# Understanding Technology Standardization Efforts

by Stephen R. Walli
stephen.walli@gmail.com

*"We're from X/Open and we're here to help forge consensus ... in both senses of the word."*
*— Former X/Open Standards Architect.*

Technology standardization is commercial diplomacy and the purpose of individual players (as with all diplomats) is to expand one's area of economic influence while defending sovereign territory.

This paper presents an economic overview of the motivation for technology interoperability standards, because many have the wrong understanding of the drivers in technology standardization. It then gives a few key examples of how to think about standards from a business strategic perspective using IBM and Linux, Microsoft and ODF (as the counter example), and Sun and Java as an interesting example that spans a dozen years and both the standards and open source worlds.

The paper focuses on the engagement strategy for a commercial organization based on economics, and not on the design (or qualification) of standards development organizations (SDO). The development of an SDO to ensure fairness of development process, membership, standards amendment and maintenance, and intellectual property rights management is a very different discussion having much more to do with governance and policy science than economics and commercial competition in the marketplace.

## The Economics of Standards

Technology interoperability standards are specifications that define the boundaries between two objects that have been put through a recognized consensus process. The consensus process may be a formal de jure process supported by national standards organizations (e.g. ISO, BSI), an industry or trade organization with broad interest (e.g. IEEE, ECMA), or a consortia with a narrower focus (e.g. W3C, OASIS). The standards process is not about finding the best technical solution, and codifying it, but rather to find the best consensus driven solution with which all the participants can live.

The best interoperability standards enable multiple implementations to be delivered into the market. They benefit customers by enabling choice in a marketplace. A successful standard usually has many implementations, and the standard with the most implementations could be considered to "win" from a standards development producer's perspective.

The economic drivers for standardization means that most standards developers, i.e. individuals in standards development working groups, live on the production side in the market. They are vendors. This is why having a well developed standards strategy is important to commercial competition. Customers generally want a standard to procure against, and generally must trust the collective of vendors in a room to arrive at a solution. For a customer to assume the cost of standards participation, they generally have very specific mission critical needs and expertise and the standard represents a serious opportunity to manage costs within their organization, i.e. the consumer organization is likely very large with very large procurement budgets. Few customers meet this criteria. Because of when standards generally happen in a market, few academic researchers participate unless the standard directly impacts their work. Most research is focused on new innovation and not codifying existing practice and experience. Even at the national government de jure standards development level, participants are often "industry experts", i.e. producers. The government responsibility to tax payers is to ensure a national program supporting standardization that is fair and transparent (and ensures vendors can't collude) to encourage the development and adoption of "good" standards — it is not to be an expert in every area that may be standardized.

Many think of standards as creating commodities. While this may be true for very simple standards around industrial products (e.g. rail gauges, DRAM), it is typically not true of complex standards (e.g. operating system interfaces). A commodity is a product where the customer cannot distinguish products from different suppliers. In a commodity market, there are no switching costs for consumers. New producers can easily enter the market and the cost structures are well understood. Complex interface standards don't turn markets into commodity markets. Operating system or database switching costs (even between various UNIX systems and Linux distributions) are non-trivial. Such complex technology standards specify

certain predictable benefits, however, and allow customers to focus on other needs and requirements in their purchasing calculations.

The counter to a successful standard with many implementations is that a standard with a single implementation should be considered a failure, or worse yet a vendor specification (regardless of a standards organization imprimatur). Vendor specifications enable the vendor's business by encouraging complements around the vendor's technology base. They benefit the vendor over the customer. They are not "standards" although they are invariably based on de facto technologies.

It is important to observe that if there is exactly one true *shared* implementation, then a standard isn't always necessary. The POSIX working groups defined a test methods standard for the first POSIX C-based standard, because at the time there were three separate test suites available competing for certification attention. Later there was no need for an Ada POSIX test methods standard, because there was exactly one true implementation of the Ada POSIX test suite that was well managed and available to all for free. Likewise, we will never need a Perl or Python language standard. There is one true implementation of each language that is well managed and available to all to use through their respective open source software communities. This does not mean that open source software replaces the need for standards. There are times where multiple implementations are to be encouraged over a single shared implementation.

Standards often occur in a domain when an incumbent vendor over delivers in the product space. In early markets, the vendor with the best integrated solution often provides the best solution to customers and holds an advantage over the competition. Once they begin to over-deliver (providing new technology faster than customers can absorb it), customers start complaining about pricing. This is a "market" signal. It typically coincides with a time a technology domain is sufficiently mature that the tightest integration of innovation no longer provides the best value solution to customers' problems. An abstraction layer can be defined to break open the technology and the market.

Competitors will form a standards effort around a piece of shared technology that represents the right level of abstraction in the problem domain. This shared technology is a place where standards developers can use an open source software implementation as the base from which to begin standards collaboration. The final standard will differ from the shared implementation experience, but it is the foundation from which collaboration begins. (ODF differed from OpenOffice.org. POSIX differed from the UNIX systems upon which it was based.)

It is important to recognize market complexity in this discussion. The first and most obvious requirement is that the market is big enough (either in real or potential terms) to support multiple competitive offerings. The second is that the dominant player is not in (or cannot maintain) a monopoly position. The competitors want to crack open the lock an incumbent has on a market, but each competitor to the incumbent offers different benefits from one another beyond the immediate object standardized and are thus willing to co-operate with one another to open the marketplace. Competitive standards developers are generally willing to team together to standardize someone else's product space, even as they are opposed to standardizing products which represent their own core value proposition to their customers.

Market timing is also important from a technology maturation perspective. Many fear that standardizing too early hobbles innovation. This could prove true, except consumers will often pay for tightly integrated technology that better solves their problems over standardized technology. From a consumer in the market perspective, it is not yet "time to standardize". The customer value equation still favours non-standardized solutions to problems. Once the point of standardization happens in the market, it is clearly time to codify an aspect of the market, allowing new innovation to build around the stable "standardized" base.

Standards occur in a historical technology context and this is where shared technologies are often seen. Good standards codify proved ways of accomplishing things. They are based on existing practice and experience. (Another way to think about this is that it is difficult to standardize too early or ahead of a market.) As mentioned, the immediate technology domain has matured to the point that the tightest integration of innovation no longer necessarily wins, but the broader technology space is also well understood. When one looks at the Open Document Format (ODF) standard, one has to recognize the rich history of document formatting products and technologies that proceeded it, from roff to troff and runoff, GML to SGML to HTML, HP printer languages, Postscript, and PDF.

Standard builders will choose an existing technology base that represents a good abstraction layer and that has a shared community aspect to it. They are all familiar with the technology and its intellectual property landscape. When DEC competitors attacked the minicomputer space, they used the UNIX API as the abstraction level. At the time, every OEM had a UNIX implementation, regardless of how successful a product it might be. They all understood their licensing requirements with respect to UNIX. Likewise, when Microsoft competitors attacked the office document space, they used OpenOffice.org as a starting point for the specification.

It is important to recognize the relationship between intellectual property policy and standards. From an economics perspective, IP protections and standards are at opposite ends of a spectrum. A standard exists to encourage multiple implementations, while a patent or copyright exists to protect a single implementation. Every standards development organization has an intellectual property rights (IPR) policy. The point is to ensure that they have early warning of IP problems, and a way to discuss them. No SDO wants to produce a standard that has a tax wired into it. Intellectual asset management within the standards setting domain needs to be carefully managed by each participant to balance the goals of the standard setting process with the goals of the asset protection.

While DVD format standards were developed, patent pools were organized in parallel organizations such that participants could easily license technology. The IETF set a simple policy in place that requires any patents to be available on reasonable and non-discriminatory terms, and then will only recognize a specification as a full Internet Standard if there are two independently developed unrelated implementations that can communicate, assuming that any IP must therefore be available under RAND terms. OASIS has a well structured IPR policy that ensures the best commitment available from direct participants be maintained over the life of the standard. [It is important to recognize that no one can prevent an IP attack from outside the standards development organization.]

Let's look at three examples of standards efforts over the past 25 years and how key players managed them.

## IBM, POSIX, UNIX, and Linux: A Mature Corporate Standards Function

"Open Source Software is the culmination of all our open systems investments over the past 15 years," Steve Mills, IBM senior VP, announced in his LinuxWorld keynote on Wall Street in early 2001. Later in the same keynote, he observed that Linux would absolutely replace AIX, and then qualified it with "in ten years." AIX was already clearly a mature product. Linux was beginning to badly hurt Solaris with IBM's help. IBM continues to support ~250 kernel engineers working on the Linux code base from a product perspective.

The POSIX (then UNIX) standards that evolved through the 1990s, along with the C-language standard, enabled customers to develop applications that were much more source-code portable (i.e. reusable) across diverse computing systems. The standardization of UNIX (initially as POSIX) dethroned DEC's dominance with VAX/VMS in the minicomputer world, which was causing IBM and others no end of grief. (DEC had over-delivered in the market. The time for a standard was at hand.) As Linux systems began to replace expensive UNIX systems, application binary portability became more important.

Geoffrey Moore's technology adoption life-cycle is a good but not perfect model for technology adoption. A product typically reaches the end of the life cycle (replaced by other technologies) before the perfect Bell curve shape is reached. Even Moore evolved the model to account for an "elastic" plateau at the top of the bell shape. From one perspective, the area under the curve represents revenue, so the question becomes what can be done to change the slope of the curve during or after the peak to maximize revenue over the long term.

IBM consistently managed its engagement with POSIX then UNIX standardization efforts, then shifted its focus to the Linux Standards Base and consistently managed many touch points in the Linux technology world to best bottom line effect. It does this around its AIX product space and services and hardware support of Linux. A little history:

- IBM joined the IEEE POSIX efforts at their inception (1985). As POSIX meetings grew from 40 to 350 people participating at quarterly one week meetings, IBM scaled its participation. At one time they had 30 participants across the working groups (representing ~10% of the working groups). Only one company (UNISYS) had more coverage. For the three years that the POSIX efforts were

at their peak, meeting four times a year for a week at a time, each Monday evening was a closed door IBM-badged employees-only “reception”. This is where co-ordination was done across the three dozen projects spread over the ~15 IEEE POSIX working groups.

- With a strong presence globally, IBM was in a position to join different national technical advisory groups that would represent the IEEE POSIX efforts up to ISO at the working group and subcommittee levels. Of the key voting “P” (participating) members, IBM had people in the US, UK, German, Japanese, and Canadian delegations.

- The Canadian POSIX Working Group (that formed the Canadian delegation to ISO) was chaired by an IBM employee. This was the early 1990s and paper was the working medium. It was not unusual for the IBM representative to deliver an enormous stack of paper to the CPWG at it’s quarterly one day meetings to bury them in reading. A lot of it was less-than-critical “liaison” documentation. The head of German delegation representing up to DIN was also an IBM employee.

- IBM was also members of X/Open and the Open Systems Foundation (OSF) which had separate voting positions at the IEEE POSIX working group and whose employees also sat on various ISO member delegations. AIX technology was contributed into the base OSF collaborations and its “portability” specification. (The OSF was developing a shared base of operating system technology that its members would then use in their own products. This all pre-dates Linux maturation, and demonstrates how far key UNIX vendors would go to escape UNIX licensing fees.) These two organizations eventually adopted POSIX as the core of their own specifications. The two organizations were merged into The Open Group in 1996. (DEC was on the ropes. UNIX had “won”. There was no longer need for two complementary organizations.)

- Spec1170 was a related work that was originally designed by Sun and HP to duplicate the IBM AIX call interface to capture applications away from AIX. The work was “joined” by IBM and turned into the UNIX specification at X/Open (anchored around X/Open’s specification which was itself already anchored around POSIX). AIX was one of the first operating systems to certify with the X/Open “UNIX” brand (along with Solaris and NEC). The IBM press release sung the praises of AIX as secure, robust, and scalable and never mentioned the UNIX word.

- When the Linux Standards Base (LSB) declared itself in 1998, IBM immediately joined and began to encourage the effort. The LSB was an application binary interface standard (as opposed to POSIX and UNIX which were application programming interface standards). The LSB was anchored on the UNIX standard.

- The Free Standards Group (FSG) formed around the LSB. IBM was a founding funding member of the FSG. They had employees helping the effort out. They were card carrying IBM employees that also carried FSG cards. They have consistently participated, encouraged and promulgated the work.

- IBM was sometimes careful in how much it helped the LSB. I watched them chair an LSB meeting in 2001 (co-incident with the Steve Mills keynote event in NYC to be clear) where they continued to encourage the C++ ABI without necessarily providing a lot of help. They were arguably already contributing heavily. (A C++ ABI is very very difficult to get right. One might suspect that the IBM compiler labs staff that participated would know this.) The meeting was held in IBM’s Manhattan offices. (The new throttle on the LSB appears to be “accessibility”.)

- IBM was also a key founding supporter of the Open Source Development Labs (OSDL). The FSG and the OSDL were merged into the Linux Foundation in 2007.

- The UNIX/POSIX work continues to slowly evolve, fully integrating the IEEE/ISO/OpenGroup efforts. IBM maintains a key engineer on the effort, and regularly hosts the meetings. This work feeds up into the LSB where their primary investment at this point continues. (Ted Tso, a senior technical IBM employee, is presently “on loan” to help the Linux Foundation on technical matters.)

The long term careful participation in POSIX, then UNIX, then Linux standardization has allowed IBM to (i.) work with competitors to move the minicomputer centre of gravity away from DEC to UNIX, (ii.) encourage

their AIX business, and (iii.) manage their AIX business while encouraging their Linux consulting services business. In each case the work was done in conjunction with corporate marketing to build a message that addressed customer needs for open systems, "open" standards, and open source software. In each case they carefully managed their intellectual assets to share them in collaborative industry efforts. In each case, their own products and services were always positioned as the superior way to attain such an open environment.

## A Side Note on "Influence"

It is important to understand that the IBM example emphasizes the simplest rule of standards as commercial diplomacy. It is a game of trust and influence and you can not influence or earn trust where you do not participate and contribute. While the history of POSIX and the LSB may look like a Machiavellian set of steps by IBM as presented, it is really a well executed process that improves the probability of success of any single standards-related step in their overall customer-focused business strategy. One cannot "manage" a standards process — one can only influence it. The IBM engagement reflects a corporate standards culture that values

- participation and contribution across an effort (and related efforts) with the best coverage needed.
- communication and co-ordination within the company all the way up to executive messaging.
- an understanding of the purpose of standards in the marketplace.
- the company's intellectual assets appropriately at different times in a market place, with a matching mature intellectual property strategy.

There is nothing coercive or sinister about it. IBM merely demonstrates that they can walk the walk, and that showing up and being present is merely the first of many steps in building a standards campaign that has a high probability of commercial market success.

## Microsoft and ODF: Misunderstanding the Process

An effort began in late 2002 at OASIS to standardize office document formats using OpenOffice.org as the shared technology base out of which to deliver a specification. Essentially Microsoft had over-delivered on their Office product space over successive releases and customers were unhappy with continual forced upgrades. Over-delivery does not mean that there weren't many people working hard at Microsoft to deliver new innovation — only that the customer didn't value it as much anymore. The tightly integrated innovation approach was no longer necessarily the best implementation in this marketplace. The technology space was ripe for a standards attack by competitors.

Microsoft refused to participate despite being a member of the OASIS board. If standardized this specification directly reflected upon one of their primary revenue streams (about 50% of the US$40B). They were concerned that participation would give the standardization effort credibility. The Open Document Format (ODF) standard was delivered in just over two years (May 2005) from OASIS and was immediately submitted to ISO for international approval. The OpenOffice code base quickly moved to the new ODF standard and participants began to use that code base in multiple implementations of products adhering to the standard.

The Massachusetts State government chose the OASIS ODF standard to meet IT procurement needs, and this was the first "surprise" for Microsoft. They attempted to directly appeal the decision using the expense of government lobbyists (and rumoured smear campaigns), and when that didn't work, they began their own standardization efforts at ECMA around the Microsoft Office 2007 product. This meant absorbing the full cost of developing the specification, taking it through ECMA, then onto ISO over the next two years.

But the efforts to bring this specification through ISO met stiff resistance. Document size, quality, and necessity all contributed to the problems encountered, along with well managed counter-standards efforts from IBM and Sun. This left Microsoft in the position of attempting to introduce new sympathetic voting members into the National Body voting pool from countries without the infrastructure and commitment to participate long term. These votes came at enormous public perception costs (especially with EU regulators) that Microsoft tried to subvert and control the process.

Regardless of the fact that the final “fast track” vote succeeded, Microsoft is vulnerable to the next problems from competitors. Their product will likely be diverging even farther from the specification through the process, and they will be exposed to a standards certification problem in the conformance space.

Here’s how Microsoft should have played the game of a standard being developed on a key revenue stream by competitors:

- When the OASIS group began Microsoft should have immediately joined and joined in force. Dipping a toe in the water here would only let them know what their competitors were doing rather than giving them the ability to effect things. Microsoft should have had experts (multiple) in the room as "the most knowledgeable people" on the broad subject of document formats for the “predominant format in the world”.

- They should have been volunteering to host meetings, assume administrative positions, and involve as many liaison points as possible. (The more external liaison points one can create the slower the process will move. This is the unfortunate by-product of Brooks' Law and standards groups.)

- They could have expanded the charter in dire ways to continue the confusion. It took 10 years to complete the C++ language standard once that working group tried to bite off the entire HP template library as part of the initial work. I can imagine linking the ODF work to character set specifications in ISO as a way to drag out the format standard discussion for quite some time, especially if cross standards organizational liaison points were set-up. Adding accessibility needs at a national level would have added interesting strain to the discussion.

- From this point forward, they would have been able to throttle expectations and directly impact the development process to maintain a core relationship to the *evolving* specification. They could have ensured their own format/asset was protected, but with a clear relationship to the standard maintained. It would have forced a nasty amount of implementation stress onto the OpenOffice world, and on Sun and IBM by forcing them into co-operation they weren't necessarily ready to assume.

Then when an OASIS standard was finally released:

- Immediately claim to be developing a conforming implementation to be released at some appropriate time. The public relations game is very clean. They have been seen to be contributing through the development cycle. They sound reasonable in relation to the customer desire to see a standard. They can continue to advance their own format (product) and innovation, while delivering the check box of conformance.

- Immediately propose the next wave of format standardization within OASIS to maintain the illusion that the standard isn't quite baked, and could be evolving for some time.

Indeed, even though Microsoft chose not to participate, the simplest way to have made the entire "mess" go away would have been to:

- Announce support for the standard in a service pack after Office 2007 shipped. It was reasonable to not immediately leap to implement and interrupt the current delivery cycle and no customer would punish them for that delay.

- It could be dressed in almost identical language to their own standard format about having the superior technology but listening to customers.

- Indeed they could even position it with existing customers as a simple archive format rather than a "real" document format, building on the original marketing mistakes made around ODF three years ago.

Microsoft has had the historical luxury of being the de facto technology for its core product revenue streams for a very long time. Standards were never perceived to be strategic internally. Historically as they developed their standards function, there was too much emphasis on IP protection over IP management. It

has left them learning the hard way in the standards arena and the marketplace on a key revenue stream. All in all, they have chosen the weakest, most expensive, highest profile way to fail to manage a business critical standards effort.

## A Side Note of Conformance and Certification Testing

Every standard contains a conformance statement, defining what a conforming implementation must do to be able to claim conformance to the standard. These sometimes define additional useful concepts. For example, the C-language and POSIX standards each defined the concept of conforming "applications" as something a conforming implementation would need to accept. But conformance statements do not ensure conforming implementations. For that, one must turn to certification or compliance testing.

As one might imagine, independent compliance or certification testing is expensive. Test suites and policies and procedures are all required. While it is tempting to argue that the standards development organizations should be responsible, that is a facile answer that doesn't meet the underlying economics of the situation. SDOs are not funded to take on such certification work. Not every standard necessarily needs such expense applied to it. Typically it is required in procurement markets where considerable amounts of money are at stake and the cost of non-conformance high to the customer.

As such, interested parties must put their money where their mouth is. This generally happens in one of two ways. A large customer such as a government with procurement concerns can put such compliance testing and certification process in place (e.g. the US National Institute of Standards and Technology and their historical FIPS programs). Likewise a vendor consortia with a vested interest in demonstrating conformance to their specifications to the buying public can develop the certification and testing infrastructure (e.g. X/Open and their XPG4 and UNIX brand programs).

The IETF has a novel way of avoiding the expense due to the network nature of their specifications — their development policy states that to become a full Internet Standard (rather than a draft) there must exist two independent (genetically dissimilar) implementations that communicate, so conformance testing is pushed back onto the standards implementers themselves.

## Sun and Java: A Long and Twisted Road

Java debuted in early 1995 as a platform-independent programming language technology that encouraged a "Write Once Run Anywhere" message, and was linked to non-"Windows PC"-centric views of technology, notably the web browser (with Netscape help) and the TV set-top box. As such, it was immediately perceived as a threat (to be subverted) by Microsoft and an opportunity (to be managed) by IBM with its breadth of platforms. Sun began licensing the technology as a way to propagate it, while maintaining strict brand control.

It has been a tortuous path through the standards and open source worlds for over a dozen years for several reasons:

- Java was perceived as an important cross platform technology that arrived as the "open systems" standardization movement was in full swing and maturing. From a customer perspective, Java needed to be a "standard" (as POSIX/UNIX and C and Ada, TCP/IP and the rest of the Internet family of standards), especially for government procurement.

- The Web was exploding and major vendors were scrambling for dominance. Sun was becoming the UNIX server company to support the growing Web, beating out HP/UX and AIX. Microsoft was launching Windows 95 + Office 95. NT 3.5 was on the market. IBM was still pushing OS/2. While we have shared software since we have written software and the Web was enabling that sharing at an exponential rate, it was still early days for what is now considered the mainstream open source software world. MySQL would be released that year. The Apache Group was still an informal band and wouldn't incorporate as the ASF for 4 more years. Red Hat would incorporate that year, and was still running out of Bob Young's home.

- The major players (IBM, Sun, HP, Microsoft, Oracle) had long standing mistrust of one another and were adamant and emotional in their positions.

In November 1996, Sun first publicly discussed moving parts of Java to the standards world. "We really believe in this business of being an open systems company," said Jim Mitchell, (JavaSoft CTO). He

believed that over the long term, a single Java standard under independent control would be critical if the technology was to achieve its "write once, run anywhere" philosophy, and, “Putting Java out into standards body means there can be independent compliance tests.”

This is arguably where the first mistakes were made. Simply creating a standard does *nothing* to foster the creation of a conformance certification body of work.

By having an unclear plan around the brand, copyright, standards, and compliance, and allowing different executives to comment at will on the strategy and indeed change direction, they spent a painful five years. Some “highlights” along the way include:

- June 1997 ECMA approved ECMAScript (Javascript) as forwarded by Netscape after a brief (2 year) process. Sun and Netscape demonstrated to the world how to quickly drive a vendor specification through the process at ECMA, with the option of using ECMA as the back door to ISO.

- Sun sued Microsoft in 1997, arguing that Microsoft built technology into its Java products that led developers to build Java software programs that operated only on Windows.

- Spring 1997, Sun considered submitting Java to ISO as a “publicly available specification” (PAS). No company had ever been a PAS submitter before then. Sun battled through votes in July, September to November and finally won consent to be a PAS submitter. Microsoft and Intel fought diligently to prevent Sun becoming a PAS submitter, but they had no voting privileges at INCITS because of their own past ignorance of the standards arena, and had to fight from the side lines. Sun still got the embarrassment of failing to win U.S. support from NCITS for their move, even as they won ISO approval. The main concern from Sun was control of the Java name and brand.

- In 1998, Sun formally created the Java Community Process (JCP) to be stewards of the Java specification, the reference implementation, and certification and brand programs.

- In March 1998, HP began to push its clean-room developed Java virtual machine (JVM). September 1998, IBM released SanFrancisco, their Java building blocks for business vertical solutions (about 750K lines-of-code). By December, HP formally created the Real-time Java Working Group after being shut out in the JCP. It was clear Sun was being too heavy handed.

- In March 1999, Sun formed the real-time expert group at the JCP to offer a separate centre of gravity from the HP-led Real-time Java Working Group. An IBM researcher was chosen by Sun to lead the real-time JCP effort.

- April 1999 Sun considered bringing Java to ECMA as it was then clearly unwilling to submit to ISO PAS and lose editorial control of the language, or the brand and trademark. Sun announced their ECMA intentions in May. Sun marketed “the success of the JCP” as the driving centre for ECMA standardization, i.e. Sun would continue to define Java and ECMA would be the editorial centre to get a standard. This was an embarrassment considering the money and political capital burned in the previous efforts with ISO.

- At the same time, HP created the J Consortium as a formal incorporation of the Real-time Java Working Group pushing their Chai implementation, and tried (unsuccessfully) to convince U.S. NCITS to standardize on Chai without Sun participation. (Microsoft was a formal member of the J Consortium.)

- December 1999, Sun pulled Java out of ECMA arguing they want to keep the Java momentum going (as they released J2EE), and had been successful so far with the JCP. This was now deeply embarrassing for the company. They promised to review JCP procedures. Java was succeeding on servers and Sun further moved to free licensing on the client side. *[The collapse of the ECMA plan was Sun’s internal failure in managing the IP work with ECMA (October 1999) compounded by the failure of a key Sun technical player to step up as document editor.]*

- JCP 2.0 was announced in March 2000, Sun loosened its control a little, and the new rules were ratified by May. *[Some feel Sun marketing still has too much control of the JCP, and it’s too easy to “buy” a JSR.]*

- January 2001, Sun and Microsoft settled their Java lawsuit. (Microsoft got to again ship a Java client and paid Sun US$20M.) September 2001 Microsoft began C# efforts at ECMA. Sun and Microsoft returned to court in June 2003 to continue to battle over the Windows JVM. Sun's attempts at an injunction failed, but Microsoft likewise was prevented from distributing it's own JVM.

At this point, the standards game around Java had been basically exhausted. What happened next is interesting because IBM began an open source software play that would end up marginalizing Sun's control of Java developers more than the standards wrangling over the specification of the previous five years.

The Eclipse Project was created by IBM in November 2001, supported by a consortium of "stewards" that included Borland, MERANT, QNX, Rationale, RedHat and SuSE. (There would be 80 members by 2003). It was created out of the IBM Visual Age product line, acquired earlier from Object Technology International out of Ottawa.

IBM claimed they developed it "to provide all the synergies of a common shared development environment for the Java world, that existed for .NET and C# in the Windows world." *[IBM came close to dropping the technology into the Apache Software Foundation according to one source, but with the success of IBM participation at both the ASF and with the Linux community, it made more sense to experiment with developing their own open source software community.]*

The Eclipse Foundation was created in January 2004 as an independent not-for-profit corporation to act as the steward of the Eclipse community. Just as Sun took heat for keeping too tight a rein on Java in the JCP, so too was IBM taking heat for being the primary control point. The independent not-for-profit corporation was created to allow a more vendor neutral community to be established around Eclipse. At that point in time IBM had a well functioning organization that no longer required tight control to protect it. The organization continues to flourish as the hub for Java developers, and is growing beyond Java at this point, as a hub for developers in general.

In the past year, Sun has finally agreed to open source Java itself. To be clear, the long standing excuse that it would drift from the specification as open source is fatuous. Apache has never drifted off the HTTP spec. Sendmail hasn't drifted off the SMTP spec. *[Indeed, in 1995 a major vendor's C compiler **did** drift off the ANSI C standard **and** its U.S. government certification suite, and the vendor remained very quiet about that fact until they could realign the product over the following 2 years.]*

Unlike the Microsoft ODF example, where mistakes were made in real-time over the past two years and easy to point out, the Sun Java wars took place over a dozen years and through a period of intense growth on the Web and upheaval in the market place. Using 20:20 hindsight, we can see things that may have worked better (assuming clarity of goals, a plan, and alignment of executive thinking):

- In 1995, Sun already had a long history of shared collaborative development in the UNIX community. They had freely licensed IP and reference implementations before Java (e.g. ONC RPC). There was no experience until Mozilla with a more "corporate" open source license, so Sun could have led here just as IBM eventually would in 2001 with their own IBM Public License. There was industry history around collaboration in consortia already (X/Open, the OSF, UNIX International) so models of sharing specifications and technology outside of the formal standards world existed. Sun had a history of deep engagement in standards (certainly POSIX and C and the IETF).

- Creating the "Java Project" immediately in 1996 to drive the growth and adoption of Java with the reference platform, JDK, etc. would have been a fast start with the developer community. Liberally license the software, but maintain the community and the brand. Introduce the JCP as the separate specification organization for vendors. Drive the brand and certification process around the JCP. Maintain the control points in each. (My IP = My License = My Community.) Internally, ensure that the code and specification are absolutely in lock step. Ensure you work with strong partners (e.g. IBM, Oracle) in the JCP and cede some control in well defined ways with vendors whose goals are aligned.

- Evolve and merge the Java Project and the JCP into the "Java Foundation" after a few years. At this point, there is a breadth of experience and deployment. Sun could have ceded even more control, structuring the Java Foundation board in a tiered fashion with key aligned partners. At this point, consider submitting the specification for formal standardization either through the PAS process, or ECMA. As the Java Foundation would be a consortium instead of a vendor, PAS submitter status would likely

have been easy, and fast-track approval may well have been straight forward because of IBM, HP, and Sun reach in the international standards community.

- Maintain the Java trademark and certification suites at the Java Foundation with strong policy-based barriers to disturbing (i.) what can be called Java, (ii.) what can be added to the core branded technology through the specification process. Have the Java Foundation work with key procurement organizations within the U.S. government and the E.U. to recognize the brand for procurement against the standard, i.e. "we procure against ISO xxxx and recognize the Java Foundation branded mark as the proof of certification." *[Certification and conformance are not necessarily the same thing. There are interesting edges to manage here depending upon goals.]*

- Depending upon the legal problems Microsoft may have been raising through the process early on, Sun may still have had to step up to the lawsuit alone. But by the time the Java Foundation existed, they may have been in a position to create a legal defence fund amongst the key members, similar to the Linux Foundation fund, to distribute the cost and gain access to key experienced legal counsel.

A standard to meet government procurement needs backed by a strong developer community and a well recognized brand program may have been within reach in as little as 5-6 years when the clock started. It certainly would have required clear goals and commitment from executive management, and consistent execution against a long term plan which seemed to be missing from the Sun Java strategy.

## Summary

Technology standardization is commercial diplomacy where the goal of participants (as with all diplomats) is to expand one's area of economic influence while defending sovereign territory. The drivers for technology standardization are economic in nature, and commercial organizations that bear the brunt of the expense of standards development must best understand those drivers in the market to participate to best bottom-line effect.

Commercial organizations need to develop corporate standards cultures and plan appropriately to:

- Develop a standards strategy that reflects the commercial needs of the organization.

- Value participation and contribution to develop trust and influence to increase the probabilities of success.

- Development communication and co-ordination mechanisms within the company all the way to executive levels to ensure long term commercial goals are reached and plans are consistently executed.

- Manage a company's intellectual assets appropriately at different times in a market place to best commercial effect and consistently against a mature intellectual property strategy.

Such commercial standards strategies cannot be quickly assembled or quickly communicated, because it takes time to develop trust through participation.

## About the Author

Stephen was a long term participant in the POSIX and UNIX standardization efforts. He was a working group participant, balloted many pieces of the standards and their amendments, and participated in the management of the standards effort at the IEEE. He was an international participant at ISO, as document editor, and participated on behalf of three different national body delegations (Canada, U.S., UK) over a number of years. He began his participation in 1989 as a customer (EDS with GM and the U.S. government as their primary customers), but quickly ended up as a vendor, working for MKS developing a conforming POSIX.2 implementation that formed the basis of implementations from IBM, DEC, HP, UNISYS and Sun. In 1995, he was a founder and the vice-president, R&D at Softway Systems, implementing the POSIX and UNIX standards on NT to enable UNIX applications to be directly migrated to NT. A large amount of free and open source software was incorporated into the product. Softway Systems was acquired by Microsoft in 1999, where he worked for five years.

Over the years Stephen has been a regular presenter and author on standards, including books on application migration for X/Open. He maintains a blog about open source software, standards, and software business at *“Once More Unto the Breach”* (http://stephesblog.blogs.com).