COLLABORATIVE WORK

Roy T. Fielding • University of California at Irvine • fielding@ics.uci.edu
Gail Kaiser • Columbia University • kaiser@cs.columbia.edu

THE APACHE HTTP SERVER PROJECT

Most reports of Internet collaboration refer to small-scale operations among a few authors or designers. However, several projects have shown that the Internet can also be the locus for large-scale collaboration.

In these projects, contributors from around the world combine their individual forces and develop a product that rivals those of multibillion dollar corporations.

The Apache HTTP Server Project* is a case in point. This collaborative software development effort has created a robust, feature-rich HTTP server software package that currently dominates the public Internet market (46 percent compared with 16 percent for Microsoft and 12 percent for Netscape, according to a June 1997 survey* published by Netcraft). The software and its source code are free, but Apache's popularity is more often attributed to performance than price.

The project is managed by the Apache Group, a geographically distributed group of volunteers who use the Internet and Web to communicate, develop, and distribute the server and its related documentation. In addition, hundreds of users have contributed ideas, code, and documentation to the project.

ORIGINS OF THE PROJECT

Prior to Apache, the most popular server software on the Web was the public domain HTTP "daemon" (httpd), developed by Rob McCool at the National Center for Supercomputing Applications,* University of Illinois at Urbana-Champaign. However, after McCool left NCSA in mid-1994 to work for Netscape, development of NCSA httpd stalled, and many Webmasters began to develop their own extensions and bug fixes.

In February 1995 a small group of these Webmasters gathered together via the Internet to coordinate their changes and produce a common distribution. A mailing list, shared information space (FTP- and HTTP-served directories), and logins for the core developers were created on a machine in California's San Francisco Bay area, with bandwidth and disk space donated by *HotWired** and *Organic Online*.*

Less than a year after its first release, the Apache server surpassed NCSA's httpd as the leading server on the Internet. By June 1997, the Netcraft survey reported more than a half million Web sites using Apache and its derivatives.

PROCESS CONTEXT

Choosing appropriate methods and tools for collaboration depends on the process context: where, when, and how each project member is able to communicate their ideas and plans, and work with other team members. Even the most ideal tool for communication is useless if it cannot run on all platforms used by those who need to communicate. Likewise, a method of interaction that requires a certain type of work behavior (for example, availability for a teleconference at a certain time of day) will fail if that behavior is not shared.

Apache's process context has been dominated by its global distribution and voluntary organizational environment. It has always been a multinational project, with the core developers located in the US, Britain, Canada, and Italy, and significant contributions from developers in many other countries. Collaboration within the group is hindered by the variation in work schedules and the effect of cross-Atlantic network latency and by restrictions of national export controls. It is too expensive to connect the entire group by telephone and too cumbersome/lossy to use the network as the conduit for synchronous voice or video conferencing. In any case, the group has yet to find a time at which any large portion of the project participants can be tied to their terminals for synchronous collaboration.

Each Apache Group volunteer has (at least) one other "real" job, usually related to either Web services or protocol research. They collaborate on producing and supporting the Apache server out of enlightened self-interest: by pooling their efforts, the resulting product is much more functional and robust than anything they could have produced alone. In contrast to traditional software development projects, all coordination tasks are entirely voluntary: There is no Apache CEO, president, manager, or even secretary. People volunteer for needed tasks and rotate tasks when they get tired or too busy to continue. Decisions are made by consensus (for code changes or legal issues) or by majority vote, with the voting taking place via the mailing list. As a result, a decision that requires input from all of the active members usually takes about 36 hours.

The constraints generated by the Apache process are not all bad. Because project communication is limited to e-mail, it can be automatically archived for later use. There is no need to take meeting notes or to transcribe design decisions after the fact, and thus nothing is irretrievably lost. Ideas

can be revisited over time, and new project members can read the entire archive when they join, thereby gaining an understanding of the project history, which is often more complete than the memory of the original contributors.

Furthermore, because the entry barrier for participation is low and requires no expensive hardware or software, the pool of potential contributors is huge.

COLLABORATION METHODS AND TOOLS

Effective large-scale collaboration requires

- a large pool of potential contributors (people);
- at least one common goal;
- a means for communication, both public (one to all, recorded for later review) and private (one to a few, to resolve personal conflicts);
- a shared information space for access to both current and past communication and development artifacts; and
- coordination, for all of the above to work in concert.

Developer Mailing List

The developer mailing list has been the primary means of public communication for Apache. Averaging 50 messages a day, with peaks of over 200 messages on days near a software release, topics include designs for new features, bug fixes, user problems, news about the Web community, product release dates, and project strategy. Private e-mail is reserved for resolving localized conflicts between group members, or when the discussion includes confidential information.

Tracking progress and potential conflicts has often resulted in a deluge of e-mail. Maintaining an adequate archive of that communication requires more sophisticated data management and retrieval techniques than string pattern-searching or hypertext reply-threading can supply. Although most e-mail discussions can be automatically categorized and threaded, Internet mail applications vary to such an extent that an "e-mail librarian" interface is needed to manually fix incorrectly categorized or threaded discussions.

Unfortunately, existing hypertext e-mail archival systems, such as Hypermail* and MHonArc,* are batch-oriented and do not provide such an interface. To handle a list with such a large volume of traffic, an archival system must provide views at varying levels of abstraction, such that the archive can be browsed without displaying too much information at once.

Change Control

At first, the Apache Group members relied solely on e-mail and FTP for project collaboration. A proposed change would be sent to the mailing list in the form of a *patch* (the set of differences between the current and proposed versions of the file(s) being changed), interested developers would apply and test the change on their own systems, and then the core developers would vote on its inclusion in the distributed software system. Coordination was achieved by numbering each patch and having a volunteer *patch coordinator* manually maintain and circulate the list of proposed patches. A *vote coordinator* (possibly the same person) would announce a voting period, generally in relation to the next proposed release date, and tally the votes for each patch. Finally, a *release builder* would take the list of approved patches and apply them to a copy of the last release, and then use FTP to store this new release in the shared information space.

Although this patch-vote-release process was effective in controlling quality, it was cumbersome and often frustrating. Proposed patches frequently overlapped or were rejected due to minor problems, causing weeks of delay. Attempts to centralize and automate the patch and vote coordination tasks via a Web-based forms interface failed because of the network latency problem: It was easier for members to simply reply to an e-mail message than connect to a remote Internet site and fill out a form.

CVS. Switching to the Concurrent Versions System* made it easier to apply and track patches. In this system, the project's current software base, documentation, and information files are held in CVS repositories. Core developers can check out a copy of the repository, in essence replicating it on their own machines. The developers can change files and test as needed, and selectively commit those changes back to the central repository using a single CVS command. A change to the repository triggers an e-mail message describing the change. This message is sent to a separate mailing list to which anyone can subscribe. Other developers can then use a CVS command that searches for modifications to the remote repository and merges them into the copy on their machine.

GNATS. Although CVS made it easier to apply and track changes, it did nothing to support the tracking of unsolved problems. Thus a central problem-tracking system with both Web and e-mail interfaces was installed. The Apache project currently uses the Gnu Problem Report Management System.* A more recently developed system called PTS (Project Tracking System) is now available. This system embeds directly into an Apache server and uses a back-end database system for all transactions.

SSH. With improved remote access to the information space came increased network security risks. Developers could not execute the remote CVS commands without their system passwords being sent in the clear over the Internet. To address this problem, the group installed the SSH (Secure Shell) Remote Login Program* on all of the development systems. SSH provides strong authentication using several public key methods, operates transparently for execution of remote commands or logins, and automatically encrypts all communication. Although the Apache project itself hasn't required encrypted communication, an Internet-based project that dealt with confidential materials (such as proprietary source code) would find it a necessity.

The mailing list, CVS, GNATS, and SSH have supported low-level coordination: retaining project history, tracking problems and revisions, providing and controlling remote access, and preventing change collisions (the "lost update problem"). However, Apache has yet to find a means to improve automation of the group decision-making process, that is, for deciding what changes should be made and when and for tracking progress of development efforts.

BEING VIRTUAL

Apache is the ultimate virtual enterprise. Developers share no corporate ties or traditional organizational infrastructure, all significant communication takes place via the Internet, and the participants are free to join or leave the project at any time. However, Apache succeeds largely because of the unusual expertise of its user/developers. All are Internet experts, experienced in communicating rapidly via e-mail and capable of installing the specialized software tools needed to support such large-scale collaboration.

The future of Internet collaboration will depend on the extent to which support for collaboration and project coordination can be built into the Internet's infrastructure. The applications that enable such collaboration will need to be easy to use and pre-installed on the collaborators' desktops, just as existing World Wide Web browsers are nearly ubiquitous on Internet-accessible computers.

An Internet Engineering Task Force* working group on Distributed Authoring and Versioning on the World Wide Web (WEBDAV)* was created in mid-1996 to enable broad interoperability of distributed Web content authoring tools (see the Collaborative Work column from March/April*). The OzWeb* project is adding project-specific subweb organization to the Web with an object-oriented database veneer. The subweb enables workflow modeling and enactment, semantics-based concurrency control and failure recovery, tool launching and management, and other group-space services on top of the Web infrastructure. Similarly, the Endeavors* project is developing a lightweight, flexible execution infrastructure for process coordination and automation. The Endeavors development environment allows user/developers to build workflow process descriptions visually, and components of the process—as well as the means to execute it—can be deployed via Java applets in Web pages and e-mail attachments.

An interesting lesson of the Apache experience is that the best tools for large-scale Internet collaboration are those that effectively minimize actual use of the Internet. Disconnected operation—the ability to perform work without being blocked by network delays and failures—is essential when the network is uncontrolled or spans long distances. The characteristics of e-mail (store-and-forward message processing) and remote CVS (replicated workspaces with batch-oriented update and commit) reflect the needs of Internet-based collaboration, and thus provide valuable lessons for future collaborative tools.

USER-DRIVEN DEVELOPMENT

Aside from the server's mere existence, the Apache project has yielded some unusual benefits from the large-scale Internet collaboration. These observations are based on the Apache experience, but similar benefits have accrued to other collaborative projects, such as The Internet Movie Database* and Linux.*

Apache is best characterized as a user-driven development: The developers of the system are also its biggest customers. Because the developers have full access to production systems, the software can be thoroughly tested on those systems before it is publicly released. This forces the core developers to act as quality control, and gives them better insight into what features will actually be useful to other end-users. Likewise, the large variation between the needs of individual Web sites leads the developers to emphasize more extensible designs and a modular API.

Although the Apache Group does not officially support the server via guaranteed service contracts, the actual software support tends to be significantly better than its commercial competition. All problem reports and answers are publicly available via the problem-tracking database. User installation and configuration questions are answered via a public newsgroup, with most such answers coming from users outside the core developers. When a problem is found, a patch to fix the problem is usually generated within a week and made publicly available. Likewise, all of the software distribution is accomplished via the Web.

With a half million Web sites using the Apache server, all with complete access to the software source code, there is an unlimited supply of ideas (good and bad) for new features and improvements. Even if all of the existing core developers were to leave the project, development of the server would continue. Fortunately, Internet collaboration is not just successful—it's addictive! It is likely that such projects will increase dramatically as more people become accustomed to the Internet, and as the Internet becomes more accustomed to collaborative applications. ■

ACKNOWLEDGMENTS

Eight core contributors founded the original Apache Group: Brian Behlendorf, Roy T. Fielding, Rob Hartill, David Robinson, Cliff Skolnick, Randy Terbush, Robert S. Thau, and Andrew Wilson. They were later joined by Ken Coar, Mark J. Cox, Dean Gaudet, Jim Jagielski, Alexei Kosut, Ben Laurie, Chuck Murcko, Aram W. Mirzadeh, Sameer Parekh, Marc Slemko, Paul Sutton, and Dirk-Willem van Gulik. There are many other regular contributors to the project.*

URLs FOR THIS COLUMN

***Apache HTTP Server Project** • www.apache.org/
***Apache project contributors** • www.apache.org/contributors/
***Collaborative Work: Distributed Authoring and Versioning** • computer.org/Internet/9702/collab9702.htm
***CVS** • www.cyclic.com/cyclic-pages/CVS-sheet.html
***Endeavors** • www.ics.uci.edu/pub/endeavors/
***GNATS** • www.alumni.caltech.edu/~dank/gnats.html
***HotWired** • www.hotwired.com/
***Hypermail** • www.eit.com/software/hypermail/
***IETF** • www.ietf.org/
***Internet Movie Database** • www.imdb.com/
***Linux** • www.linux.org/
***MHonArc** • www.oac.uci.edu/indiv/ehood/mhonarc.html
***NCSA** • www.ncsa.uiuc.edu/
***Netcraft June 1997 Survey** • www.netcraft.co.uk/Survey/
***Organic Online** • www.organic.com/
***OzWeb** • www.psl.cs.columbia.edu/ozweb.html
***PTS** • www.homeport.org/~shevett/pts/
***SSH Remote Login Program** • www.cs.hut.fi/ssh/