Journal of Internet Banking and Commerce

An open access Internet journal (http://www.arraydev.com/commerce/jibc/)

Journal of Internet Banking and Commerce, December 2010, vol. 15, no.3
(*http://www.arraydev.com/commerce/jibc/*)

Quality of Web-based information systems

Kazimierz Worwa, PhD, DSc
Associate Professor, Institute of Information Systems, Faculty of Cybernetics, Military University of Technology, Warsaw, Poland
Postal Address: **Military University of Technology, ul. Kaliskiego 2, 00-908 Warszawa, Poland**
Author's Personal/Organizational Website: **www.isi.wat.edu.pl**
Email: **kworwa@wat.edu.pl (please use to correspond with the authors)**
Prof. Kazimierz Worwa is an associate professor at the Faculty of Cybernetics of Military University of Technology in Warsaw, Poland. His areas of interest are software engineering and software reliability modeling.

Jerzy Stanik, PhD
Assistant Professor, Institute of Information Systems, Faculty of Cybernetics, Military University of Technology, Warsaw, Poland
Postal Address: **Military University of Technology, ul. Kaliskiego 2, 00-908 Warszawa, Poland**
Author's Personal/Organizational Website: **www.isi.wat.edu.pl**
Email: **jstanik@wat.edu.pl**
Dr. Jerzy Stanik is a head of Division of Information Management Systems at the Faculty of Cybernetics of Military University of Technology in Warsaw, Poland. His research interests include information systems used in management and analytical methods used in the management.

Abstract

The scope and complexity of current World Wide Web applications vary widely: from smallscale, short-lived services to large-scale enterprise applications distributed across the Internet and corporate intranets and extranets. As Web applications have evolved, the demands placed on Web-based systems and the complexity of designing, developing, maintaining, and managing these systems have also increased significantly. They provided vast, dynamic information in

multiple media formats (graphics, images, and video). Web site design for these and many other applications demand balance among information content, aesthetics, and performance. In accordance with the growth of the Internet and World Wide Web, there has been some research on quality issues of Web-based software systems. The differences between the Web-based information systems and conventional information systems are discussed in a paper from the perspective of software quality.

Keywords: **Web-based information systems; Web engineering; quality of Web-based software; software quality modeling**

1. Introduction

In recent years, the Internet and World Wide Web (www) have become ubiquitous, surpassing all other technological developments in our history. They've also grown rapidly in their scope and extent of use, significantly affecting all aspects of our lives. Industries such as manufacturing, travel and tourism, banking, education, and government are Web-enabled to improve and enhance their operations. E-commerce has expanded quickly, cutting across national boundaries. Even traditional legacy information and database systems have migrated to the Web. Advances in wireless technologies and Web-enabled appliances are triggering a new wave of mobile Web applications. As a result, we increasingly depend on a range of Web applications. Using Web technologies, an organization can reach out to customers and provide them with not only general information about its products or services but also the opportunity of performing interactive business transactions. Organizations investing in Web technologies and applications are looking forward to realizing the benefits of these investments, however, this would not be possible without an appropriate tool for measuring the quality of their Web sites. Therefore, the quality of a Web-based information system has become a main concern of all the users of the system, the developers of the system, and the managers of the corresponding company.

2. Evolution of Web-based information systems

The commercial use of the Internet and Web has grown explosively in the past five years. In that time, the Internet has evolved from primarily being a communications medium (email, files, newsgroups, and chat rooms) to a vehicle for distributing information to a full-fledged market channel for e-commerce. Web sites that once simply displayed information for visitors have become interactive, highly functional systems that let many types of businesses interact with many types of users.

The common first stage in the development of Web-based information systems is an information kiosk where marketing type information is presented either to boost prestige, enhance brand identification or to encourage conventional sales activity.

The second stage is to open up the one-way information kiosk system using say Web-forms, so that browsers can place orders and make enquiries from the Web pages of interests, treating the Web site as an electronic mail order catalogue.

The third stage is a re-think of the traditional boundaries between the customer and the enterprise. For example a customer might be able to go beyond browsing an on-line catalogue and clicking buttons to select products, moving instead into examining production schedules.

They might then enter customized changes to product designs and could see the impact of their requirements on production schedules, delivery times and contribution to overall cost.

The scope and complexity of current Web applications vary widely: from small-scale, short-lived services to large-scale enterprise applications distributed across the Internet and corporate intranets and extranets. Web-based applications can be grouped into the seven categories (Ginige and Murugesan, 2001):

- informational, e.g. online newspapers, product catalogs, newsletters, service manuals, online classifieds, online electronic books;
- interactive, e.g. registration forms, customized information; user-provided presentation, online games;
- transactional, e.g. electronic shopping, ordering goods and services, online banking;
- workflow e.g. online planning and scheduling systems, inventory management, status monitoring;
- collaborative work environments, e.g. distributed authoring systems, collaborative design tools;
- online communities, marketplaces, e.g. chat groups, recommender systems that recommend products or services, online marketplaces, online auctions;
- Web portals, e.g. electronic shopping malls, online intermediaries.

As Web applications have evolved, the demands placed on Web-based systems and the complexity of designing, developing, maintaining, and managing these systems have also increased significantly. For example, Web sites such as for the 2000 Sydney Olympics, 1998 Nagano Olympics, and Wimbledon received hundreds of thousands of hits per minute (Ginige and Murugesan, 2001). They provided vast, dynamic information in multiple media formats (graphics, images, and video). Web site design for these and many other applications demand balance among information content, aesthetics, and performance.

Changes in use of the Internet and Web-based information systems have had an enormous impact on software engineering (Offutt, 2002). As the use of the Internet and Web has grown, the amount, type and quality of software necessary for powering Web sites has also grown. Just a few years ago, Web sites were primarily composed of static HTML files, so-called “soft brochures,” usually created by a single webmaster who used HTML, JavaScript, and simple CGI scripts to present information and obtain data from visitors with forms. Early Web-based information systems have a typical client-server configuration in which the client is a Web browser that people use to visit Web sites that reside on different computers, the servers, and a software package called a Web server sends the HTML files to the client. HTML files contain JavaScripts, which are small pieces of code that are interpreted on the client. HTML forms generate data that are sent back to the server to be processed by CGI programs. This very simple model of operation can support relatively small Web sites. It uses small-scale software, offers very little security, usually cannot support much traffic, and offers limited functionality. This was called a two-tier system because two separate computers were involved. The Web’s function and structure have changed drastically, particularly in the past two years, yet most software engineering researchers, educators, and practitioners have not yet grasped how fully this change affects engineering principles and processes. Web sites are now fully functional software systems that provide business-to-customer ecommerce, business-to-business e-commerce, and many services to many users. Instead of referring to visitors to Web sites, we now refer to users, implying much interaction. Instead of Webmasters, large Web sites must employ Web managers leading diverse teams of IT professionals that include programmers, database administrators, network administrators, usability engineers, graphics designers,

security experts, marketers, and others. This team uses diverse technologies including several varieties of Java (Java, Servlets, Enterprise JavaBeans, applets, and Java Server Pages), HTML, JavaScript, XML, UML, and many others. The growing use of third-party software components and middleware represents one of the biggest changes. The technology has changed because the old two-tier model did not support the high quality requirements of Web software applications. It fails on security, being prone to crackers who only need to go through one layer of security on a computer that is, by definition, open to the world to provide access to all data files. It fails on scalability and maintainability because as Web sites grow, a two-tier model cannot effectively separate presentation from business logic, and the applications thus become cumbersome and hard to modify. It fails on reliability: whereas previous Web software generations relied on CGI programs, usually written in Perl, many developers have found that large complex Perl programs can be hard to program correctly, understand, or modify. Finally, it fails on availability because hosting a site on one Web server imposes a bottleneck: any server problems will hinder user access to the Web site. Current Web site software configuration has expanded first to a three-tier model and now more generally to an “N-tier” model. Clients still use a browser to visit Web sites, which are hosted and delivered by Web servers. But to increase quality attributes such as security, reliability, availability, and scalability, as well as functionality, most of the software has been moved to a separate computer - the application server. Indeed, on large Web sites, a collection of application servers typically operates in parallel, and the application servers interact with one or more database servers that may run a commercial database. The client-server interaction, as before, uses the Internet, but middleware - software that handles communication, data translation and process distribution - often connects the Web and application servers, and the application and database servers. New Web software languages such as Java are easier to modify and program correctly and permit more extensive reuse, features that enhance maintainability, reliability, and scalability. The N-tier model also permits additional security layers between potential crackers and the data and application business logic. The ability to separate presentation (typically on the Web server tier) from the business logic (on the application server tier) makes Web software easier to maintain and to expand in terms of customers serviced and services offered. Distributed computing, particularly for the application servers, allows the Web application to tolerate failures and handle more customers, and allows developers to simplify the software design. Java Server Pages (JSPs) and Enterprise JavaBeans (EJBs) let developers separate presentation from logic, which helps make software more maintainable. To further subdivide the work, developers can create a software dispatcher that accepts requests on the Web server tier, then forwards the request to an appropriate hardware/software component on the application tier. Such design strategies lead to more reliable software and more scalable Web sites. Of course the technology keeps changing, with the latest major addition to the technology being Microsoft’s .NET. It is too early to say today what type of affect .NET will have, although it does not seem to provide additional abilities beyond what is already available. Clearly, modern Web sites’ increased functionality creates a need for increasingly complex software, system integration and design strategies, and development processes.

Contrary to the perception of some software developers and software engineering professionals, Web engineering isn’t a clone of software engineering although both involve programming and software development. While Web engineering adopts and encompasses many software engineering principles, it incorporates many new approaches, methodologies, tools, techniques, and guidelines to meet the unique requirements of Web-based systems. Developing Web-based systems is significantly different from traditional software development and poses many additional challenges. There are subtle differences in the nature and life cycle of Web-based and software systems and the way in which they’re developed and maintained. Web development is a mixture between print publishing and software development, between marketing and

computing, between internal communications and external relations, and between art and technology (Ginige and Murugesan, 2001). Building a complex Web-based system calls for knowledge and expertise from many different disciplines and requires a team of diverse people with expertise in different areas. As a result, Web engineering is multidisciplinary and encompasses contributions from areas such as systems analysis and design; software engineering; hypermedia and hypertext engineering; requirements engineering; human-computer interaction; user interface development; information engineering; information indexing and retrieval; testing, modelling, and simulation; project management; and graphic design and presentation.

Successful Web-based system development and deployment is a process, not just an event as currently perceived and practiced by many developers and academics. Web engineering is a holistic approach, and it deals with all aspects of Web-based systems development, starting from conception and development to implementation, performance evaluation, and continual maintenance. Building and deploying a Web-based system involves multiple, iterative steps. Most Web-based systems continuously evolve to keep the information current and to meet user needs. Web engineering represents a proactive approach to creating Web applications. Web engineering methodologies have been successfully applied in a number of Web applications, for example, the ABC Internet College, 2000 Sydney Olympics, 1998 Nagano Winter Olympics, Vienna International Festival (Ginige and Murugesan, 2001).

Several factors inherent to Web development contribute to the quality problem. Developers build Web-based software systems by integrating numerous diverse components from disparate sources, including custom-built special-purpose applications, customized off-the-shelf software components, and third-party products. In such an environment, systems designers choose from potentially numerous components, and they need information about the various components suitability to make informed decisions about the software's required quality attributes. Much of the new complexity found with Web-based applications also results from how the different software components are integrated. Not only is the source unavailable for most of the components, the executables might be hosted on computers at remote, even competing organizations. To ensure high quality for Web systems composed of very loosely coupled components, we need novel techniques to achieve and evaluate these components' connections. Finally, Web-based software offers the significant advantage of allowing data to be transferred among completely different types of software components that reside and execute on different computers. However, using multiple programming languages and building complex business applications complicates the flows of data through the various Web software pieces. When combined with the requirements to keep data persistent through user sessions, persistent across sessions, and shared among sessions, the list of abilities unique to Web software begins to get very long. Thus, software developers and managers working on Web software have encountered many new challenges. Although it is obvious that we struggle to keep up with the technology, less obvious is our difficulty in understanding just how Web software development is different, and how to adapt existing processes and procedures to this new type of software.

3. Quality criteria for Web-based information systems

We evaluate software by measuring the quality of attributes such as reliability, usability, and maintainability, yet academics often fail to acknowledge that the basic economics behind software production has a strong impact on the development process (Offutt, 2002). Although the field of software engineering has spent years developing processes and technologies to improve software quality attributes, most software companies have had little financial motivation to improve their software's quality. Software contractors receive payment regardless of the delivered software's quality and, in fact, are often given additional resources to correct problems

of their own making. So-called “shrink wrap” vendors are driven almost entirely by time-to-market; it is often more lucrative to deliver poor-quality products sooner than high-quality products later. They can deliver bug fixes as new “releases” that are sold to generate more revenue for the company. For most application types, commercial developers have traditionally had little motivation to produce high-quality software. Web-based software, however, raises new economic issues. Unlike many software contractors, Web application developers only see a return on their investment if their Web sites satisfy customers’ needs. And unlike many software vendors, if a new company puts up a competitive site of higher quality, customers will almost immediately shift their business to the new site once they discover it. Thus, instead of “sooner but worse,” it is often advantageous to be “later and better.” Despite discussions of “sticky Web sites” and development of mechanisms to encourage users to return, thus far the only mechanism that brings repeat users to Web sites has been high quality. This will likely remain true for the foreseeable future. In software development, a process driver is a factor that strongly influences the process used to develop the software. Thus, if software must have very high reliability, the development process must be adapted to ensure that the software works well. Web software development managers and practitioners see the seven most important quality criteria for Web application success (Offutt, 2002):

- reliability,
- usability,
- security.
- availability,
- scalability,
- maintainability,
- time-to-market.

Of course, this is hardly a complete list of important or even relevant quality attributes, but it provides a solid basis for discussion. Certainly speed of execution is also important, but network factors influence this more than software does, and other important quality attributes such as customer service, product quality, price, and delivery stem from human and organizational rather than software factors.

Reliability

Extensive research literature and a collection of commercial tools have been devoted to testing, ensuring, assuring, and measuring software reliability. Safety-critical software applications such as telecommunications, aerospace, and medical devices demand highly reliable software, but although many researchers are reluctant to admit it, most software currently produced does not need to be highly reliable. Many businesses’ commercial success depends on Web software, however - if the software does not work reliably, the businesses will not succeed. The user base for Web software is very large and expects Web applications to work as reliably as if they were going to the grocery store or calling to order from a catalog. Moreover, if a Web application does not work well, the users do not have to drive further to reach another store; they can simply point their browser to a different URL. Web sites that depend on unreliable software will lose customers, and the businesses could lose much money. Companies that want to do business over the Web must spend resources to ensure high reliability. Indeed, they cannot afford not to.

Usability

Web application users have grown to expect easy Web transactions—as simple as buying a product at a store. Although much wisdom exists on how to develop usable software and Web sites, many Web sites still do not meet most customers’ usability expectations. This, coupled with the fact that customers exhibit little site loyalty, means unusable Web sites will not be used - customers will switch to more usable Web sites as soon as they come online.

Security

We have all heard about Web sites being cracked and private customer information distributed

or held for ransom. This is only one example of the many potential security flaws in Web software applications. When the Web functioned primarily to distribute online brochures, security breaches had relatively small consequences. Today, however, the breach of a company's Web site can cause significant revenue losses, large repair costs, legal consequences, and loss of credibility with customers. Web software applications must therefore handle customer data and other electronic information as securely as possible. Software security is one of the fastest growing research areas in computer science, but Web software developers currently face a huge shortfall in both available knowledge and skilled personnel.

Availability

In our grandparents' time, if a shopkeeper in a small town wanted to take a lunch break, he would simply put a sign on the front door that said "back at 1:00." Although today's customers expect to be able to shop during lunchtime, we do not expect stores to be open after midnight or on holidays. On the Web, customers not only expect availability 24 hours a day, seven days a week, they expect the Web site to be operational every day of the year—"24/7/365." Availability means more than just being up and running 24/7/365; the Web software must also be accessible to diverse browsers. In the seemingly never-ending browser wars of the past few years, some software vendors actively sought to make sure their software would not work under competitors' browsers. By using features only available for one browser or on one platform, Web software developers become "foot soldiers" in the browser wars, sometimes unwittingly. To be available in this sense, Web sites must adapt their presentations to work with all browsers, which requires significantly more knowledge and effort on developers' part.

Scalability

We must engineer Web software applications to be able to grow quickly in terms of both how many users they can service and how many services they can offer. The need for scalability has driven many technology innovations of the past few years. The industry has developed new software languages, design strategies, and communication and data transfer protocols in large part to allow Web sites to grow as needed. Scalability also directly influences other attributes. Any programming teacher knows that any design will work for small classroom exercises, but large software applications require discipline and creativity. Likewise, as Web sites grow, small software weaknesses that had no initial noticeable effects can lead to failures (reliability problems), usability problems, and security breaches. Designing and building Web software applications that scale well represents one of today's most interesting and important software development challenges.

Maintainability

One novel aspect of Web-based software systems is the frequency of new releases. Traditional software involves marketing, sales, and shipping or even personal installation at customers' sites. Because this process is expensive, software manufacturers usually collect maintenance modifications over time and distribute them to customers simultaneously. For a software product released today, developers will start collecting a list of necessary changes. For a simple change, (say, changing a button's label), the modification might be made immediately. But the delay in releases means that customers won't get more complex (and likely important) modifications for months, perhaps years. Web-based software, however, gives customers immediate access to maintenance updates - both small changes (such as changing the label on a button) and critical upgrades can be installed immediately. Instead of maintenance cycles of months or years, Web sites can have maintenance cycles of days or even hours. Although other software applications have high maintenance requirements, and some research has focused on "on-the-fly" maintenance for specialized applications, frequent maintenance has never before been necessary for such a quantity of commercial software. Another ramification of the increased update rate has to do with compatibility. Users do not always upgrade their software; hence, software vendors must ensure compatibility between new and old versions. Companies can control the distribution of Web software to eliminate that need, though Web applications must

still be able to run correctly on several Web browsers and multiple versions of each browser. Another possible consequence of the rapid update rate is that developers may not feel the same need to fix faults before release - they can always be fixed later.

Time-to-market

This has always been a key business driver and remains important for Web software, but it now shares the spotlight with other quality attributes. Most of the software industry continues to give priority to first to market. Given the other factors discussed here, however, the requirement for patience can and must impact the process and management of Web software projects. Software researchers, practitioners, and educators have discussed these criteria for years, but no type of application has had to satisfy all of these quality attributes at the same time. Web software components are coupling more loosely than any previous software applications. In fact, these criteria have until recently been important to only a small fraction of the software industry. They are now essential to the bottom line of a large and fast growing part of the industry, but we do not yet have the knowledge to satisfy or measure these criteria for the new technologies used in Web software applications.

4. Quality modelling for Web-based information systems

In the literature, there are numerous reports on various quality models for program-based software information systems, such as McCall's model (McCall and Richards and Walters, 1977), Boehm's model (Boehm et al., 1978), and Gillies's model (Gillies, 1997) etc. Regarding information systems as a much wider concept than software systems, the SOLE model (Eriksson and Torn, 1991) and its variants provide three different views for three different interest groups of stakeholder: users, technical staff and managers. In fact, the basic idea of the SOLE model is from the general theory of quality proposed by Garvin (Garvin, 1987). In accordance with the growth of the Internet and World Wide Web, there has been some research on quality issues of Web-based software systems. In (Lindroos, 1997), the differences between the Web-based information systems and conventional information systems are discussed from the perspective of software quality. It is realized that a quality model for Web-based information systems is needed. In (Olsina et al., 1999), a quality model for Web sites of universities, called Web site QEM, was proposed based on the users' view. It breaks down the quality of Web sites into more than a hundred attributes, Existing quality models were almost all proposed as general models that are intended to be suitable for a large class of software and information systems. The development of such models is mostly based on many years' experiences in the development and maintenance of software and information systems. The validation of such models is mostly by empirical studies, such as by analyzing data collected from questionnaires and interviews. The quality models investigated by such an approach may not fully address all the quality characteristics of the target systems. It is recognized that the special requirements of software and information systems must be considered in the application of quality models (Kitchenham and Walker, 1989). Different kinds of information systems may have different requirements on quality. For instance, both e-commerce systems and personal homepages are Web-based information systems, the former have quite different quality requirements from the later, in terms of information security and information searching issues. Hall argues that development of Web applications today is currently at the stage software development was 30 years ago, when 'software crisis' was first considered (Lowe and Hall, 1999). Most e-commerce systems are developed using *ad hoc* approaches, which have brought many quality problems, such as maintainability, efficiency and security. However what is lacking is a systematic method that enables developers to build a quality model for any given system.

Quality is not a new concept in information systems management and research. Information systems practitioners have always been aware of the need to improve the information systems function so it can react to external and internal pressures and face the critical challenges to its

growth and survivability (Aladwani, 1999). Moreover, information systems scholars have been concerned with definitions of quality in information systems research. Information systems researchers have attempted to define data quality (Kaplan et al., 1998), information quality (King and Epstein, 1983), software/system quality (Olsina et al., 1999), documentation quality (Gemoets and Mahmood, 1990), information systems function service quality (Kettinger and Lee, 1994), and global information systems quality (Nelson, 1996). More recently, there has been some effort to define quality in the context of the Internet (Liu and Arnett, 2000). However, the Web quality concept remains underdeveloped. Web quality is a vastly undefined concept. For the most part, existing scientific research discusses the meaning of some aspects of Web quality in a descriptive manner without delineating its major dimensions or providing tested scales to measure it. For example, Liu and Arnett (Liu and Arnett, 2000) named such quality factors as accuracy, completeness, relevancy, security, reliability, customization, interactivity, ease of use, speed, search functionality, and organization. Huizingh (Huizingh, (2000) focused on two aspects of Web quality: content and design. Wan (Wan, 2000) divided Web quality attributes into four categories: information, friendliness, responsiveness, and reliability. Rose et al. (Rose et al., 1999) provided a theoretical discussion of technological impediments of Web sites. The authors highlighted the importance of factors such as download speed, Web interface, search functionality, measurement of Web success, security, and Internet standards. Misic and Johnson (Misic and Johnson, 1999) suggested such Web-related criteria as finding contact information (e.g. e-mail, people, phones, and mail address), finding main page, speed, uniqueness of functionality, ease of navigation, counter, currency, wording, and color and style. Olsina et al. (Olsina et al., 1999) specified quality attributes for academic Web sites. These authors took an engineering point of view and identified factors such as cohesiveness by grouping main control objects, direct controls permanence, contextual controls stability, etc. Bell and Tang (Bell and Tang, 1998) identified factors such as access to the Web, content, graphics, structure, user friendliness, navigation, usefulness, and unique features. Another useful stream of research is key Web-site characteristics on the purpose and value dimensions. While the purpose dimension relates directly to the contents of the site, the value dimension relates more to the quality aspects. The trade press and Internet sources also discussed some aspects of Web quality attributes. Levin (Levine, 1999) offered tips to help a company with Web site design including fast Web page download, Web page interactivity, and current content, among other factors.

Three conclusions can be drawn from the above review (Aladwani and Palvia, 2002). First is that past Web quality research, albeit useful, is fragmented and focuses only on subsets of Web quality. For example, Rose et al. (Rose et al., 1999) list six factors, and Bell and Tang (Bell and Tang, 1998) mention eight factors. Misic and Johnson's (Misic and Johnson's, 1999) study was more extensive, but it missed several critical factors such as Web security, availability, clarity, and accuracy, to name a few. Liu and Arnett (Liu and Arnett, 2000) list 11 items and two dimensions of Web quality—information and system quality. Like the other studies, several important quality dimensions are missing from the authors' list. Second, past research lacks rigor when it comes to measuring the Web quality construct. In some cases, an ad hoc Web quality tool was suggested (Bell and Tang, N. 1998, Liu and Arnett, 2000, Misic and Johnson, 1999). However, it is not clear to the reader what is the domain of the measured construct or what refinement, validation, and normalization procedures were employed. For example, Liu and Arnett's scale included items about information to support business objectives, empathy to customers' problems, and follow-up services to customers. These items loaded on the same factor, which they called ''information quality''. In addition, double barreled items can be found in their scale, e.g. security and ease of use. Third, the majority of the suggested Web quality attributes and scales are relevant to Web designers than to Web users; like for instances, the ideas and scales proposed by Liu and Arnett (Liu and Arnett, 2000) and Olsina et al. (Olsina et

al., 1999).

The previous discussion underscores the fact that the Web quality construct lacks a clear definition and Web quality measurement is still in its infancy.

A number of psychometric researchers have proposed several procedural models to help other researchers develop better scales for their studies, e.g. (Babbie, 1992 and Churchill, 1979). Applying these concepts, MIS researchers have developed several instruments, e.g. the end user computing satisfaction by Doll and Torkzadeh (Doll and Torkzadeh, 1988) and the microcomputer playfulness instrument by Webster and Martocchio (Webster and Martocchio, 1992). Straub (Straub, 1989) described a process for creating and validating instruments in IS research, which includes content validity, construct validity and reliability analyses. Three generic steps common in all these models include conceptualization, design, and normalization (Aladwani and Palvia, 2002). Conceptualization focuses on content validity, involves such activities as defining the construct of interest and generating a candidate list of items from the domain of all possible items representing the construct. The second step, design focuses on construct validity and reliability analysis. It pertains to the process of refining the sample of items from the previous step to come up with an initial scale, deciding on such operational issues as question types and question sequence, and pilot-testing the initial scale that has been developed in the preparation stage. The third and last step concerns the effort to normalize the scale that has been developed. It involves the important steps of subsequent independent verification and validation. Unfortunately, this step is omitted in many scale development efforts. In the conduct of these steps, several analytical techniques can be used such as factor analysis and reliability analysis as we will describe next.

5. Conclusion

Achieving the high quality requirements of Web software represents a difficult challenge. Although other segments of the software industry have already mastered some of these, such as the need for reliability in telecommunications and network routing, aerospace, and medical devices, they have typically done so by hiring the very best developers on the market, using lots of resources (time, developers, and testing), or relying on old, stable software and technologies. Unfortunately, these solutions will not work for Web applications. There are simply not enough of the “best developers” to implement all of the Web software needed today, and few but the largest companies can afford to invest extra resources in the form of time, developers, and testing. Finally, it should be obvious that old, stable software technologies will not suffice as a base for the Web - it relies on the latest cutting-edge software technology. Although the use of new technology involves some risk, it allows us to achieve otherwise unattainable levels of scalability and maintainability.

Past Web quality research has focused on general description of some specific aspects of Web quality and paid little attention to construct identification and measurement efforts. In this study, we moved beyond descriptive and narrative evidence to empirical evaluation and verification by developing a multidimensional scale for measuring user-perceived Web quality. The results of the two-phased investigation uncovered four dimensions of perceived Web quality (technical adequacy, specific content, content quality, and appearance) and provided evidence for the psychometric properties of the 25-item instrument. Another contribution is that while past Web quality research focuses mostly on the perspectives of Web developers and designers, the current study targets the Web users. In this era of intense competition and customer responsiveness, the users are major stakeholders and should not be ignored. The limitations of the study include those customarily associated with instrument-building and survey methods. However, the extensive testing and validation improved the internal validity, and using several groups of subjects improved the external validity and generalizability of the instrument to a larger

population. Nevertheless, instruments are always subject to further improvement and we encourage fellow researchers to do so. The Web quality model/instrument has practical as well as theoretical and research applications. In terms of practical applications, a validated instrument provides an important tool for assessing the quality of Web site. The Internet is hosting hundreds of millions of Web sites varying widely in terms of quality. The scales might be used to assess the quality of a given Web site. This could be carried out at the overall quality level using the 25-item instrument or at a specific quality dimension level, e.g. using a sub-scale of one of the four dimensions of perceived Web quality. This evaluation may provide a fast and early feedback to the firm. If the firm finds itself lacking in any of the dimensions, then it may do a more detailed analysis and take necessary corrective actions. The four dimensions and the 25-items of the instrument may also be used by Web site designers in a proactive manner. The dimensions and the items may be considered explicitly in the site design. Additionally, a firm may want to assess the relative importance of the four quality dimensions and the specific items in its own context. While this study provided their relative importance based on its own sample, each firm is unique based on its products, services, customers, business strategies, etc. Such an evaluation would facilitate the design of a quality Web site in the first place.

References

Aladwani, A.M. (1999) Implications of some of the recent improvement philosophies for the management of the information systems organization, Industrial Management and Data Systems 99 (1), pp. 33–39.

Aladwani, A.M., Palvia, P.C. (2002) Developing and validating an instrument for measuring user-perceived Web quality. *Information & Management*, *39*, 467–476.

Babbie, E.R. (1992) The Practice of Social Research, Wadsworth Publishing Company, Belmont, CA.

Bell, H. Tang, N. (1998) The effectiveness of commercial Internet Web sites: a user's perspective, Internet Research, 8 (3), pp. 219–228.

Boehm, B.W., Brown, J., Kaspar, H., Lipow, M., MacLeod, G., Merrit, M. (1978) Characteristics of Software Quality, Vol. 1 of TRW Serious of Software Technology, North-Holland, New York.

Churchill, G.A. (1979) A paradigm for developing better measures of marketing constructs, Journal of Marketing Research, 16 (1), pp. 64–73.

Doll, W.J., Torkzadeh, G. (1988) The measurement of end-user computing satisfaction, MIS Quarterly, 12 (2), pp. 259–274.

Eriksson, I., Torn, A. (1991) A Model for IS Quality, Software Engineering Journal, July, pp. 152-158.

Garvin, D. (1987) Competing on the Eight Dimensions of Quality, Harvard Business Review, 1987, (6), pp. 101- 109.

Gemoets, L.A., Mahmood, M.A. (1990) Effect of the quality of user documentation on user satisfaction with information systems, Information and Management, 18 (1), pp. 47–54.

Gillies, A. (1997) Software Quality: Theory and Management, International Thomson Computer Press.

Ginige A., Murugesan S. (2001) Web Engineering: An Introduction. IEEE Multimedia, 1-3, pp. 14-18.

Huizingh, E.K. (2000) The content and design of Web sites: an empirical study, Information and Management, 37 (3), pp. 123–134.

Kaplan, D., Krishnan, R., Padman, R., Peters, J. (1998) Assessing data quality in accounting information systems, Communications of the ACM, 41 (2), pp. 72–78.

Kettinger, W.J., Lee, C.C. (1994) Perceived service quality and user satisfaction with the information services function, Decision Sciences, 25, pp. 737–766.

King, W.R., Epstein, B. (1983) Assessing information system value, Decision Sciences, 14 (1), pp. 34–45.

Kitchenham, B.A., Walker, J.G. (1989) A quantitative approach to monitoring software development, Software Engineering Journal, 4(1), pp.2-13.

Levine, G. (1999) 10 steps to building a successful Web site. Bobbin, 40 (8), pp. 61–63.

Lindroos, K. (1997) Use Quality and the World Wide Web, Information and Software Technology, vol. 39, pp. 827-836.

Liu, C., Arnett, K.P. (2000) Exploring the factors associated with Web site success in the context

of electronic commerce, Information and Management, 38 (1), pp. 23–33.

Lowe, D., Hall, W. (1999) Hypermedia and the Web: an engineering approach, Chichester: John Wiley.

McCall, J., Richards, P., Walters, G. (1977) Factors in Software Quality, Technical report CDRL A003, US Rome Air Development Centre, Vol. I.

Misic, M.M., Johnson, K. (1999) Benchmarking: a tool for Web site evaluation and improvement, Internet Research, 9 (5), pp. 383–392.

Nelson, K.G. (1996) Global information systems quality: key issues and challenges, Journal of Global Information Management, 4, pp. 4–14.

Offutt, J. (2002) Quality Attributes of Web Software Applications. *IEEE SOFTWARE***,** 3-4, pp. 25-32.

Olsina, L., Godoy, D., Lafuente, G.J., Rossi, G. (1999) Specifying Quality Characteristics and Attributes for Websites, in Proceedings of the First ICSE Workshop on Web Engineering, 16 -17 May Los Angeles, USA.

Olsina, L., Godoy, D., Lafuente, G.J., Rossi, G. (1999) Specifying quality characteristics and attributes for Web sites, First ICSE Workshop on Web Engineering (WebE-99), Los Angeles.

Rose, G., Khoo, H., Straub, D.W. (1999) Current technological impediments to business-to-consumer electronic commerce, Communications of AIS, 1(16).

Straub, D.W. (1989) Validating instruments in MIS research, MIS Quarterly, 13 (2), pp. 147–169.

Wan, H.A. (2000) Opportunities to enhance a commercial Web site, Information and Management, 38 (1), pp. 15–21.

Webster, J., Martocchio, J.J. (1992) Microcomputer playfulness: development of a measure with workplace implications, *MIS Quarterly*, 16 (2), pp. 201–226.

Zhang, Y., Zhu, H., Greenwood, S., Huo, Q. (2001) Quality Modelling for Web-Based Information Systems. *Proceedings of the Eighth IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS.01)*.