The real departure for Web-based enterprise applications is the possibility of wide-ranging accessibility. A system that might be deployed in-house at a manufacturing company can now be deployed to all the dealers of that manufacturer’s products. Indeed, it can be deployed to all the customers. In the past, such a wide deployment was a major hurdle. With Web-based technologies, providing external access to internal systems is as easy as publicizing a URL.

The real departure for Web-based enterprise applications is the possibility of wide-ranging accessibility. A system that might be deployed in-house at a manufacturing company can now be deployed to all the dealers of that manufacturer’s products. Indeed, it can be deployed to all the customers. In the past, such a wide deployment was a major hurdle. With Web-based technologies, providing external access to internal systems is as easy as publicizing a URL.

Without doubt, the Web has significantly affected the users of computer systems, but what effect has it had on enterprise application designers? That’s the question we wanted to explore with this special issue. What things are the same, what things are new, and what things have changed in some subtle but interesting ways? The activities that people need to do to build Web software do not differ from those we do with any software. We still analyze requirements, design, write code, test, and deploy. Yet the pressures of building Web applications add some new variations of familiar activities. None of these changes are necessarily massive, but they suffice to suggest some new directions.

**Quality, scalability, usability, maintainability**

Jeff Offutt’s article “Quality Attributes of Web Software Applications” is a good starting point for this discussion; he explores quality attributes for Web-based systems. Although the Web hasn’t necessarily introduced any new quality issues, Web applications often combine more quality challenges than previous generations of applications. Furthermore, the Web’s capabilities introduce new twists on traditional ideas of quality.

The Web also brings the question of scalability to the fore. Whereas a system might have had a few thousand users a decade ago, today Web system designers must consider the possibility that hundreds of thousands of users might log in at once. Colleen

---

**Elisabeth Hendrickson**, Quality Tree Software

**Martin Fowler**, ThoughtWorks
Roe and Sergio Gonik dig into the issue of scalability in “Server-Side Design Principles for Scalable Internet Systems.” Their article culls principles from their experiences with software that is both behaviorally complex and that needs that elusive scalability.

Additionally, many Web users are end customers. Web applications have a greater need for usability than many enterprise applications previously demanded. Larry Constantine and Lucy Lockwood tackle this problem with techniques for designing user interfaces through user models, task models, and content models in their article, “Usage-Centered Engineering for Web Applications.”

When we talk about user interfaces, we must remember that as well as designing them from the outside to be usable, we must also design them from the inside to be maintainable. Web applications need to change as business changes, and even the most usable site can become a problem if the interface’s internal design is difficult to change and fix. In their article, “Objects and the Web,” Alan Knight and Naci Dai peer inside a design to reveal that one of the oldest patterns for user interfaces, the Model-View-Controller, is a key part of maintainable Web systems. Indeed, MVC plays a bigger-than-ever part in the Web.

Agile, international, testing, lessons learned

Web applications must be up quickly to compete. In “Going Faster: Testing the Web Application,” Edward Hieatt and Robert Mee explore this dilemma and suggest that high quality is not just compatible with high speed—quality enables speed. By writing the tests before the code, the authors discovered that they not only produced a more solid end product but also could do it more quickly.

It’s hard to write an introduction this far without raising the spirit of another recent fad: agile methods such as Extreme Programming. Throughout many of these articles, you’ll see that theme. Web applications seem particularly suited to agile processes’ emphasis on quality and speed. One feature of agile methods is their de-emphasis on modeling. In “Lessons in Agility from Internet-Based Development,” Scott Ambler describes a case study of how a Web development team used modeling within an agile process. He concludes that there is less emphasis on documentation and tools but more emphasis on communication and quality. Modeling is there, it’s recognizable, but a little different.

The Web’s wide reach plays particular havoc with those who like to think that world revolves around the English language. Native English speakers often run into Web pages they can’t read and get a taste of what it’s like for many in the world who don’t have a dominant grasp of English. This leads to a new challenge: enabling Web applications for international use. In “Software Localization for Internet Software: Issues and Methods,” Rosann Webb Collins focuses on the needs of internationalization, providing one step toward a checklist for Web applications that need a global reach.

Eric Altendorf, Moses Hohman, and Roman Zabicki provide a nice summary of lessons learned from Web development in “Using J2EE on a Large, Web-Based Project.” This is a case study of a moderately large Web application using the popular enterprise Java technologies. The authors discuss the challenges they faced designing dynamic HTML user interfaces, tackling internationalization, and integrating third-party systems that were developed before the Web was around.

Our challenge as a profession, both in Web development and in wider software development, is to better understand and communicate these lessons. We hope this issue helps to do just that.