Conflict and Consensus: The Role of Standards

Charles Severance, Michigan State University

Sometime in the future we will all look back at January 1998 and laugh about the current conflicts in the technology industry. A hindsight perspective inevitably generates a few chuckles, but it also allows us to recognize that conflict is essential to innovation. Conflict energizes the entire process. If there were no conflict (over market shares, protocols, pricing structures, formats, programming languages, platforms, or standards) innovation would almost certainly stagnate.

While conflict ensures that technology will continue to change and grow stronger, it also ensures a certain forced honesty. As one organization "invades" the turf of another—especially when it comes to standards activities—we get to see the cards held in the hands of the players. It usually takes a few years, however, before we're able to smile at all the poker faces.

As one organization "invades" the turf of another, we get to see the cards held in the hands of the players.

THE PAST

While we certainly need to practice laughing about these conflicts, we must try to remember how serious these conflicts seemed during their time. One of the most serious technology wars began with IBM mainframes reigning supreme over all challengers. In the mid-to-late 80s, Unix systems powered by fast RISC processors began to invade the IBM glass houses. Innovative risk-taking organizations tentatively deployed Unix.

Like all great conflicts, there were a great many fans cheering or booing as each skirmish was played out. And within the Unix field itself there were many battles. Versions of AT&T Unix and BSD Unix waged war for shares of the marketplace. Instead of continuing the battles head to head—a course of action that could have crippled all parties involved—the vendors eventually decided to work together, forming the Open Software Foundation (OSF). While nearly all of the Unix vendors joined OSF, not everyone wanted to join. Those abstaining from OSF formed Unix International.

Instead of reducing the intensity of the battles, however, formalizing the conflicts served to increase it. Instead of small wars between individual companies, there were now two Unix superpowers. The fight for the control of the operating system of the future was on. The major battle of the Unix wars—I call it Unix War I—was the Motif versus Open Look conflict, a multiyear conflict about the shape of some buttons and whether or not the outlines of windows should have a 3D look.

Don't laugh. This was really serious business and kept many software developers on the sidelines for several years, waiting for one or the other of the technologies to win.

At the time, the two combatants were so engrossed in launching press releases at each other that they failed to notice that Microsoft was developing a graphical interface and application suite that ran on the lowly Intel processors. Instead of focusing on the deployment of a low-cost PC-based desktop with good applications, the Unix vendors waged war over the $40,000 workstation market that was about to become much less significant. By the time they stopped fighting and began to work together, the battle for the desktop operating system was over without a single shot being fired.

Interestingly, though, by the end of the warfare both sides adopted the best features from each other. Standards organizations like IEEE Posix and X/Open continuously produced standards that broadened the "least common denominator" between different Unix versions.

Because of cross-pollination—due to competition and intense standards development—Unix emerged as the most completely specified multivendor computing environment ever produced. The Unix-based standards have had some impact on all remaining viable operating systems, including MVS, VMS, Unix, and NT.

Editor: Charles Severance, Michigan State University, Department of Computer Science, 1338 Engineering Bldg., East Lansing, MI 48824; voice (517) 353-2268; fax (517) 355-7516; crs@egr.msu.edu; http://www.egr.msu.edu/~crs

THE PRESENT

The conflicts that we are currently watching—we will all be laughing about them in 2003—are the Web browser and Java wars. Netscape and Microsoft are battling over the Web browser while Sun and Microsoft are battling over Java.

The parallels between the browser battles and Open Look versus Motif are dramatic. Both battles are about user interface issues. Both battles are between two mature products, either of which would be quite sufficient for most user goals. As with the Open Look/Motif battle, once the browser battles are over, the result will be just another part of our desktop. In a few years, we won't even remember that browsers used to be separate from the OS. And maybe there is another technology quietly being developed in the background—as there was with Open Look and Motif—that will make the choice of a browser more or less irrelevant.

In Java, everything is too new to fight over. Unlike the browser battle, which features mature products, if Java stopped improving, we would probably not be very happy.

Java is the best new language to come along in a long time. It improves on C++ object facilities, it adds fundamental classes to support window-oriented programming, and it strives toward object code portability. But no language can be deemed mature until it has withstood the test of large end-user applications. Having a million or so person-years of effort resulting in a wide range of solid, mature, long-lived applications is the true test of a language. Creating a great programming language is an iterative process; Java is just beginning.

Also remember that this is not the first time that "write-once and run-anywhere" has been tried. In the late 1980s, there was a format called the Architecture Neutral Distribution Format (ANDF), which attempted the same flexibility. Vendors who were afraid that too much portability would reduce their market share were hesitant to adopt ANDF. Luckily, however, Java is the language for Internet widgets, so it has a somewhat better chance for survival than ANDF. After all, we all *need* Internet widgets.

THE FUTURE

Standards are often near the center of many of the great conflicts in the technology industry. Companies often try to create a quick "consensus standard" for their product when in the midst of a battle. When a product becomes a "consensus standard," vendors have a great opportunity for a few more press releases. Unfortunately, the standards process is a bit too slow to fit into the marketing or legal department's timeline.

Sun originally asked for a quick rubber stamp on its Java specification. While Java was initially turned down as an ISO standard, Sun has now been approved as a Publicly Available Specifications (PAS) submitter, and Java will be standardized shortly. This is a good step. Standardizing Java should help move from fierce innovation into a more stable production environment. However, if we try to standardize Java too quickly, we will miss the fundamental value of standards in the product life cycle.

Standards foster consensus and establish baselines to help innovate in new areas. Standards have a "ratchet effect" on innovation, keeping us from moving backwards. Ultimately, standards are worthless when one company tries to use them as a tool for market share. For Sun to standardize Java, Sun must allow the control of the Java standard to move outside Sun and become part of the consensus process. Java may change as a result. This loss of control must be frightening to Sun.

So what should Sun (or any company with an emerging technology) do with respect to standards? Sun needs to standardize those areas of Java in which there is consensus. If there are areas of Java that are in heavy use by application developers, then there is little risk that the specification will be changed as part of the standards process. The only reasonable approach is to standardize those parts of Java that already have consensus. Once those areas become standards, they will become the least common denominator across implementations. Then, as other areas gain consensus, they too can be standardized, which will ultimately expand the core standard.

It is not fair to claim that we were silly five years ago, but aren't any longer. Five years from now, what we're sweating over today will indeed look silly.

Issuing a press release is much less work than developing a working product. One company fires a salvo of press releases at another and the second company retaliates with another press-release strike. Meanwhile, the market gets nervous when it appears that press releases are being used to defend tenuously held turf instead of being used to showcase technological innovation.

> The market will not choose sides, no matter how much the combatants try to sway public opinion.

An exchange of lawsuits is another method used in times of extreme conflict, and is generally just as pointless. There are exceptions, of course, as in the recent Digital-Intel lawsuit. One begins to wonder, however, whether accusatory press releases and lawsuits have become just another phase in the typical product life cycle.

So what should we do? We need to cut to the chase.

In most conflicts, the only losers are the users themselves or a particular technology's market share. Brinksmanship, bravery, and team loyalty are out of place in our field. The overall market will not choose sides no matter how much the combatants try to sway public opinion with press releases. The market follows the faster, better, cheaper, more reliable, and stable technology.

We all should remember that—as demonstrated by the PC snookering Unix—the truly revolutionary battles leading to billions of profit dollars and new markets were won quietly, without a single shot being fired. The loudest battles often serve to distract the public eye from the ongoing wars. ❖