Web engineering–: An introduction

Article *in* IEEE Multimedia · February 2001

DOI: 10.1109/93.923949 · Source: IEEE Xplore

2 authors:

Athula Ginige
Western Sydney University

San Murugesan
BRITE Professional Services

CUTTER IT JOURNAL

Web Engineering:

A Methodology for Developing Scalable, Maintainable Web Applications

by Athula Ginige and San Murugesan

Although the development of Web applications may seem easy, it is often more complex and challenging than many of us think. In many ways, it is also different and more complex than traditional software development [6]. Two key attributes distinguish Web-based systems development from traditional software development: rapid growth of the requirements of Web-based systems and the continual change of their information content. Web-based systems need to be designed and built for scalability and maintainability; these features can't be added later. Success in building, implementing, and maintaining a Web-based system largely depends on how well we address these issues.

In addition, a Web-based system must meet the needs of its many different stakeholders — the diverse range of the system's users, persons who maintain the system, the organization that needs the system, and also those who fund the system development. These needs add to the complexity of Web-based system design and development. Furthermore, development of Web-based systems calls for people with knowledge and expertise in many different areas.

Many organizations and developers have successfully developed large, high-performance Web sites and applications, but others have failed or face the potential for major failures. A recent survey on Web-based application development by the Cutter Consortium highlighted the problems plaguing large Web-based projects [4]:

- Delivered systems didn't meet business needs 84% of the time.
- Delivered systems didn't have the required functionality 53% of the time.
- Schedule delays plagued the projects 79% of the time.
- Projects exceeded the budget 63% of the time.

The primary causes of Web system failures are a flawed design and development process and poor management of their development [6]. The way we address these concerns is critical to realizing the Web's full potential.

The emerging Web engineering discipline deals with the process of developing Web-based systems and applications. The essence of Web engineering is to successfully manage the diversity and complexity of Web application development and hence avoid potential failures, which can have serious implications. It is a proactive approach to building Web applications. For a brief introduction to Web engineering and a review of progress in this field, see [6, 7, 9, 11, 13].

Based on our experience in building Web-based systems and on our research, we present a methodology for successful and sustainable development of Web applications. We believe the Web development methodology and the guidelines we offer can help Web developers and project managers avoid many of the problems currently hampering Web-based system development.

EVOLUTIONARY WEB APPLICATION DEVELOPMENT

Web applications are evolutionary. For many Web applications, it's not possible to specify fully what they should or will contain at the start of their development, because their structure and functionality will evolve over time. Hence, Web-based system development isn't a one-time event, as currently perceived and practiced by many Web developers; it is, instead, an iterative process with a long lifecycle.

As we now place greater emphasis on the performance, correctness, and availability of Web-based systems, the development process assumes greater significance. A sound process is a prerequisite for success, especially when a distributed team of people with different types and levels of skills and expertise work together to develop large Web applications.

Web development, therefore, should be considered as a process consisting of many phases, steps, and activities. The Web development process breaks the Web development efforts into manageable chunks and offers techniques to help developers successfully manage and complete Web projects. A sound process helps developers address the complexities of Web-based systems, minimizes the development risks, deals with the likelihood of change, delivers the Web applications on time, and provides feedback for management on project progress.

To be effective, the development process should be planned well and clearly define a set of steps that developers can follow. Furthermore, it should be measurable and trackable. The process should also facilitate the Web application's continual refinement and evolution based on feedback from users and clients. Such processes and methodologies have been applied in a number of successful Web applications, including those of the ABC Internet College, University of Western Sydney, 2000 Sydney Olympics, 1998 Nagano Winter Olympics, and Vienna International Festival.

Steps to Successful Development

Building and deploying a Web-based system involves multiple steps. These steps influence one another and are iterative. We recommend the following 10 key steps for successful development and deployment of Web applications [7]:

1. Understand the system's overall function and operational environment, including the business objectives and requirements.

2. Clearly identify the stakeholders — that is, the system's main users, the organization that needs the system, and who funds the development.

3. Specify the (initial) functional, technical, and nontechnical requirements of the stakeholders and the overall system. Furthermore, recognize that they may not remain the same; rather, they are bound to change and evolve over time during system development and also at each iteration/revision.

4. Develop an overall architecture of the Web-based system that meets the technical and nontechnical requirements.

5. Identify subprojects or subprocesses to implement the architecture. If the subprojects are too complex to manage, further divide them until they become a set of manageable tasks.

6. Develop and implement the subprojects.

7. Incorporate effective mechanisms to manage the Web system's evolution, change, and maintenance. As the system evolves, repeat the overall process, or some parts of it, as required.

As we now place greater emphasis on the performance, correctness, and availability of Web-based systems, the development process assumes greater significance.

8. Address the nontechnical issues, such as revised business processes, organizational and management policies, human resources development, and legal, cultural, and social aspects.

9. Measure the system's performance.

10. Refine and update the system.

WEB DEVELOPMENT PROCESS

Based on our practical experience in building successful Web applications, we recommend the development process shown in Figure 1. This process assists in capturing the requirements, enables integration of know-how from different disciplines, facilitates communication among various members involved in the development process, supports continuous evolution and maintenance, facilitates management of the information content, and assists in successfully managing the complexity of the development process.

Context Analysis

The first essential step in developing a Web-based system is context analysis. In this step, we elicit and understand the system's major objectives and requirements, gather information about the operational and application environment, and identify the system's primary stakeholders. In addition to the functional requirements, the system's requirements for scalability, maintainability, availability, and performance need to be specifically elicited and understood by the developers at the beginning of the development process. Based on this information, we then arrive at the system's technical and nontechnical requirements, which, in turn, influence the system design.

For instance, if the information content and the system's functions are going to evolve considerably, as in e-business systems, the system needs to be designed for scalability. On the other hand, if the information changes frequently — as in online

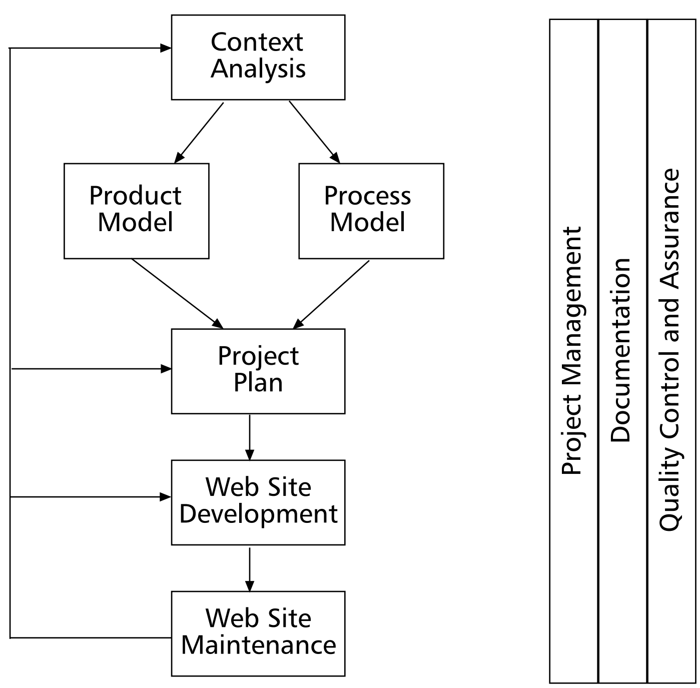

Figure 1 — Development process for Web-based systems.

electronic product catalogs, product price lists, brochures, and daily news columns — the system must be designed for easy information maintainability in order to keep the information current and consistent. Where an application demands very high availability and has to allow for very high peak or uncertain demands, the system may need to run on multiple Web servers with load balancing and other performance enhancement mechanisms [2, 8]. Examples of this category of application are online stock trading, online banking, and highly accessed sports and entertainment Web sites, such as those for the Olympics, Wimbledon, and the Oscars.

Thus, it is very important to recognize that scalability, maintainability, and/or performance need to be built into the initial system architecture. It would be very hard, or even impossible, to add these features if the initial architecture is not designed to support them. To illustrate this, consider an e-business Web site that provides product information, such as color, price, and availability, that appears on many different pages and may change frequently. If the Web site is designed as static HTML pages, then every time a product's information changes, someone has to incorporate the change in every page that contains this information. This is a cumbersome and laborious task, and often changes may be made only on a few pages, not on all of them. As a consequence of this, information appearing on different pages will be inconsistent.

A better approach to ensure consistency of information across all Web pages is to automatically retrieve the information, when and where needed, from a single information source. If product information is stored in a single central database, then by extracting the relevant information from this database, we can dynamically create various Web pages that contain this information. In the database-driven approach, we need to change the information only in one place — the database. Further, the database-driven Web sites can have a back-end system that allows an authorized person, who need not be skilled in Web page development, to make information changes easily through a Web interface, from anywhere. Such a Web site requires a completely different architecture than a Web site that has only static HTML pages. Hence, an appropriate architecture needs to be chosen early in system development.

Thus, the context analysis assists us in capturing and deriving key information required to develop a Web application. In addition, it can also identify nontechnical issues and needs that have to be addressed for successful implementation and application of the system. These may include reengineering of business processes where required, organizational and management policies, staff training, and legal, cultural and social issues.

Context analysis can minimize or even eliminate the major problems plaguing large Web-based system development, such as delivered systems that don't meet business needs and don't have the required functionality. But many developers and project managers overlook this essential first step in Web system development and hence face the problems later, when it is hard to correct them.

Context analysis can minimize or even eliminate the major problems plaguing large Web-based system development.

Product Model

A product model of the system describes various components of the system and how they are linked. The following three types of architectures constitute the product model:

1. An overall *system architecture* describing how the network and the various servers such as Web servers, application servers, and database servers interact

2. An *application architecture* depicting various information modules and the functions available
3. A *software architecture* identifying various software and database modules required to implement the application architecture

A suitable product model for an application is developed based on the system's requirements derived from the context analysis.

System Architecture

Development of an appropriate system architecture is important, especially if high performance and/or high security of the Web-based system are critical needs. For applications such as those used for the 1998 Nagano Winter Olympic Games and the 2000 Sydney Olympic Games, high performance and high availability were primary requirements. Hence, design of these systems' network and server architecture incorporated redundant hardware and advanced features such as load balancing, Web server acceleration, and efficient management of dynamic data to boost the performance [8]. Where performance and security are not critical requirements, a standard Web server is adequate.

Application Architecture

The application architecture shows us a map of various information and functional modules. An information module may provide the same information to all the users or provide customized or personalized information to each user. Functional modules — such as login pages, registration pages, Web forms for data collection, and the shopping carts used in e-commerce systems — collect and process the users' input.

As an illustration, consider the application architecture of the ABC Internet College (www.abccollege.com), shown in Figure 2, which provides students with online personalized tutoring and dynamically generates learning materials based on each student's past performance. When a student logs in, learning activities appropriate for that day are presented on the student's personalized home page. Based on his or her past performance, the student is directed to do the next test module, a personalized revision

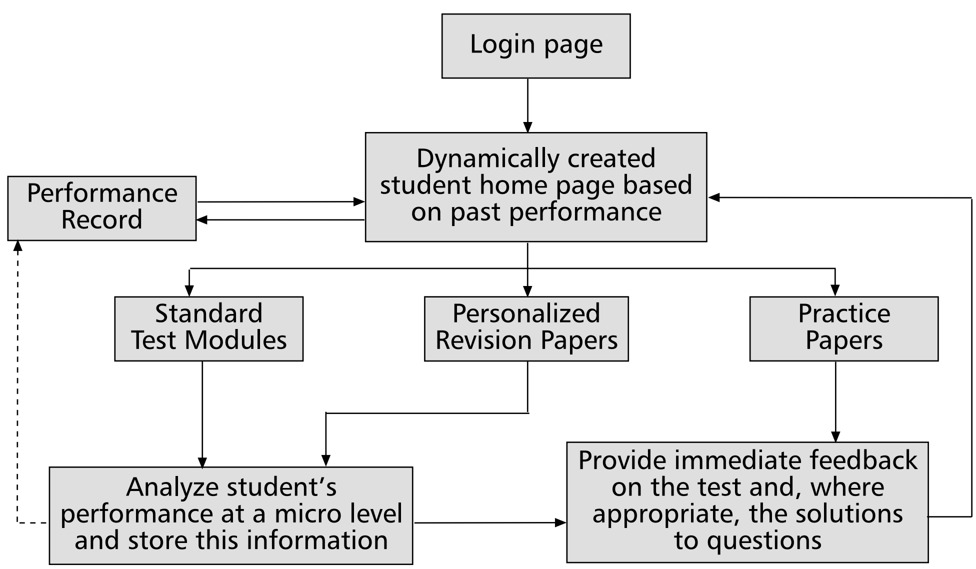

Figure 2 — Application architecture of the ABC Internet College Web application.

paper, or practice questions in areas in which the student did not perform well. This online system has been operational for the last three years, and the way it was designed and structured has made upgrades and enhancements easy.

Software Architecture

The application architecture is then mapped into a software architecture, shown in Figure 3, displaying various software and database modules required to provide the required functionality. Specific requirements for look and feel, consistency of information, scalability, maintainability, and quality control mechanisms determine the software architecture. Table 1 highlights the means of fulfilling these requirements.

The product model also helps us to decide on an appropriate process model for development and to estimate development time and cost.

Process Model

A process model specifies the activities that need to be carried out to develop and implement the system. The activities include detailed analysis of requirements, design, testing, and deployment. To better manage a large project, each major activity may, however, be divided into a set of subprojects. In addition, the process model should also include activities that address the nontechnical issues identified in the context analysis.

A Web-based application consists of: (1) information content, and (2) software required to deliver the content, to assist in maintenance and quality assurance of the content, and to provide various interactive capabilities. Hence, we need a process to develop the information structure, information content, screen layout, and navigation mechanism. We also require a process to develop the required software.

In a large Web application development project, the content is generally developed by people with journalism, library science, marketing, or public relations backgrounds. Computer graphic designers and visual art specialists develop page layouts, and

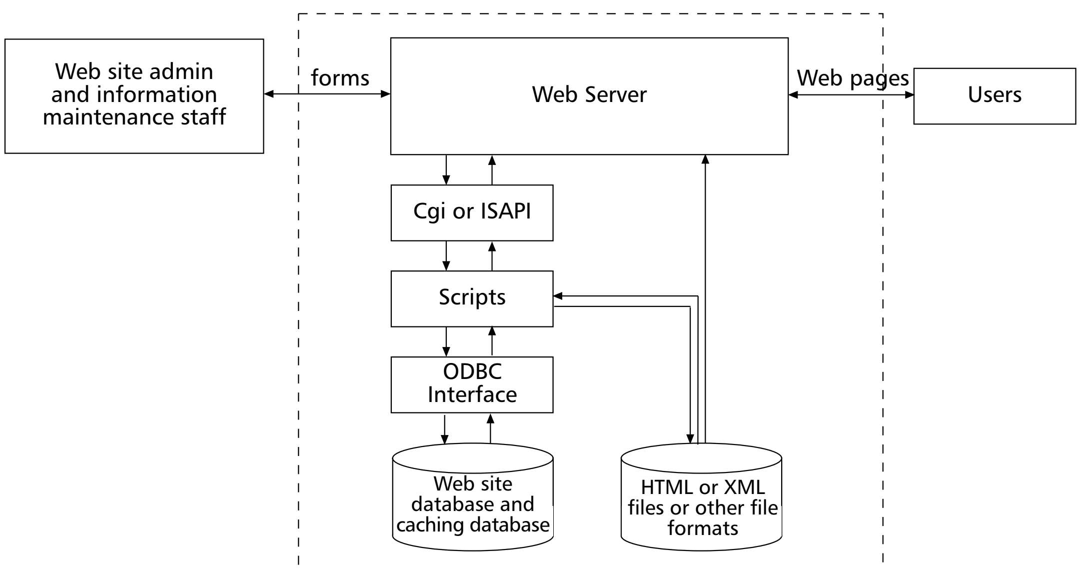

Figure 3 — Software architecture.

Table 1 — Means of Fulfilling Web System Requirements

Requirement	Means of fulfillment
Uniform look and feel across all the Web pages that can easily be modified	Creation of Web pages using templates
Consistency of information that may appear in different places or pages	Storing information in a single place (in a database or as an XML file) — without duplication of information in different places or databases — and retrieving the required information for presentation where and when needed
Ease of information update and maintenance	Provision of a back-end system to edit information in a data repository; could have Web interface for easy access from anywhere
Ability to add new Web pages easily	Dynamic generation of navigational links, rather than predetermined static navigational links
Decentralized system administration	Provision of a multiuser login system to access back-end systems and use of a "user administration system" that can assign specific functions and data sets to content managers and other developers/ administrators
Mechanisms for quality control and assessing the relevance of information	Inclusion of metadata for Web pages; use of a Web robot for gathering salient information, processing the information gathered, and taking appropriate action(s) for ensuring quality or relevance of information presented

computing, software, and IT professionals develop the software, database, and information systems. It is important to ensure that the development processes enable these three diverse groups to work together and communicate effectively.

Many software development process models [11] can be suitably adapted for Web development. Considering the strengths and limitations of the various models and the requirements of the application, we can choose and use an appropriate process model. For example, for small-scale trial applications, a model incorporating iterative refinement of an initial prototype is most suitable.

Project Plan

To successfully manage Web development, a sound project plan and a realistic schedule are necessary. Progress of development activities must be monitored and managed. Project planning and scheduling techniques used in other disciplines can be used for Web development.

Web Site Maintenance

After a Web-based system is developed and deployed, it needs to be continually maintained. Web maintenance can be classified into three main categories: content maintenance, software maintenance, and hardware and network maintenance.

Content maintenance is a continual process. We need to formulate content maintenance procedures, based on the decision taken at the context analysis stage on how the information content would be maintained, and then we need to implement them. The hardware and the network also need to be periodically maintained, and failures need to be fixed when they surface. It is also important to periodically review information and Web security risks and take suitable measures to fix those weaknesses.

Project Management, Documentation, and Quality Assurance

Poor project management will defeat good engineering. The purpose of project management is to ensure that all the key processes and activities work in harmony. Building successful Web-based applications requires close coordination among the various efforts involved in the Web development cycle.

Activities such as project management, quality assurance, and documentation spread throughout the Web development lifecycle. Software engineering methodologies and techniques used for these activities can also be used with suitable modification for Web development.

WEB SITE DEVELOPMENT

Web site development consists of development of Web page content and application software to deliver the content and functionality. In addition, computer hardware and the network infrastructure also need to be developed and deployed. The performance, availability (up-time), and security required for the application govern the design of the network and computer hardware. Arun Iyengar, et al. [8] and Valeria Cardellini, et al. [2] discuss the design of computer hardware and network infrastructure to support high-performance Web sites.

To better manage Web site development, we developed and refined a two-stage approach, shown in Figure 4, and have used it successfully in our projects. It decouples the content development from application software development, and this decoupling is necessary, as content developers and software developers come from different

Poor project management will defeat good engineering.

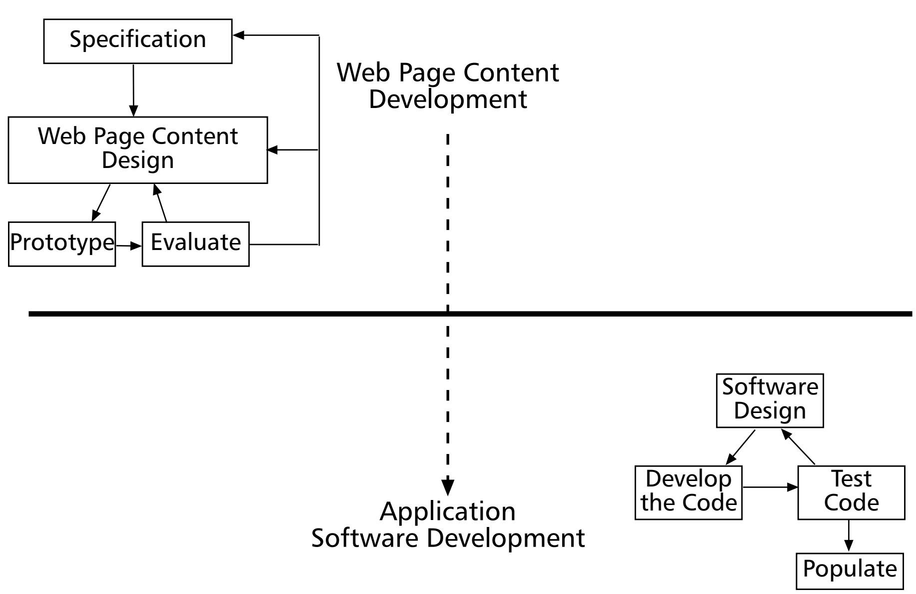

Figure 4 — Two-stage approach to Web site development.

backgrounds and have different skills and expertise. The outcome of the Web page content development process, described below, forms the input for the application software development process.

Web Page Content Development

Web page content development is an important activity; it determines what information is presented and how it is presented to the system's users. Content development starts with a detailed analysis of the Web-based system requirements and moves on to development of appropriate specifications for Web pages. Web pages are then designed to meet the specifications. The prototype usually contains a set of sample pages to evaluate the page layout and navigation among different pages. Based on the feedback from the stakeholders, the page design and/or the specifications may be suitably modified. This process may go through a few iterations until the stakeholders and designers are satisfied with the screen layout and the navigation structure.

To meet the needs of global users, a Web site's usability may have to be examined from a global perspective.

Design is the creative part of Web page content development, and it needs to take into consideration the stakeholders' requirements, users' cognitive abilities, technical issues, nontechnical issues, and earlier experiences, as shown in Figure 5.

Knowledge of users' cognitive skills and issues is useful for arriving at a good design [12]. The designer should know how the users would perceive and comprehend information, as well as how the fonts, color, and layout contribute to enhancing the users' comprehension. This is very important, as the usability of a Web site will determine its success. Furthermore, to meet the needs of global users, a Web site's usability may have to be examined from a global perspective. Shirley Becker and Florence Mottay [1] argue that usability requires cultural sensitivity in language translation along with appropriate use of color, design, and animation. Jacob Nielsen discusses how to design usable Web sites [10], and Molly Hammar Cloyd describes how to design user-centered Web applications [3].

In designing a Web-based system, it is important that the designers be aware of and take into consideration the technological constraints and their impact on the Web application. For example, it may not be feasible to use very large graphics or long video clips due to bandwidth limitations. The designer should also consider nontechnical aspects such as legal, moral, and social issues that are relevant to the environment in which the application will be used.

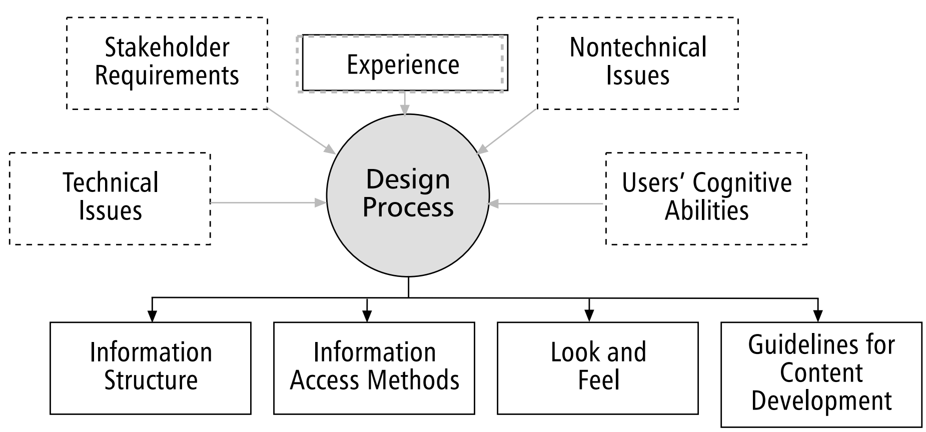

Figure 5 — Web page content design process.

The outcomes of the design process include an appropriate information structure, information access methods, look and feel for the Web pages, and guidelines for content development.

Information Structure

To better organize, store, and present information in a given application, we need to create an appropriate information structure. The choice of information structure depends on the type of application, the nature of the information, and the technology used to store and retrieve the information. The information structure also determines the granularity of information (i.e., the level of detail) that can be directly accessed.

For instance, information can be stored as a document using HTML or XML, or it can be stored in a database. If we store the information in a database or as an XML document, we can further subdivide the information content into smaller subsections (such as title, headings, subheadings, author, and keywords) and provide direct access to the content of those subsections. If, on the other hand, the information is stored as a HTML document, we cannot get this fine granularity in information access, as HTML is not a content markup language but a presentation markup language.

After determining the information structure, we then need to design appropriate navigation mechanisms to access the information content. Hyperlinks and search facilities are the commonly used access methods.

We also need to develop sample Web pages to present the information. For example, if the application is an online product catalog, we need to develop the home page and a sample Web page for each product type. These sample pages will serve as templates for the creation of other product Web pages automatically by the application software. The product Web pages will be created in response to a user's request (when the Web site is operational). We can use development tools such as Frontpage and Dreamweaver to develop the sample pages and to prototype the Web site. The prototype Web site is used to test the proposed navigation mechanisms for their ease of use and other usability features.

Furthermore, as part of the design process, guidelines for content creation also need to be formulated if the information content is to be newly created or to be changed. If information is drawn from legacy or ERP systems, a mechanism to convert the information into a suitable structure and format for the Web-based system must be developed.

Application Software Development

To provide interactivity and present customized information, most Web applications use external software modules, known as application software. A simple example of this is the software for processing the information a user submits via a Web form.

Application software also enables us to develop maintainable and scalable Web sites. As we mentioned earlier, one basic principle of building a maintainable Web site is to store the data that changes over time in a single location. This way, even if data appears on different pages on a site (e.g., a salesperson's contact information might appear on every page that describes a product he or she deals with), all the changes can be made in just one place.

We can further enhance the maintainability by developing a back-end Web site that will enable authorized persons to make the changes through an online form. This approach enables us to implement a decentralized maintenance mechanism in which different people can be assigned responsibility for maintaining specific sections of the information.

To facilitate scalability, a Web-based system needs to be built on a component-based architecture, and the navigational links and buttons need to be dynamically created. A component-based architecture allows us to easily add new functions or information modules simply by adding the required functions as new components. As navigational links and buttons are dynamically created, user-requested Web pages will contain links to these new components as well.

As the system needs to be designed for scalability and maintainability, we emphasize in our methodology the need to decide on the application software architecture as part of the product model. The application software architecture impacts the processes to be used for developing the system, and this will in turn determine the development time and cost. The application software architecture shown in Figure 3 is suitable for a scalable and maintainable Web site. It can also support Web-enabled back-end systems to facilitate decentralized maintenance.

CONCLUSION

As we begin to place greater emphasis on the performance, correctness, and availability of Web-based systems, sound development processes and methodologies assume greater significance. And as systems become larger, large teams of people with different types and levels of skills will be required to work together, necessitating distributed collaborative development. A well-defined process and good project management are the keys to success in Web development.

We have presented a Web development process and methodology that have been successfully applied in the development of many Web applications, including those of the ABC Internet College and the University of Western Sydney [5]. Our key recommendations for successfully developing and implementing large maintainable and scalable Web applications are to:

- Adopt a sound strategy and follow a suitable methodology to successfully manage the development and maintenance of Web sites.
- Recognize that, in many cases, development of a Web application is not an event, but a process, since the application's requirements evolve. It will have a start, but it will not have a predictable end as in most traditional software development.
- Within the continuous process, identify, plan, and schedule various development activities such that they have a defined start and finish.
- Remember that the planning and scheduling of the activities is very important to successfully manage the overall development, allocate resources, and monitor progress.
- Repeat these activities, as required, to meet the changing needs and requirements of the Web application.
- Consider the big picture during planning and designing Web applications. If you do not, you may end up redesigning the entire system and repeating the process all over again. If you address the changing nature of requirements and information, appropriate look and feel, and all other related aspects early on, you

Athula Ginige is a professor of information technology and head of the School of Computing and Information Technology at the University of Western Sydney, Australia. Dr. Ginige is also the director of the AeIMS (Advanced enterprise Information Management Systems) Research Centre at the university. His research interests are in the areas of multimedia systems, development of large-scale Web-based information systems, information structures for interactive flexible learning modules, information retrieval strategies for the Web, and e-business systems for small to medium-sized enterprises.

Dr. Ginige has been involved in design and development of large Web-based systems and has published on Web development methodologies. He is the main author of three chapters in the book Hypermedia and the Web: An Engineering Approach *(John Wiley, 1999).*

He graduated with a B.Sc. in engineering, first class honors, from the University of Moratuwa, Sri Lanka, and received his Ph.D. in computer vision from the University of Cambridge. Dr. Ginige is a fellow of the Cambridge Commonwealth Society and a member of the editorial board of IEEE Multimedia. *He was the national president of the Australian Pattern Recognition Society (APRS), 1994-1995.*

can build into the design efficient and cost-effective ways of managing change and new requirements.

- Recognize that development of a large Web application calls for teamwork and shared responsibility among the team members, and motivate a team culture.

We would like to stress that without a well-defined, systematic approach, it is impossible for a group of people to effectively work together on a large Web development project. We strongly recommend that Web developers and their project managers move away from an ad hoc approach to a well-planned, systematic approach for the development of large, high-performance, evolutionary, and/or mission-critical Web sites and applications.

REFERENCES

1. Becker, Shirley A., and Florence E. Mottay. "A Global Perspective on Web Site Usability." *IEEE Software*, Vol. 18, No. 1 (January-February 2001), pp. 54-61.

2. Cardellini, Valeria, Michele Colajanni, and Philip S. Yu. "Dynamic Balancing on Web-Server Systems." *IEEE Internet Computing*, Vol. 3, No. 3 (May-June 1999), pp. 28-39.

3. Cloyd, Molly Hammar. "Designing User-Centered Web Applications in Web Time." *IEEE Software* (January-February 2001), pp. 62-69.

4. Epner, Mike. "Poor Project Management Number-One Problem of Outsourced E-Projects." *Cutter Consortium Research Brief* (7 November 2000).

5. Ginige, Athula. "Web Engineering in Action." In San Murugesan and Yogesh Deshpande (eds.), *Web Engineering* (Lecture Notes in Computer Science – Hot Topics). Springer-Verlag, 2001.

6. Ginige, Athula, and San Murugesan. "Web Engineering: An Introduction." *IEEE Multimedia*, Vol. 8, No.1 (January-March 2001), pp. 14-18 (www.computer.org/multimedia/mu2001/pdf/u1014.pdf).

7. Ginige, Athula, and San Murugesan. "The Essence of Web Engineering." *IEEE Multimedia*, Vol. 8, No. 2 (April-June 2001), pp. 22-25 (www.computer.org/multimedia/mu2001/pdf/u2022.pdf).

8. Iyengar, Arun, Jim Challenger, Daniel Dias, and Paul Dantzig. "High-Performance Web Site Design Techniques." *IEEE Internet Computing*, Vol. 4, No. 2 (March-April 2000), pp. 17-26.

9. Murugesan, San, and Yogesh Deshpande, eds. *Web Engineering: Managing Diversity and Complexity of Web Application Development* (Lecture Notes in Computer Science – Hot Topics), Vol. LNCS 2016 (www.springer.de/cgi-bin/search_book.pl?isbn=3-540-42130-0). Springer-Verlag, 2001.

10. Nielsen, Jacob. *Designing Web Usability: The Practice of Simplicity*. New Riders Publishing, 1999.

11. Pressman, Roger S. *Software Engineering: A Practitioner's Perspective*, 5th edition. McGraw-Hill, 2000.

12. Shneiderman, Ben. "Universal Usability." *Communications of the ACM*, Vol. 43, No. 5 (May 2000), pp. 85-91.

13. WebE (Web Engineering) site: http://aeims.uws.edu.au/webEhome/.

San Murugesan is an associate professor in the School of Computing and Information Technology at the University of Western Sydney. He is also the associate director of AeIMS (Advanced enterprise Information Management Systems) Research Centre at the university. His research interests include Web engineering, e-business technologies and applications, e-business transformation, intelligent agents, enterprise information systems, and information-retrieval schemes.

Dr. Murugesan has been the founding co-program chair of the annual workshop series on Web Engineering held in association with the World Wide Web conferences (1998-2001). He is coeditor of the book Web Engineering: Managing Diversity and Complexity of Web Application Development *(Springer Verlag, 2001) and is also the founding chair of the ACM SIGWEB Work Group on Web Engineering. Prior to joining the University of Western Sydney, he served in various senior positions at the Indian Space Research Organisation, Bangalore, India, for 19 years.*

Dr. Murugesan has been an invited speaker at several conferences and professional meetings and is currently a chapter tutorial speaker of the IEEE Computer Society. He was a distinguished visitor of the IEEE Computer Society, Asia-Pacific Region (1995-96).

The authors can be reached at the School of Computing and Information Technology, University of Western Sydney, Campbelltown Campus, Bldg. 17, Locked Bag 1797, Penrith South DC, NSW 1797, Australia. E-mail: a.ginige@uws.edu.au, s.murugesan@uws.edu.au; Web site: www.aeims.uws.edu.au.