Preface

Not since the industrial revolution have people all over the world experienced such dramatic business and lifestyle changes as are now occurring due to information technology (IT) in general and to the Internet specifically. According to the RAND organization (Hundley, 2004):

Advances in information technology are affecting most segments of business, society, and governments today in many if not most regions of the world. The changes that IT is bringing about in various aspects of life are often collectively called the “information revolution.”

THE INFORMATION AND INTERNET REVOLUTION

The current IT revolution is not the first of its kind. Historians and nations may debate the exact time and place of previous information revolutions, but they were:

- The invention of writing, first in Mesopotamia or China about 3000B.C.
- The invention of the written book in China or Greece about 1000B.C.
- Gutenberg’s printing press and engraving about 1450A.D.

One is reminded of the opening sentence from *A Tale of Two Cities* by Charles Dickens: “It was the best of times, it was the worst of times.” Dickens was referring to the French Revolution, but today in the 21st century we are well into the “IT Revolution.”

All major revolutions help some people and organizations, and these may think it is the “best of times” but they also hurt some, and these organizations and people think it to be the “worst of times.” With big revolutions there always will be big winners and big losers. As an example, when the printing press was invented, the largest occupation in Europe was the hand copying of books in thousands of monasteries each of which was home to hundreds of monks; fifty years later the monks had been completely displaced. The impact to society was enormous, not because of the displacement of monks by other craftsmen and machines, but because the price of books dropped so drastically that common people could now afford to educate themselves.

For many this new IT revolution is bringing great things with unprecedented improvements in the quality and efficiency of all we do as organizations and as individuals. However for others, IT is a double-edged sword bringing about many problems, disturbances, and unresolved issues such as the creation of a “Digital Divide” between the IT the “haves” and the “have-nots.” In addition IT security and privacy problems are getting out of control as evidenced by computer viruses, worms, e-mail fraud and spam, the compromise of personal and private digital information, spyware, piracy of intellectual property,
identity theft, IP hijacking, hacking, and other computer crimes. Today, there are major and numerous security “holes” in most software that corporations and individuals use on an everyday basis.

The single most important technology of this information revolution has to be the Internet, which is the combination of several underlying technologies. Consider the penetration rate (in the time to reach 50 million users) of milestone information technologies of the recent past compared to the Internet:

- It took the telephone 40 years to reach 50 million users.
- It took radio 38 years to reach 50 million users.
- It took cable TV 10 years to reach 50 million users.
- It only took the Internet only 5 years to reach 50 million users!

It seems that most of the really cool and innovative IT products and projects today involve Web applications, one way or another; such products and technologies as Amazon, Google, YouTube, MapQuest, eBay, Priceline, and iPhone immediately come to mind. However, as well as these glamorous innovations, many more everyday business applications are migrating to the Web. This is true for both intra-company applications and extra-company applications.

But the Internet and related technologies are beginning to cause significant industrial disruptions including:

- Internet shopping is disrupting traditional sales channels for hard goods;
- Internet sharing and distribution is disrupting traditional intellectual property rights and sales of soft goods (print, audio, video, multi-media);
- “Voice Over IP” combined with ultra high speed optical and wireless media will start to disrupt traditional telecommunications;
- Open source software with community on-line support will start to disrupt the traditional software marketplace;
- Separation of “work” from “workplace” will disrupt corporate and personal real estate and related business sectors;
- As national barriers (political, physical, economic, and temporal) are removed, massive “globalization” will allow the free flow of both work and product;
- The need for retraining and “lifetime learning” coupled with “distance education” is transforming the traditional higher education landscape.

The process and results of these disruptions has been called “creative destruction” by the RAND Corporation, and this results in the “economic eclipse” of organizations not embracing the new IT world. Traditional mechanisms of government (i.e., jurisdiction, taxation, regulation, permits and licenses, etc.) will also be subject to significant disruption in response to these other changes, as will the insurance and finance industries.

**BETTER CHEAPER FASTER**

Today’s commerce landscape is defined by fierce global competition, thus the “battle cry” of the modern business world is:
Better!
Cheaper!
Faster!

That battle cry starts in the board room then down through the management chain as these themes are the crux of market positioning (quality, cost, and time to market) as illustrated in Figure I-1. As Tom Cruise said in the movie *Top Gun*: “I feel a need, a need for speed.” That “need for speed” has been emphasized to IT project managers and software development teams by upper management.

To produce better and/or cheaper products or services and get them to market quicker requires better, cheaper, faster processes as illustrated in Figure I-2. In today’s world, information systems play a key role and an ever increasing role in the overall process of producing/delivering products or providing services. Today almost every aspect in the design, creation, delivery, and support of products or services today is highly dependent on IT.

The IT platform of choice for implementing business applications has evolved over the last 50 years mainly by mutations rather than by the continuous change of natural selection. This is illustrated in Figure I-3. Today’s Internet and Web applications specifically offer business a way to tap into a global market very quickly. Both large and small organizations can easily, cheaply, and quickly put forth a global presence and/or storefront via today’s Internet.

The initial simplicity of Web-based HTTP/HTML applications provided a rapid proliferation of simple internet applications. In those early days of the Internet, Web-user interface designers may have felt a little envious of their colleagues who created desktop (PC) client-server software. PC and client-server desktop applications had a richness and responsiveness that initially seemed out of reach for early

---

**Figure I-1. Marketing dimensions**

![Marketing Dimensions Diagram](image)

**Figure I-2. Products and processes**

![Products and Processes Diagram](image)
HTML Web software. However, today software technology stacks such as DHTML and Ajax now allow rich and responsive Web applications to be built which increase the depth and breadth of possible Web application penetration and proliferation.

Web-based applications have continued to evolve as more and newer technologies become available. We are now into the “Web 2” era with these rich and dynamic Web pages, and we are beginning to experiment with “Web 3” technologies and methodologies to create a hyperspace not only of documents, but also of multi-dimensional database objects.

WEB SOFTWARE ENGINEERING

*From a software engineering perspective, and also from a project management perspective, developing Web-based applications is different from developing traditional applications based on mainframe*

*Figure I-3. Application platform evolution*

- Better
- Automation
- Mainframe
- Cheaper
- Distributed Processing
- Client Server
- Faster
- Universal Access
- Web Based

*Figure I-4. Web evolution*

199x - 2003  
WWW  
HTTP, HTML, JavaScript, DHTML, CSS

2004 - 200x  
Web 2  
XML, SOAP, SOA, Ajax

200x - 201x  
Web 3  
Semantic Web
and/or client-server technologies. Some of these differences arise due to changes in the technologies involved, some differences are due to the business climate today, and some differences are due to the scope and nature of the Web applications themselves.

The pace of business life is much quicker today than in the past, stimulated in a large extent due to global competition and the rate of new technology advances. Thus Web applications need to be deployed much quicker than earlier IT applications. Also due to the rapid business pace today, Web applications need to be engineered for more flexibility and adaptability in terms of changing requirements, changing content, and changing presentation. Business growth can be rapid for some products and services, thus Web applications need to be more scalable to accommodate large and rapid growth in terms of number of users and number of “hits.” Many Web applications are also for products that feature the concept of “mass customization” which often translates into user customization of the Web user interface as well as the product and/or service being delivered.

Web applications can serve a global audience and thus there are more diverse stakeholders for these applications. Issues such as language, culture, time zones, weights and measures, currency, logistics, and so forth, need to be considered. Norms and even laws may be different where Web applications are defined as opposed to where Web applications are utilized. Also because of the larger and more diverse stakeholder sets, Web applications are subject to much more external scrutiny.

Web applications are physically or virtually delivered over a network rather than installed as was the case with mainframe or desktop applications. Thus network considerations such as responsiveness, throughput, and security are much greater.

Older mainframe, client-server, and/or desktop applications generally involved a single programming language (and toolset), at least for each distinct application/program. Mainframe applications were typically written in COBOL, RPG, or FORTRAN and client-server/desktop applications were written in C, C++, or Visual Basic. However, producing Web applications generally involves multiple languages and tool sets including HTML, JavaScript, and CSS on the browser (client) side, and Java, PHP, ASP, or Perl on the server side. New technologies may also be intermixed on both sides including XML, SOAP, and Ajax. For business applications, a common denominator between older applications and Web applications has been the use of SQL for database queries and manipulation.

Web applications generally manage content and not just data; further, many Web applications are document centric versus data centric. There are many more people involved in the definition, design, development, testing, and approval for Web applications as opposed to older IT applications, both technical types and business types. This is illustrated in Figure I-5, which shows that many Web applications have not only a software development lifecycle, but also a content management lifecycle.

This book addresses many of the above differences between Web software engineering and traditional software engineering platforms, and introduces and discusses methodologies and technologies for successful application development in the Web environment:

Chapter I: Web Engineering: Introduction and Perspectives. Web-based systems and applications now deliver a complex array of functionality to a large number of diverse groups of users. As our dependence and reliance on the Web has increased dramatically over the years, their performance, reliability and quality have become paramount importance. As a result, the development of Web applications has become more complex and challenging than most of us think. In many ways, it is also different and more complex than traditional software development. But, currently, the development and maintenance of most Web applications is chaotic and far from satisfactory. To successfully build and maintain large, complex Web-based systems and applications, Web developers need to adopt a disciplined development process and a sound methodology. The emerging discipline of Web engineering advocates a holistic,
disciplined approach to successful Web development. In this chapter, we articulate and raise awareness of the issues and considerations in large, complex Web application development, and introduce Web engineering as a way of managing complexity and diversity of large-scale Web development.

Chapter II: Augmented WebHelix: A Practical Process for Web Engineering. Process is an important element in the success of any information systems development project especially in academia where typically an undergraduate term project needs to go through the development phases within the space of a semester. Traditionally academic processes have been adapted versions of well-known industrial processes with one major exception—lack of customer feedback in the process. This omission of customer feedback results in students completing “toy” projects without significant real-world experience; efforts to incorporate artificial customer interactions have not been very successful either. It is our opinion that the industry processes cannot be simply copied in academia; what is required is a process that will better equip the students to face real-world challenges. WebHelix has been recently introduced as a practical process for Web engineering that helps students gain valuable real-world experience without sacrificing project and product management phases. In this chapter, we propose the Augmented WebHelix process that augments the WebHelix in three ways: provides an option at the end of each slice of the helix to both release the current version and continue to the next slice of development, and provides a qualitative evaluation framework, called the project evaluation framework (PEF); that provides a systematic approach for evaluating the status of the project; and the ability to evaluate the project at the end of each phase in a slice of the helix. The first augmentation provides the ability to release and continue which is more practical than the go/no-go approach adopted by WebHelix; the second augmentation, the PEF, allows different factors besides the return-on-investment as in WebHelix to be considered for evaluating the current phase and status of the project, and the third augmentation provides the ability to ensure the project is on track. In this chapter, we describe the augmented WebHelix process and demonstrate its applicability to both academia and industry with examples.

Chapter III: Model-Centric Architecting Process. This chapter defines a methodical approach, named model-centric architecting process (MAP), to effectively cope with the architecture design
complexity and manage the architecting process and lifecycle of information systems development in a service-oriented paradigm. This comprehensive method comprises four dimensions of architecting activities: requirement analysis, specification, validation, and planning (RSVP). The process is broken down to 9 interrelated models: meta architecture, conceptual architecture, logical architecture, physical architecture, deployment architecture, management architecture, information architecture, aspect architecture, and component architecture. A 2-D matrix serves as a blueprint to denote a step-by-step procedure to produce and manage the architectural artifacts and deliverables in the lifecycle of systems architecture design, development and governance. The holistic framework provides a multidisciplinary view of the design principles, tenets, idioms, and strategies in the IT architecting practices. The characteristics and features of the constituent elements in the MAP approach are articulated in great detail. Recommendations and future trends are also presented in the context. It helps build high-quality service-oriented solutions focused on different domains, and in the meantime keeps the agility, flexibility and adaptiveness of the overall method. This systematic framework may be customized in different formats to design various information systems in different industries.

Chapter IV: Architecture, Specification, and Design of Service-Oriented Systems. Service-oriented software systems (SOSS) are becoming the leading paradigm of software engineering. The crucial elements of the requirements specification of SOSSs are discussed as well as the relation between the requirements specification and the architecture of SOSS. It is preferable to understand service orientation not to be limited to Web services and Internet only. It is shown that there are several variants of SOSS having different application domains, different user properties, different development processes, and different software engineering properties. The conditions implying advantageous user properties of SOSS are presented. The conditions are user-oriented interfaces of services, the application of peer-to-peer philosophy, and the combination of different technologies of communication between services (seemingly the obsolete ones inclusive), and autonomy of the services. These conditions imply excellent software engineering properties of SOSSs as well. Service orientation promises to open the way to the software as a proper engineering product.

Chapter V: Data Integration Through Service-Based Mediation for Web-Enabled Information Systems. The Web and its underlying platform technologies have often been used to integrate existing software and information systems. Traditional techniques for data representation and transformations between documents are not sufficient to support a flexible and maintainable data integration solution that meets the requirements of modern complex Web-enabled software and information systems. The difficulty arises from the high degree of complexity of data structures, for example in business and technology applications, and from the constant change of data and its representation. In the Web context, where the Web platform is used to integrate different organisations or software systems, additionally the problem of heterogeneity arises. We introduce a specific data integration solution for Web applications such as Web-enabled information systems. Our contribution is an integration technology framework for Web-enabled information systems comprising, firstly, a data integration technique based on the declarative specification of transformation rules and the construction of connectors that handle the integration and, secondly, a mediator architecture based on information services and the constructed connectors to handle the integration process.

Chapter VI: Using Patterns for Engineering High-Quality Web Applications. In this chapter, we view the development and maintenance of Web applications from an engineering perspective. A methodology, termed as POWEM, for deploying patterns as means for improving the quality of Web applications is presented. To that end, relevant quality attributes and corresponding stakeholder types are identified. The role of a process, the challenges in making optimal use of patterns, and feasibility issues involved in doing so, are analyzed. The activities of a systematic selection and application of patterns are explored. Following a top-down approach to design, examples illustrating the use of patterns during
macro- and micro-architecture design of a Web application are given. Finally, the implications towards Semantic Web applications and Web 2.0 applications are briefly outlined.

Chapter VII: Component-Based Deployment for Web Applications. The software engineering community touts component-based software systems as a potential silver bullet for many of its woes: reducing cycle time; reducing cost; increasing productivity; allowing easier integration - to name just a few. Indeed, many Web-based systems are now built with open-source and vendor provided component technologies. While these advances have led to improvements in the development process, they have also led to a great deal of pressure on downstream processes as these systems must be deployed, tuned, and supported. The complexities in deploying and supporting component-based software for distributed and Web-based applications are not understood in the academic or professional communities. This chapter stresses the need for addressing this problem by presenting component-based software for Web applications from a deployment perspective, characterizing the issues through real-world experiences with highly component-based applications, and presents strategies and directions for the community to pursue.

Chapter VIII: Evolving Web Application Architectures, from Model 2 to Web 2. This chapter explores how Web application software architecture has evolved from the simple beginnings of static content, through dynamic content, to adaptive content and the integrated client-server technologies of the Web 2.0. It reviews how various technologies and standards have developed in a repeating cycle of innovation, which tends to fragment the Web environment, followed by standardization, which enables the wider reach of new technologies. It examines the impact of the Web 2.0, XML, Ajax and mobile Web clients on Web application architectures, and how server side processes can support increasingly rich, diverse and interactive clients. It provides an overview of a server-side Java-based architecture for contemporary Web applications that demonstrates some of the key concepts under discussion. By outlining the various forces that influence architectural decisions, this chapter should help developers to take advantage of the potential of innovative technologies without sacrificing the broad reach of standards based development.

Chapter IX: Applying Agility to Database Design. Agile methods are flexible, allowing software developers to embrace changes during the software development life cycle. But the introduction of agile practices into software organizations may cause unhealthy tensions between the developers and data professionals. The underlying reason is that when agile methodologies are employed, the two communities use incompatible approaches, namely simple design and iterative development, which are practices associated with all agile methodologies, and big design up front (BDUF), a popular database technique. BDUF is inflexible, as once the database foundation is set, it is difficult to make changes throughout the software development life cycle. This chapter describes a database development method for a Web environment. Using this method, a data professional divides the database into loosely coupled partitions and resolves the above conflicts by applying certain agile practices. The result is that the database development becomes more iterative and incremental. This has the added benefit of supporting rapid application development in a dynamic environment, a fundamental characteristic of most Web applications.

Chapter X: Automatic Creation of GUI’s for Web Based Systems. Service-oriented architecture (SOA) is an emerging architectural style for developing and structuring business applications, especially enterprise resource planning (ERP) systems. SOA applications are composed of small, independent and network-accessible software components, named services. The service composition is normally based on the enterprise’s business processes. However, current composition standards like BPEL have no ability to interact with users. Therefore, we propose a mechanism for including user interaction descriptions into the composition and extending the composition platform for generating user interfaces. In our case study, a federated ERP (FERP) system, this mechanism has been implemented in a prototype based on yet
another workflow language (YAWL) dynamically generating Web pages for accessing the ERP system. Because every aspect including the user interfaces can be manipulated through the service composition, such systems are highly flexible yet maintainable.

Chapter XI: Prototyping in Web Development. This chapter addresses issues, alternatives, and best practices for prototyping in Web development. The chapter’s primary objective is to provide a clear and concise overview of key concepts and best practices for practitioners and students, as well as other audiences. The chapter focuses on graphical user interface (UI) prototyping for Web development, but many of the principles apply to non-UI prototyping and other sorts of software development. First, we introduce the chapter, and review the major objectives, benefits and risks, and classifications of prototypes. Second, we describe the major approaches to prototyping. Finally, we conclude with future trends and a summary of best practices.

Chapter XII: Testing Methods for Web Applications. This chapter introduces several modern methods and tools for creating tests and integrating testing in Web applications, as well as presenting some practices that expand the role of testing in software development. Teams adopting practices such as test-driven development (TDD) and acceptance (Customer) testing have shown significant gains in the quality and correctness of the code produced. These practices encourage a more holistic view of testing by integrating a sound testing solution into current software development life cycle models. Furthermore, tests can play an important role in gathering requirements and providing documentation. In this chapter, in addition to offering an initiation to some of the modern testing methods and tools, the authors hope to motivate readers to consider testing as a multi-purpose tool to be used throughout all stages of development.

Chapter XIII: Outsourcing Issues in Web Development. This chapter addresses issues, alternatives, and best practices that apply when outsourcing Web development. The chapter’s primary objective is to provide a concise overview of key concepts and best practices for practitioners and students, as well as other audiences. First, we introduce the chapter, provide background, and present three key ideas that are expanded and developed in the two subsequent sections. The first describes four steps to help executives and upper management address strategic issues and decisions in outsourcing. The second describes four more steps to help managers, team leaders, and development teams address more tactical issues. We conclude with future trends and implications, and a summary of the best practices.

Chapter XIV: Engineering Wireless Mobile Applications. Conventional desktop software applications are usually designed, built, and tested on a platform similar to the one on which they will be deployed and run. Wireless mobile application development, on the other hand, is more challenging because applications are developed on one platform (like UNIX or Windows) and deployed on a totally different platform like a cellular phone. While wireless applications can be much smaller than conventional desktop applications, developers should think in small terms of the devices on which the applications will run and the environment in which they will operate instead of the amount of code to be written. This chapter presents a systematic approach to engineering wireless application and offers practical guidelines for testing them. What is unique about this approach is that it takes into account the special features of the new medium (mobile devices and wireless networks), the operational environment, and the multiplicity of user backgrounds; all of which pose new challenges to wireless application development.

Chapter XV: Project Management and Web Software Engineering. The process involved with the development of Web applications is significantly different from the process of developing applications on older platforms. This is a difference not only in technologies but in the overall business process and associated methodology, in other words the project management. Web applications generally manage content and not just data, and many Web applications are document centric versus data centric. In addition,
there are many more people involved in the definition, design, development, testing, and approval for Web applications. The pace of business life is much quicker today than in the past, thus Web applications need to be deployed much quicker than earlier IT application and they need to be engineered for more flexibility and adaptability in terms of changing requirements, changing content, changing presentation, mass user customization. In addition, security concerns are more prevalent in Web applications since the end users are outside as well as inside the corporate virtual perimeter. Web applications can serve a global audience and thus there are more diverse stakeholders for these applications. Issues such as language, culture, time zones, weights and measures, currency, logistics, and so forth, need to be considered. This chapter re-examines the project management issues involved with Web applications.

**Chapter XVI: Resources on Web-Centric Computing.** This chapter provides an extensive listing of resources for Web software engineering as well as an author sponsored link to more detailed and more current resources as they become available

**REFERENCE**