roadblock, according to 56 percent of you, was the need to *educate people about SOA.*”

5. Differences between object-oriented and service-oriented computing paradigms

SOC is evolved from the object-oriented computing paradigm. However, there are significant differences between the two paradigms (Table 1).

Table 1. Differences Between Object-Oriented and Service-Oriented Computing

Features	Object-Oriented Computing	Service-Oriented Computing
Methodology	Application development by identifying tightly coupled classes. Application architecture is hierarchical based on the inheritance relationships.	Application development by identifying loosely coupled services and composing them into executable applications.
Level of abstraction and cooperation	Application development is often delegated to a single team responsible for the entire life cycle of the application. Developers must have knowledge of application domain and programming.	Development is delegated to three independent parties: application builder, service provider, and service broker. Application builders need to understand application logic and may not know how individual services are implemented. Service providers can program but do not have to understand the applications that use their services.
Code sharing and reuse	Code reuse through inheritance of class members and through library functions. Library functions have to be imported at compilation time and are platform dependent.	Code reuse at the service level. Services have standard interfaces and are published on Internet repository. They are platform-independent and can be searched and remotely accessed. Service brokerage enables systematic sharing of services.
Dynamic binding and re-composition	Associating a name to a method at runtime. The method must have been linked to the executable code before the application is deployed.	Binding a service request to a service at runtime. The services can be discovered after the application has been deployed. This feature allows an application to be recomposed at runtime.
System maintenance	Users need to upgrade their software regularly. The application has to be stopped to perform the upgrading.	The service code resides on service providers' computers. Services can be updated without users' involvement.

6. The need to teach SOC as the first programming language

Universities or colleges that currently teach SOC often require students to have mastered Object-Oriented (OO) techniques (e.g., UML and OO design). Once students have learned the OO concepts, it is difficult for them to switch to SOC quickly. In Spring 2005, Dr. Tsai taught a class on SOC at ASU to a group of 35 senior/graduate students. In the first assignment, only 2 out of 15 work groups turned in an SOC design for a class project. The other 13 groups turned in an OO design in spite of their training on SOC in the class. The students' difficulty to switch to SOC is similar to a phenomenon encountered during the previous paradigm shift, when FORTRAN programmers were trained to construct Pascal programs using structured programming as a key concept. Most trainees turned in FORTRAN-like designs that used the Pascal syntax. In other words, their mindset was still FORTRAN. This phenomenon has been observed repeatedly for 40 years, and it generated numerous debates over five generations of programming languages including FORTRAN, APL, COBOL, Algol, PL/1, Pascal, C, Ada, Prolog, Smalltalk, C++, and Java (Chen & Tsai 2006). Based on this experience, researchers have long accepted that the first programming language has a tremendous impact on the way programmers think (Jadud 2003). Note that previously the debates mainly focuses different programming styles (such as procedural vs. logic, or abstraction vs. pointers) or specific programming constructs, whereas the SOC brings in a new programming paradigm or way of

thinking about computing. That makes it even more difficult for students to switch from a previously learned object-oriented paradigm to the new service-oriented programming.

The dramatic differences between SOC and object-oriented computing call for different teaching approaches (Table 2). While researchers continue to have heated arguments about the first programming languages in education (Lea 1996, Jadud 2006), the debate is mainly at the programming language construct level. The teaching of SOC as the first programming course offers a drastically different approach: Instead of focusing on programming language constructs to solve problems, the SOC approach focuses on reusable components, which they utilize to solve problems. Thus, the debate between SOC and the traditional approach is not at the language construct level, but between two radically different approaches to problem solving.

Table 2. Comparison of Teaching Methods for Object-Oriented and SOC

Teaching Methods	Object-Oriented Computing	Service-Oriented Computing
Educational goal	Students learn programming language constructs and how to apply these constructs to develop program modules and small applications.	Students learn the overall application architecture and how to compose large applications using existing software services.
Teaching philosophy	Bottom-up: Advance from small-scale to large-scale programming.	Top-down: Advance from large-scale programming to detailed, small-scale programming.
Focus	Focus on hardware and software interface and system interaction. Focus on low-level programming techniques and low-level reusability rather than applications.	Focus on human-computer interactions as well as hardware/software interface and system interaction. Focus on architecture in key applications domains and high-level reusability via composition.
Curriculum Content	The syntax of the programming language, with an emphasis on the construction of program modules.	The software architecture and the use of existing services to compose applications within the service-oriented architecture.
Order of curriculum contents	Bottom-up: Learn procedural programming (basic computing constructs and flowcharts) before architecture design. Advance from small to large applications.	Top-down: Start with the overall Web-services architecture, followed by SOC and model-based computing. Bypass the object-oriented programming.
Learning activities	Developing applications from scratch.	Developing applications using existing tools, services and infrastructure.

For example, under the traditional programming paradigm, students are taught the syntax and semantics of constructs, and are asked to develop small programs. Under the SOC paradigm, students learn a repository of reusable services, possibly supplied by third parties and vendors, and then they are asked to compose applications by using the available services. First, a student does the visual modeling of the overall application, and then selects components for the application. Furthermore, SOC students are taught from the beginning that programming is no longer writing programs, but visual composition and reusing of existing components. As far as program evolution is concerned, a traditional programming course teaches that program evolution can be achieved by changing code. However, in SOC, program evolution may involve changing a component with another, or changing the visual model slightly and re-composing services to form a new application. These fundamental differences account for the much greater

ease and productivity of the SOC paradigm, which was not possible before the massive accumulation of components produced under the previous paradigm.

Teaching SOC as the first programming course also addresses a serious and well-known problem often encountered by IT professionals, i.e., programmers are often reluctant to reuse code developed by others due to irregularity of specifications and programming styles. The SOC offers a paradigm where services will be published in a standard specification language and have standard interface, and thus promotes software reusability and interoperability.

The proposers of the SOA center have extensive background in teaching lower division computer science courses (freshman and sophomore), and recently jointly completed a book titled *Introduction to Programming Languages* (Chen & Tsai 2006) with a new chapter on SOC. This book is for freshman/sophomore students who wish to learn a collection of common programming languages. This is probably the first such textbook that covers SOC among hundreds of introductory programming languages textbooks.

7. Feasibility of teaching the SOC paradigm in high schools and in university lower divisions

This section addresses the feasibility of teaching SOC in high schools and in university lower divisions.

Do high school students and teachers have sufficient background and knowledge to learn SOC?

SOC programming is a high-level visual modeling approach that will be easier to understand than traditional command line-based programming constructs. The visual modeling will be further assisted by animation and simulation tools, e.g., Microsoft Viso and DirectX, so that students can follow the execution flow visually. These features make SOC materials easy to understand and to teach. Visual programming tools (e.g., Visual Basic) have been found to be useful aids for the teaching of programming languages. They have been used in numerous high schools for many years. Even a difficult programming language like C++ can be taught in high schools today using a visual programming tool (Visual C++). For example, Sandusky High School in Michigan offers Visual C++ (<http://webserver.sandusky.k12.mi.us/compsci.htm>). In addition to common visualization capabilities, the course to be developed in the proposed project will use other features such as visual architecture modeling and automated code generation. Visual programming is perceived by most high school students as cool features, which facilitates involvement, motivation, and ultimately learning.

As an example, Figure 2 shows how to use the Microsoft ASP.Net 2005 framework to develop an application in a few simple steps. The programming task is to develop an online bookstore, which takes orders from clients, processes the orders, and places charges by using the services of remote banks. If a charge is successful, the desired book will be ordered from the publisher and delivered to the client. It is very difficult to program such a system from scratch using C++ or Java. However, if all the components (boxes in Figure 2) can be found in the service repository, the development task in SOC would be simple. What an application builder would have to do is (1) draw the model (Figure 2); (2) provide a text file containing the data to be changed between the services, for example, the data behind the message "getOrder" must contain the client's name, address, credit card information; (3) define the logic and flow (for example, in the node "processOrder", we need to define the order: first place the charge, wait for the result of the operation; if the charge is successful, accept the order, otherwise, reject the order).

Researchers at ASU have extensive experience with the utilization of the latest modeling languages and tools; they actually developed a SOC-based Process Modeling and Specification Language (PSML) that can specify the process and logic without using a programming language

like C++ and Java. Once a PSML-S model is developed and the remote services are linked, the executable code can be automatically generated and executed on the .Net framework, or released as a Web service to a Web server. The new paradigm takes a "modeling, automatic code generation, and execution" approach that can shorten task time from months in traditional programming languages to hours in SOC languages. The visual aid and high-level of abstraction make the development simple and reliable.

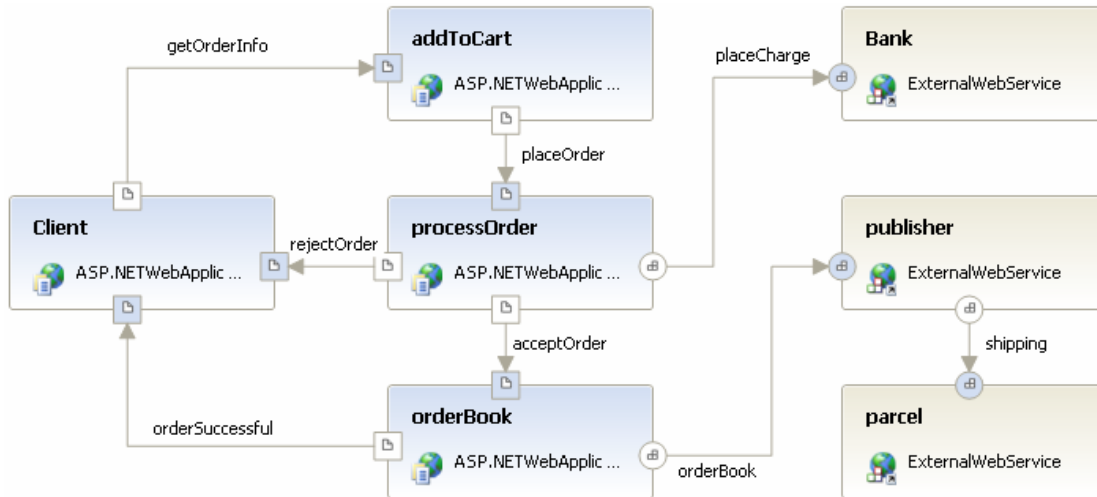


Figure 1. Specification of an Online Bookstore

Will the tools to be used in the course introduce vendor biases or preferences? The proposed project is designed to use as many open sources tools as possible, and will use a variety of tools from different vendors to avoid inculcating biases. This project will use Microsoft Visual Studio because it is the only integrated platform currently available that allows both architecture modeling and code generation for SOC. However, we will use the Shared Source Common Language Infrastructure (SSCLI), which is the open source version of the .NET Common Language Infrastructure. We will also use several IBM tools that are freely available for education purposes including IBM Websphere and SIBus (System Integration Bus). ASU researchers also developed a Specification and Modeling Language for Services (PSML-S) and associated tools (Tsai 2005a), sponsored by DoD. We are in the process of making PSML-S and its tools open source.

Is SOC mature enough to teach at the high school level?

SOC is not futuristic; it is actually being used by leading industry players on a daily basis. It may appear futuristic to the public because current computer science education and even computer science students know little about SOC unless they take a SOC graduate course or seminar. The main problem in SOC is not technology, but education, as pointed out by LaPante (2005).

Is there a reliable and readily available repository of services? Currently, several repositories of reliable services are available, for example, Amazon Web Service Developer Connection (<http://Developer.amazonwebservices.com>), or .NET XML Web Services Repository (www.xmlwebservices.cc), or Microsoft UDDI service registry at www.uddi.org. IBM UDDIS at <http://www-306.ibm.com/software/solutions/webservices/uddi/>. Many of these services are open source, and thus their reliability can be determined. Many vendors also provide their own repositories of services today.

Do the services available in repositories require verification before use in SOC applications?

Services need to be verified and validated, otherwise they cannot be used by application builders. Numerous approaches are available at this time. While this is still an active research area, practical solutions already exist: One solution by SAIC (Science Applications International Corporation) is to allow open source services only. Another solution is to allow services from business partners only because business partners share source code for verification (SAP takes this approach). The third approach is to have an independent agent for verification and validation. Services that pass verification and validation will have digital signatures. The third approach was proposed by Dr. Tsai in 2005, who is an active and leading researcher in this area, has a large number of publications on this topic, and pioneered the CV&V (Collaborative Verification and Validation) approach to verify and validate services at runtime. Recently, this approach has been adopted by DoD to verify services in DoD SOC systems.

8. ASU SOA and SOC-based Research

As shown in Figures 2 and 3, In software research laboratory at ASU, all research activities are centered around the SOA and SOC foundation, including

- Development of modeling and specification languages;
- Development infrastructure, related techniques and tools;
- Service-oriented system engineering;
- Service-oriented simulation and testing;
- Service-oriented reliability modeling evaluation;
- Application of SOC and SOA in embedded systems and robotics: **Our robot won the 2005 Ultimate Architect Sumobot Competition Championship Title:**
<http://asusrl.eas.asu.edu/EmbeddedExplorer/Publishings/ASU%20Fulton%20School%20of%20Engineering.mht>
- Application of SOC and SOA in mission-critical system: **Our E2E tool has been adopted by DoD in Command-and-Control systems.**
- Application of SOC and SOA in bio-engineering;
- Application of SOC and SOA in e-business and e-commerce;
- Application of SOC, SOA, and ontology in education.

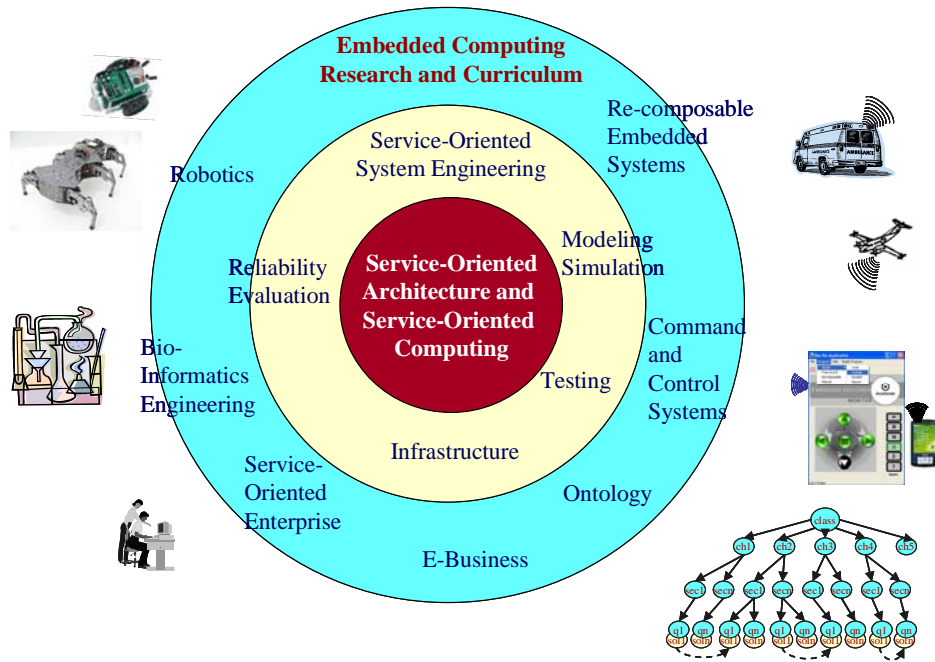


Figure 2. Overview of ASU SOA and SOC-based research

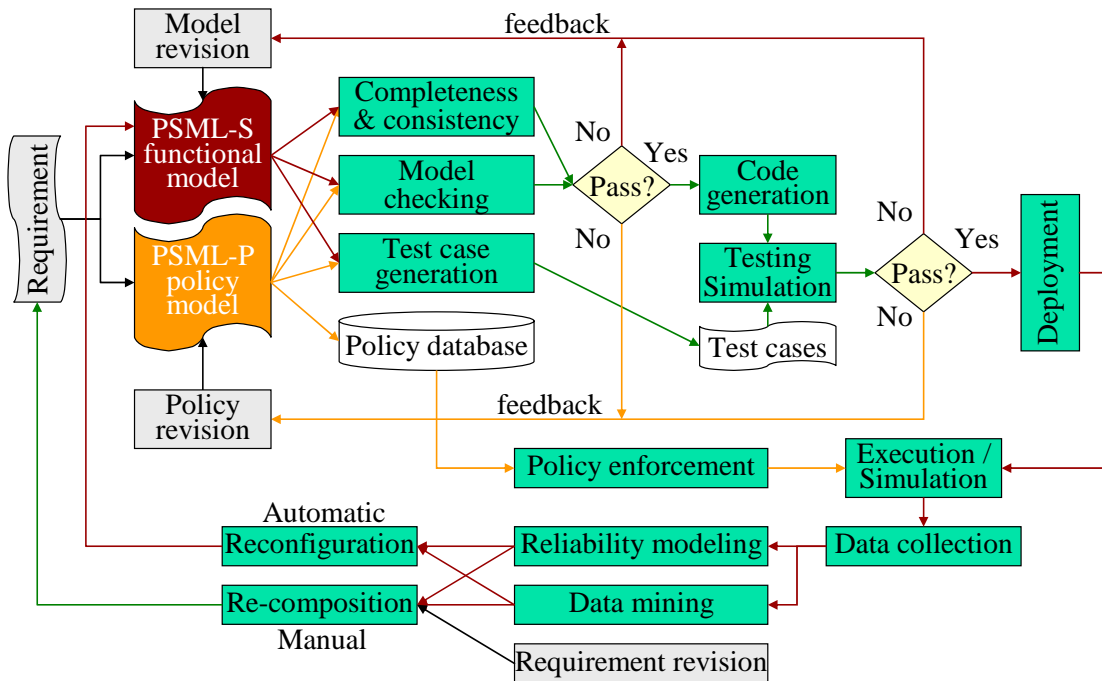


Figure 3. ASU Service-oriented development environment

References

- Achieve, Inc. (2000). Setting the Record Straight. Achieve Policy Brief. Issue Number One. Washington, D.C.
- American Association for the Advancement of Science (1993). Benchmarks for Science Literacy. New York, NY: Oxford University Press.
- American Association for the Advancement of Science (2001). Atlas of Science Literacy. Washington, DC: AAAS.
- Bergin, J. "fourteen Pedagogical Patterns", 2002, <http://csis.pace.edu/~bergin/PedPat1.3.html>
- Black, P., Harrison, C., Lee, C., Marshall B. and Wiliam, D. (2002). Working inside the black box: Assessment for learning in the classroom. London: Department of Education and Professional Studies, Kings College.
- Borko, H. (2004). Professional Development and Teacher Learning: Mapping the Terrain. *Educational Researcher*, 33 (8), pp. 3-15.
- Bransford, J.D., Brown, A.L., & Cocking, R.R. (1999). How people learn: Brain, mind, experience, and school. Washington, DC: National Academy Press.
- Catley, K., Lehrer, R., & Reiser, B. (2005). Tracing a prospective learning progression for developing understanding of evolution. Paper Commissioned National Academies Committee on Test Design for K-12 Science Achievement. <http://www7.nationalacademies.org/bota/Evolution.pdf>
- Chen, Y., Introduction to programming languages: Principles, C, C++, Scheme, and Prolog, Kendall/Hunt Publishing Company, 2003, ISBN 0-7575-0367-5.
- Chen, Y., Z. He, "The Simulation of a Highly Dependable Distributed Computing Environment", *Simulation, Transactions of the Society for Modeling and Simulation International*, Vol.79, No. 5 - 6, May/June, 2003, pp.316-327.
- Chen, Y., "A Service Scheduler in a Trustworthy System", in proceedings of the 37th Annual Simulation Symposium, Arlington VA, April 2004, pp. 89-96.
- Chen, Y., Service-Oriented Computing: Architecture, Programming, and Applications, the 9th International Conference on Software Engineering and Applications (SEA), November 2005. <http://www.iasted.org/conferences/2005/phoenix/sea-tutorial.htm>
- Chen, Y., H. Huang, W.T. Tsai, Scheduling Simulation in a Distributed Wireless Embedded System, *SIMULATION: Transactions of the Society for Modeling and Simulation International*, Vol. 81, Issue 6, June 2005, pp.425 - 436
- Chen, Y., C. Angderson, R. Mears, E. Mesa, "Web Service Composition and Demonstration for TestPro", Technical Report, Computer Science and Engineering Department, Arizona State University, December 2005, pp.1-29. <http://asusrl.eas.asu.edu/srlab/Publications/Reports/TestPro.pdf>
- Chen, Y., W.T. Tsai, *Introduction to programming languages: Programming in C, C++, Scheme, Prolog, C#, and SOA*, second edition, Kendall/Hunt Publishing Company, 2006.
- Ecksten, J., J. Bergin, and H. Sharp, "Patterns for Active learning", 2002, <http://csis.pace.edu/~bergin/patterns/ActiveLearningV24.html>
- Christensen, E., Francisco Curbera, Greg Meredith, Sanjiva Weerawarana, Web Services Description Language (WSDL) 1.1, W3C Note 15 March 2001. <http://www.w3.org/TR/wsdl>

- Chung, J.-Y., "Services sciences, management and Engineering", IBM Report, 2005. <http://www.research.ibm.com/ssme/>
- Chung, J.-Y., & Tsai, W. T. (2005, October). Future trends and directions for e-business engineering. Keynote address at the IEEE Conference on e-Business Engineering, Beijing, China. Retrieved October 15, 2005, from <http://www.cs.hku.hk/icebe2005/keynotes.htm>
- Clark, J., Inside "Indigo"--Infrastructure for Web Services and Connected Applications, Microsoft Press, April 2005.
- Esposito, D. (2004). Introducing ASP.NET 2.0. Redmond, WA: Microsoft Press.
- Etkina, E., Mestre, J. and O'Donnell, A. (2005). The Impact of the Cognitive Revolution on Science Learning and Teaching. In J.M. Royer (Ed.), The Cognitive Revolution in Educational Psychology. Greenwich, CT: Information Age Publishing.
- Foster, I., ComputerWorld, "Grids' Place in the Service-Oriented Architecture", Nov. 30, 2004, in TechWorld, <http://www.techworld.com/opsys/features/index.cfm?FeatureID=1029>.
- Friedman, Thomas, The World is Flat: A Brief History of the Twenty-First Century, Douglas & McIntyre, 2005.
- Gates, Bill, Maria Klawe, Keynote Address at Microsoft Faculty Summit 2005. <http://www.microsoft.com/billgates/speeches/2005/07-18FacultySummit.asp>
- Gomez-Perez, A., M. Fernandez-Lopez, O. Corcho, Ontology Engineering, Springer-Verlag, London, 2004.
- Gomez-Perez, A., Fernandez-Lopez, M., & Corcho, O. (2004). *Ontology engineering*. London: Springer
- Heinemann, F., Christian Rau, Web Programming with the SAP Web Application Server, SAP Press, 2003.
- Hestenes, D., Wells, M. and G. Swackhamer, G. (1992). "Force Concept Inventory," *The Physics Teacher*, (30);141-158.
- Huang, H., Tsai, W. T., Paul, R., & Chen, Y. (2005, May). Automated model checking and testing for composite web services. In Proceedings 8th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC), Seattle, WA (pp. 300-307). Los Alamitos, CA: IEEE Computer Society.
- Intel (2005). Service-oriented enterprise, the technology path to business transformation. Retrieved October 2, 2005, from <http://www.intel.com/business/bss/technologies/soe/>
- Jadud, M., My First Programming Language, 2003, <http://www.cs-ed.org/blogs/mjadud/archives/000244.html>.
- LaPlante, Alice, "Education Key to SOA Success", in SOApipeline.com, Nov. 21, 2005.
- Lea, D., Some Questions and Answers about Using Java in Computer Science Curricula, <http://gee.cs.oswego.edu/dl/html/javaInCS.html>.
- Lehrer, R., & Schauble, L. (2003). Origins and evolution of model-based reasoning in mathematics and science. In R. Lesh & H. M. Doerr (Eds.), Beyond constructivism: A models and modeling perspective on mathematics problem-solving, learning, and teaching. Mahwah, NJ: Lawrence Erlbaum Associates.
- Leymann, Frank, Web Services Flow Language, Version 1.0, May 2001. <http://www-306.ibm.com/software/solutions/webservices/pdf/WSFL.pdf>.
- Lohr, Steve, "A Techie, Absolutely, and More", New York Times, August 23, 2005.

- Nagappan, R., R. Skoczylas, R. Sriganesh, Developing Java Web Services, Wiley Publishing, Inc, 2003.
- Nagel, Christian, Enterprise Services with the .NET Framework : Developing Distributed Business Solutions with .NET Enterprise Services, Addison-Wesley, June 2005.
- National Research Council, National Academy of Sciences (1996). National science education standards. Washington, DC: National Academy Press.
- National Council of Teachers of Mathematics (2000). Principles and standards for school mathematics. Reston, VA: Author.
- National Research Council (2002). Scientific Research in Education, Committee on Scientific Principles for Educational Research. Shavelson, R.J. and Towne, L. Editors. Washington, DC: National Academy Press.
- National Research Council, National Academy of Sciences (2003). Assessment in Support of Instruction and Learning. Washington, DC: National Academy Press.
- Pallmann, David, Programming Indigo, the Code Name for the Unified Framework for Building Service-Oriented Applications on the Microsoft Windows Platform, Microsoft Press, July 2005. <http://www.microsoft.com/MSPress/books/7703.asp>
- Paul, R. (2005, February). DoD towards software services. In Proceedings 10th IEEE International Workshop on Object-oriented Real-time Dependable Systems WORDS 05 (pp. 3-6). Los Alamitos, CA: IEEE Computer Society.
- Paul, R., W. T. Tsai and J. Bayne, The Impact of SOA Policy-Based Computing on C2 Interoperation and Computing, 10th International Command and Control Research and Technology Symposium (ICCRTS), McLean, Virginia, June 2005.
- Pelligrino, J., Chudowski, N., & Glaser, R. (2001). Knowing what students know: The science and design of educational assessment. Washington, DC: National Academy Press.
- Brian A. Randell and Rockford Lhotka, Bridge the Gap Between Development and Operations with Whitehorse, Microsoft, 2005. <http://msdn.microsoft.com/msdnmag/issues/04/07/whitehorse/default.aspx>.
- Reid, Betty, and Sarah Anchors, "Urban Schools Not Measuring Up; Mobility, Language Barrier Top Woes", The Arizona Republic, Oct. 19, 2002.
- Reinfelds, J., Programming as An Engineering Discipline, 32nd ASEE/IEEE Frontiers in Education Conference, November 2002.
- Roff, Jason, UML, McGraw Hill, 2003.
- Sastry, S., J. Sztipanovits, R. Bajcsy, H. Gill: Scanning the issue - Special issue on modeling and design of embedded software, Proceedings of the IEEE, Vol. 91, 2003, pp. 3 - 10.
- Shulman, L. (2000). Teacher development: Roles of domain expertise and pedagogical knowledge. Journal of Applied Developmental Psychology, 21(1); 129-135.
- Singh, M. P., M. N. Huhns, Service-Oriented Computing, John Wiley & Sons, The Atrium, 2005.
- Syntelligence, "2005 Trends: Analysts Predict Modest Growth IT Budgets to Hide Some Major Shifts below Surfaces", Vol. 05, No. 1, 2005, <http://www.syntelinc.com/syntelligence/index.aspx?id=545>
- Thastte, Satish, XLANG: Web Services for Business Process Design, Microsoft, 2001. http://www.getdotnet.com/team/xml_wsspecs/xlang-c/default.htm.

- W. T. Tsai, L. Yu, R. Paul, T. Liu, and A. Saimi, "Developing Adaptive Test Frameworks for Testing State-Based Embedded Systems", in Proc. of IDPT, 2002
- Tsai, W. T., Paul, R., Wang, Y., Fan, C., & Wang, D. (2002). Extending WSDL to facilitate web services testing. In Proceedings of the 7th IEEE International Symposium on High-Assurance Systems Engineering HASE 2002. Tokyo, Japan (pp. 171-172). Los Alamitos, CA: IEEE Computer Society.
- Tsai, W. T., Paul, R., Cao, Z., Yu, L., Saimi, A., & Xiao, B., (2003). Verification of web services using an enhanced UDDI server. In Proceedings 10th IEEE International Workshop on Object-oriented Real-time Dependable Systems WORDS(pp. 131-138). Los Alamitos, CA: IEEE Computer Society.
- Tsai, W. T. (2005). Service-oriented system engineering: A new paradigm. In Proceedings of the IEEE International Workshop on Service-Oriented System Engineering (SOSE), Beijing, China (pp. 3-6). Los Alamitos, CA: IEEE Computer Society.
- Tsai, W. T., Paul, R., Xiao, B., Cao, Z., & Chen, Y. (2005, November). PSML-S: A process specification and modeling language for service oriented computing. Paper presented at the 9th IASTED International Conference on Software Engineering and Applications (SEA), Phoenix, AZ, pp. 160-167.
- Tsai, W. T., L. Yu, F. Zhu, and R. Paul, Rapid Embedded System Testing Using Verification Patterns, IEEE Software, Vol. 22, No. 4, July/August 2005, pp. 68 - 75.
- Tsai, W. T., Chen, Y., Paul, R., Huang, H., Zhou, Z., & Wei, X (2005, July). Adaptive testing, Oracle generation, and test script ranking for web services. In Proceedings of the 29th IEEE Annual International Computer Software and Applications Conference COMPSAC, Edinburgh, UK (pp. 101-106). Los Alamitos, CA: IEEE Computer Society.
- Tsai, W. T., Chen, Y., & Paul, R. (2005). Specification-based verification and validation of web services and service-oriented operating systems. In Proceedings of the 10th IEEE International Workshop on Object-oriented Real-time Dependable Systems WORDS 05, Sedona, AZ (pp. 139-147). Los Alamitos, CA: IEEE Computer Society.
- Tsai, W. T., Wei, X., Chen, Y., Xiao, B., Paul, R., & Huang, H. (2005, April). Developing and assuring trustworthy web services. In Proceedings of 7th International Symposium on Autonomous Decentralized Systems (ISADS), Chengdu, China (pp.43-50). Los Alamitos, CA: IEEE Computer Society.
- Tsai, W. T., Liu, X., Chen, Y., & Paul, R. (2005, April). Simulation verification and validation by dynamic policy enforcement. In Proceedings of the 38th Annual Simulation Symposium, San Diego, CA (pp. 91-98). Los Alamitos, CA: IEEE Computer Society.
- Tsai, W. T., C. Fan, Z. Cao, B. Xiao, H. Huang, X. Liu, X. Wei, R. Paul, Y. Chen, and J. Xu, A Scenario-Based Service-Oriented Rapid Multi-Agent Distributed Modeling and Simulation Framework for SoS/SOA and Its Applications, Foundations 04: A Workshop for VV&A in the 21st Century, Tempe, October 2004. <http://www.scs.org/confernc/foundations/foundations04.htm>
- Tsai, W. T., Paul, R., Huang, H., Xiao, B., & Chen, Y. (2005, June). Semantic interoperability and its verification and validation in C2 systems. Paper presented at the 10th International Command and Control Research and Technology Symposium (ICCRTS), McLean, Virginia. Retrieved October 15, 2005, from <http://www.dodccrp.org/events/2005/10th/CD/papers/200.pdf>
- Tsai, Chen, Paul, Service-oriented computing and system engineering, McGraw-Hill, to appear.