Technology | DOI:10.1145/1610252.1610260 Alex Wright

Ready for a Web OS?

A new generation of browsers may finally herald the long-awaited convergence of the Web and operating system.

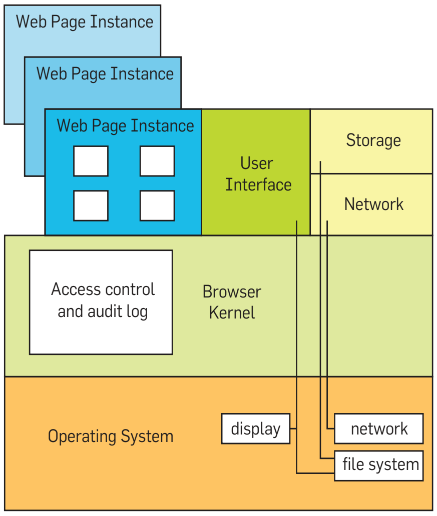
BACK IN 1995, Netscape cofounder Marc Andreessen predicted that his fledgling Web browser would one day render Windows obsolete. Fifteen years later, Netscape is long gone, and the traditional desktop operating system (OS) remains firmly established on most personal computers. Meanwhile, Web browsers still look a lot like they did in the mid-1990s, running inside application windows. In hindsight, Andreessen may have spoken a bit too soon. But history may yet prove him right.

The hegemony of the desktop OS is starting to fracture with the emergence of a new generation of browsers that may finally herald the long-awaited convergence of Web and OS. An enormous amount of Web OS development is currently under way, with the development of Web standards, such as HTML5, to add richer capabilities and features; new technologies like Microsoft's Xax and Google's Native Client that make browsers and their applications as capable and powerful as desktop applications; and architectural changes to browsers, making them process oriented, which increases their robustness and security.

A Web OS offers enormous promise. Potentially, it could take the best of the Web—the rapid deployment and updating of new applications, device independence, and the ease and convenience with which large communities can collaborate and share information—and combine it with the advantages of desktop applications—operating at machine speed, rich and interactive interfaces, and access to local hardware—and sidestep many of the security and compatibility issues currently plaguing desktop OSs. Before the Web OS becomes a practical reality, however, browser developers must overcome several major obstacles to security and device integration that continue to tilt the balance of power in favor of the desktop OS.

For many average computer users, the browser has become their de facto OS—a tool of choice for e-mail, personal finance, and other activities that were once the domain of desktop applications. Today's Web has come a long way from its original incarnation as a collection of passive, hyperlinked documents. Web developers now routinely use sophisticated scripting languages and other active client-side technologies to provide users with rich experiences that approximate the performance of desktop applications, including features like drag-and-drop, keyboard shortcuts, and other desktop-like affordances that have become commonplace.

The latest Web browsers include powerful features that further close the gap between Web applications and native desktop applications. Most major browsers have significantly increased the speed of their JavaScript engines, allowing more complex and computationally demanding applications to be developed. And Web applications will soon benefit from evolved Web standards, such as HTML5, featuring offline support, local storage, geolocation capabilities, graphics acceleration, and perhaps access to client devices, such as a scanner or video camera.



To improve the performance and security of the OP Web browser, the main architecture is divided into five main subsystems: browser kernel, storage subsystem, network subsystem, user-interface subsystem, and Web page instances.

"With HTML5, we're going to see a new generation of rich Web applications," says Adam Barth, a postdoctoral fellow at the University of California at Berkeley who focuses on privacy and browser security. "But I suspect it will take a while for application developers to realize the full potential of the various HTML5 technologies like canvas, local storage, and video."

In a similar vein, experimental technologies, such as Xax and Native Client, allow Web publishers to implement Web programs as native x86 code that executes directly and safely on the client's processor, eliminating the interpretation or compilation overhead of scripted or byte-coded languages and frameworks such as Java, JavaScript, Silverlight, and Flash.

"I think we're going to see a proliferation of different scripting languages in the browser," says Barth. "In the past, JavaScript had a mortal lock as the lingua franca of the Web, but now, with technologies like Xax and Native Client, anyone can write an interpreter for their favorite language and run the interpreter in the browser without pestering the user to install the interpreter."

Sam King, assistant professor in the computer science department at the University of Illinois at Urbana-Champaign, thinks JavaScript is too entrenched in today's Web ecosystem to cede its dominant position anytime soon. "JavaScript is tightly integrated into modern browsers and a fundamental part of the current Web," he says. "However, as technologies such as Xax and Native Client show up in

FIGURE: CHRIS GRIER, SHUO TANG, AND SAMUEL T. KING, "SECURE WEB BROWSING WITH THE OP WEB BROWSER", PROCEEDINGS OF THE 2008 IEEE SYMPOSIUM ON SECURITY AND PRIVACY (OAKLAND), MAY 2008.

browsers, Java, Silverlight, and Flash will certainly have new competition."

Multiple Concerns

As Web sites take advantage of improved client-side technologies, browsers will need to start coping with a growing range of performance, reliability, and security concerns. "As we make Web browsers more powerful, we need to keep in mind how malicious Web sites might abuse that additional power," says Barth. For example, a truly secure browser should let users visit sites that contain buggy or malicious code without fearing about the integrity of their OS, applications, or private data.

To address these concerns, researchers are reconsidering the underlying architecture of the Web browser. New multiprocess browsers, such as Tahoma, Google Chrome, Internet Explorer 8, and the OP browser, place separate Web applications in their own operating system processes or virtual machines, allowing the underlying host operating system or virtual machine monitor to ensure that crashes and slowdowns in one Web application do not affect the performance or robustness of other applications. "By decomposing the browser, you can separate out the security logic from the implementation," says King.

To improve security, browsers such as Chrome and Internet Explorer 8 have been refactored to run untrusted components and Web code in a low-privilege sandbox, limiting the exposure of browser vulnerabilities and the damage that can be inflicted by a malicious Web application. Mozilla is exploring a similar protection mechanism in its Electrosis project that will strengthen security features in a future version of Firefox. Meanwhile, Microsoft Research developers have taken this approach a step further with the prototype Gazelle browser, which separates different Web sites into discrete protection domains. With the OP browser, researchers at the University of Illinois at Urbana-Champaign explored applying OS principles to Web browser design by breaking the browser program into smaller subsystems.

By isolating processes within a browser, browsers can mediate interactions between Web programs, such

As Web sites take advantage of improved client-side technologies, browsers must cope with a growing range of performance, reliability, and security issues.

as mashups and embedded widgets, to prevent one program from stealing information from or causing damaging side effects to other programs. However, the techniques that accomplish this often come at the expense of backward compatibility, making them more difficult to deploy.

Looking further ahead, questions of compatibility will continue to arise as browsers must negotiate a growing tangle of local computing resources such as offline storage, cameras, microphones, geolocation, and graphics acceleration hardware—all of which have their own security and performance issues.

Geolocation provides an instructive example of the types of challenges that browser developers will likely face in the near future. "Geolocation is an interesting case from a security point of view because we're resorting to asking users to grant Web sites additional privileges to read location data," says King. Developers must negotiate delicate trade-offs in giving users an appropriate level of control without letting them also damage their systems. "Developing a security policy for accessing local data and hardware is still an open, difficult, and important research problem," he says.

The continued proliferation of devices—each with its own OS and interface—may provide further impetus toward a consistent, predictable Web OS. With new devices like the Palm Pre smartphone, the CrunchPad Web tablet, and various netbooks running Google's forthcoming Chrome OS, all user interaction will take place through a browser or Web-based applications. While each of these devices may still have a different OS providing a scaffolding of background processes, users will increasingly experience these devices through the filter of a Web interface.

As developers take advantage of these emerging technologies to craft Web-based experiences across a growing range of devices, the OS will likely continue to recede from users' awareness—and perhaps eventually disappear altogether. But even if the traditional OS sticks around in some form for years to come, it may not matter much to anyone except developers. "I'm not sure users care that much about the computing platform," says Barth. "Users seem to care much more about what they can do with technology than how it gets done." C

Further Reading

Barth, A., Jackson, C., Reis, C., and the Google Chrome team
The Security Architecture of the Chromium Browser, http://seclab.stanford.edu/websec/chromium/, 2008.

Cox, R. S., Hansen, J. G., Gribble, S. D., and Levy, H. M.
A safety-oriented platform for web applications, *Proceedings of the 2006 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2006.

Grier, C., Tang, S., and King, S.T.
Secure web browsing with the OP web browser, *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2008.

Reis, C. and Gribble, S. D.
Isolating web programs in modern browser architectures, *Proceedings of the 4th ACM European Conference on Computer Systems* (EuroSys 2009), Nuremberg, Germany, March 2009.

Wang, H. J., Grier, C., Moshchuk, A., King, S. T., Choudhury, P., and Vente, H.
The multi-principal OS construction of the Gazelle web browser, *Proceedings of the 18th USENIX Security Symposium*, Montreal, Canada, August 2009.

Alex Wright is a Brooklyn-based writer and information architect. Charles Reis (Google), Steve Gribble (University of Washington), and Hank Levy (University of Washington) contributed to the development of this article.