POSIX: A CASE STUDY IN A SUCCESSFUL STANDARD
OR, WHY WE DON'T NEED RADICAL CHANGE IN THE SDO PROCESS

Stephen R. Walli
Softway Systems, Inc.
185 Berry Street, Suite 5514,
San Francisco, CA 94107
stephe@interix.com

Abstract

The standards processes of the various Standards Development Organizations (SDO) have been accused for the last decade of being too slow to keep pace with technological innovations surrounding such areas as the Internet and the World Wide Web (WWW). There are those that go so far as to suggest that the standards directly conflict with the ability to innovate (while at the same time providing a constant interface against which to develop this new networked world.)

This is incorrect and at best a marketing platform from which companies can attempt to push their own solutions as the "standard" solution. This paper looks at the history of one standard developed against a tumultuous background and its complete success in the face of counter-marketing. It demonstrates why the SDO process served it well and resisted changes that could have had a catastrophic effect. Relationships with industry consortia are explored in relationship to the POSIX family of standards, and their supportive role. It closes with observations for the future of the SDO process

1. Introduction

The standards development organizations (SDO) have been accused for the past decade of being too slow to keep pace with the technological innovations surrounding areas such as the Internet and the World Wide Web (WWW). Taking the time to develop a standard through the SDO process is tantamount to destroying our ability to quickly move forward to seize the technological day. It is customary marketing practice to at once claim to conform to all existing relevant standards while at the same time suggesting that the SDO process should be sped up to allow the next great leaps to occur. Marketing practices use the spectre of becoming technology Ludites to counter the customer demand for standards-based (i.e. vendor independent) solutions.

This conflict between customers desiring vendor independent solutions, and vendors desiring long term customer relationships meets in the standards arena. The reality is that the standards process embodied in the SDOs is not broken and requires no revolutionary change beyond the simple evolution of any successful process.

To demonstrate this reality, we will look at a large standards effort (POSIX) that was developed through two de jure SDO processes and become a market success through ubiquitous implementation and adoption by commercial consortia, in the face of continual marketing about its lack of success.

2. POSIX: A Brief History

POSIX is a family of standards developed through the IEEE and ISO/IEC. POSIX stands for Portable Operating System Interface for Computing Systems. It describes the operating system service interface, based on UNIX existing practice and experience, and exists to permit application programs to be written that are source code portable across multiple diverse operating systems. It was born out of work in the early 1980s led by a UNIX users group, aptly named */usr/group*, which was attempting to re-integrate the diverging AT&T System V and Berkeley CSRG system interfaces in what was known as *The 1984 /usr/group Standard* [1]. In 1985, the Technical Committee on Operating Systems – Standards Subcommittee (TCOS-SS) of the Institute of Electrical and Electronic Engineers (IEEE) began work under the auspices of the ANSI accredited IEEE Standards Board to build a formal *de jure* standard around the idea of source code portable applications with respect to the operating system service interface. By April 1986, an IEEE Trial-Use Standard was in place. The first standard was formally ratified in August 1988, as IEEE 1003.1-1988 (hereafter referred to as POSIX.1). It was at that time aligned with the developing ANSI accredited X3 work in X3J11 to develop the C-language standard, which became an ANSI standard in 1989[2]. The POSIX work was being carried forward into the ISO/IEC community by 1989 and work began through ISO/IEC JTC1 Subcommittee 22 (Programming Languages), Working Group 15 (POSIX) to guide the IEEE developed work through the ISO standards process. In 1990, POSIX.1 was aligned with the formally approved C-language standard and re-ratified as both IEEE 1003.1-1990 [3](an ANSI standard) and ISO/IEC 9945-1:1990. The only difference between the two documents was the white cover on the ISO/IEC document.

POSIX.1 covers just the system service applications programming interface (API). It is by its own scope and introduction a minimal standard to cover the essential system services. It was expected that further POSIX standards within the "family" would cover other functionality. And the IEEE POSIX work grew.

By 1990, there were 10 approved projects, and upwards of 300 people attending working group meetings quarterly. Work had begun on a Commands and Utilities standard (POSIX.2), test methodologies standards to address conformance testing issues (POSIX.3) along with test methods for the POSIX.1 work, real-time API (POSIX.4), work for POSIX.1 in Ada and FORTRAN77 (POSIX.5 and POSIX.9), and profiling work in the super computing domain (POSIX.10). Work continued to grow through the first half of the 1990s, such that at one point there were approximately 25 projects spread across about 16 working groups. Co-ordination committees sprung up around common themes (e.g. test methods, profiles, base API). Along the way, TCOS-SS became the Portable Applications Standards Committee (PASC) still under the auspices of the IEEE Standards Board.

Then participation began to fall off. This happened for a number of reasons. As some

standards documents completed, people no longer needed to attend those projects. Some documents were particularly contentious, and made little progress so interest waned in that work. The recession deepened, and participation continued to fall as employers cut back standards based expenditures. (UNISYS cut almost its entire standards organization at one point in its restructuring. This was the company that at one time had the greatest participation of the vendors at POSIX meetings). As of this writing (Spring 1999) the quarterly working groups have about 25-40 participants. Much of the work surrounds U.S. military-based requirements for additional real-time interfaces, and co-ordination activities to complete some well defined extensions into the base documents and work with the Open Group to ensure best use of resources and to ensure efforts aren't duplicated.

3. The [Unsung] POSIX Success Story

The POSIX standards efforts are often painted as a failure by vendor organizations wanting to escape the expensive yoke of standards conformance. As SDO do not have marketing organizations, it is often difficult for them to defend these accusations with counter-marketing. Concerns are raised with respect to the POSIX family of standards in such areas as:

- Falling participation (How can "POSIX" still be relevant?)
- Lack of ratified functionality (One can't do enough with just "POSIX" to be useful.)
- "POSIX" takes too long. (If "POSIX" doesn't hurry up, it won't be relevant.)

In each of these items, "POSIX" is placed in quotations because each of these supposed failures relies on a listener's ignorance or lack of context on what "POSIX" is under discussion.

Failing participation has been discussed. At some time, work on a project stops. It is either complete (a standard exists) or not. In either case the participants (modulo some maintenance function) go home.

The latter two issues are completely incorrect. The PASC working groups have produced on the order of 27 standards, many of them ratified through ISO, numbering in the thousands of pages over the past 14 years.

Of all the POSIX standards, several stand out.

- IEEE 1003.1-1990 == ISO/IEC 9945-1:1990
- IEEE 1003.2-1992 == ISO/IEC 9945-2:1993
- IEEE 1003.1b-1993 (POSIX Real-time)
- IEEE 1003.1c-1993 (POSIX Threads)

These have become a core set of functionality implemented on a wealth of operating systems.

POSIX has also always had powerful patrons on the customer side. The United States government took an early supportive view of the POSIX family of standards. The National Institute of Standards and Technology (NIST, formerly the National Bureau of Standards) contributed staff to the IEEE and ISO POSIX efforts from the beginning. As a major representative of the user community, able to literally put their money where their mouth was on procurement policy, they were able to act as a fundamental anchoring influence. They stood to greatly benefit from a successful standard in this space for applications portability, i.e. vendor independence. They sunk considerable effort as was the policy of those days into a certification process, and the attendant supportive standards work around test methods.

The United States government is the largest single organization procuring computer equipment in the world. Using the POSIX.1 Federal Information Processing Standard (FIPS) 151-2 [4] and its POSIX Certification Test Suite (PCTS), it acted as a huge impetus to developing POSIX.1 certified implementations. [1] At this point operating systems as diverse as Windows NT, Compaq/Digital's VMS, and IBM's OpenEdition MVS have been certified along with all mainstream versions of the UNIX operating system.

The core POSIX.1 standard is over ten years old (1988) along with its certification programme. It is widely implemented. Applications that have been ported at least once since 1989 with the advent of POSIX.1 and ISO/ANSI C have been generally modified to use the standard library

[1] It is important to note that NIST was never so foolish as to claim that a POSIX FIPS certificate was a guarantee of conformance to the standard. An operating system is far to complex to completely test, so the FIPS certificate merely states that the implementation (operating system under test) has run the PCTS, and the results have been verified and validated.

and system service interfaces. New applications written in C on UNIX systems are written to these standards.

The POSIX standards also form the core of the industry's Single UNIX Specification. POSIX is so successful at this point as to be invisible. It is taken for granted by the customer community that purchased platforms support a POSIX.1 system service interface and an ISO/ANSI C-language compiler will be available.

4. The Consortia Relationship

The relationship between the formal POSIX standards and UNIX industry consortia is a synergistic (if contentious) one. Individual consortia members and the consortia themselves attended IEEE and ISO POSIX working group meetings and were very active standards development participants. Once ratified, the consortia then adopted the formal standards, adding functionality where necessary to satisfy their own paying membership community. Both processes benefitted.

While the IEEE was taking on the task of developing POSIX in 1985 in North America, an industry consortium was formed in Europe amongst the UNIX vendors of the day to tackle the same problem of providing a basis for applications portability at the demands of the customers. The BISON group was formed (Bull, ICL, Siemens, Olivetti, and Nixdorf) which shortly thereafter became X/Open. The X/Open Portability Guide (XPG) was developed and largely described the System V Interface Definition (SVID).

X/Open developed a branding programme, whereby the vendor not only passes a rigorous test environment, but further warrants that they conform to the specification. If anyone proves they do not conform to the specification (regardless of their test suite results) the vendor is legally obligated to fix the implementation.

X/Open membership grew to include the North American and Far East vendors, and worked to attract the POSIX buying customer base. The only way to do this was to align with POSIX. By XPG, Issue 4 (1992) [5], the X/Open industry specifications were completely aligned with POSIX.1, POSIX.2, and ANSI/ISO C, and had publicly stated their commitment to continuing the alignment in the future.

In 1994, X/Open published the Single UNIX Specification as an update to XPG4, adding a wealth of traditional system service and library interfaces found on UNIX systems [6]. A UNIX 95 brand was created shortly thereafter. If one ignores the obvious duplication of interfaces (provided to support porting historical applications), there is an 80% overlap in the interface definitions between POSIX.1 and ANSI/ISO C. (This overlap continues between POSIX.2 and the X/Open Commands and Utilities specification in the Single UNIX Specification.)[7][8][9].

By 1998, X/Open and the Open Software Foundation (OSF) another fundamental but complementary UNIX industry consortium had merged into The Open Group, and the next version of the Single UNIX Specification was released with its attendent UNIX98 branding programme. The fundamental additions to the SUS were POSIX.1b (Real-time), POSIX.1c (threads), and additional work on symbolic link interfaces that are currently being added to an amendment of POSIX.1. While there was additional work added, the 80% overlap was still very much in effect.

Ratified and well implemented standards from the IEEE and ISO formal processes were incorporated into the core "UNIX" specification of the day. X/Open (and The Open Group) certainly contributed a great deal to the process of developing POSIX in the de jure processes, open to all, then adopted and incorporated the formal work into their specifications, adding work that was important to their paying membership community.

As the economic fall-out of the recession caused vendor membership in consortia to re-evalute their standards and specifications participation in both the de jure process as well as the consortia process, many players rankled at the apparent "double" cost to develop the specifications they ultimately required. To that end, all of the IEEE PASC POSIX working groups, the ISO/IEC POSIX WG15, and The Open Group's Base Working Group are looking at ways to ensure that there is no duplication of effort in this process. This process evaluation and evolution makes sense at this point in the history of the POSIX standards.

5. Why Does the POSIX De Jure Model Work

Despite the public complaints (from vendors) with the POSIX process being too slow, and standardizing old technology, a number of things about the formal POSIX model seem to have supported its success:

- The IEEE process allows any group of people to come together to form a standard. All it essentially takes is three people, a base document, and the desire to build a standard according to the formal rules. This does not mean it is easy to create a standard. It merely means that the process caters to any sized group willing to come to an agreement on a standard. More complex standards with larger circles of interest will obviously require greater consensus efforts to come to closure. The process scales itself.
- There is also a process by which standards that no one has modified or referenced in five years to be reaped as unused.
- Sponsorship of several projects that saw no progress within the working groups was withdrawn, i.e. the rules allow the body responsible for sponsorship to close unworkable projects.

At the same time, the early work of PASC (then TCOS-SS) was based on existing UNIX practice and experience, with base documents feeding the standards process that documented implemented proven ways of accomplishing the task. This is very similar to the Internet Engineering Task Force mandate for multiple non-descended reference implementations being required before an RFC can be considered complete. This requirement of a burden of implementation provides is incredibly useful.

- It provides a focus to the working groups discussions. With POSIX.1, POSIX.2, and the ANSI C work, consensus could be more easily driven around a customer perspective of whether or not existing applications would be broken if particular changes were made. Likewise, the working group members (both users and vendors) had a strong understanding of the problem domain being standardized.
- From the IETF perspective, it ensures that more than the idea's originator sees sufficient merit in the idea to attempt real economic work to develop the implementation.

POSIX also had a powerful patron in the U.S. government participation. While balloting groups within the IEEE require a certain balance of users and developers, it is often easy of lone individual users to be overwhelmed by the work load inherent in a large complex standards development project. Having a well funded knowledgeable user to solidly debate and balance large well funded vendor initiatives.

It should not be easy to create a standard in a contensious space. Having the entire SDO process go through a revolution to allow this to happen, or worse yet to replace SDOs with industry consortia with no obligation to include any voices other than their own members will break the process of developing good standards.

It is a matter of good government that most democratic governments are based upon a two house system of one form or another, whereby there are checks and balances in the process to ensure that special interest groups and lobbyists cannot "buy" or drive their way to a law quickly. A government is simply unable to quickly and easily create law. (Indeed any Canadians may remember the outrage at Brian Mulroney's conservative government that managed to "stack" the senate, Canada's upper house, such that he could drive through certain unpopular tax legislation.)

So should it be with standards development. It should not be easy in the face of controversy to force a standard. The present desire to radically change the various formal standards processes to create more standards faster is economically unsound. The economics of standards is such that standards documents provide a contract for how interfaces work such that large diverse groups can provide and use implementations of the domain (whether it is computer technology or film in a camera). This supports a commodity economic view of the world encouraging competition. Standards will be developed wherever there exists sufficient need, by participants willing to put their money where their mouth is to do work. These processes should be open to all knowledgeable participants without undue burden (e.g. high membership fees).

POSIX was developed in such a process. Despite marketing to the effect that POSIX is past due, old technology, and underdeveloped, the reality is the process has led to the

development of a set of formal specifications, based upon proven ways to solve a set of problems in applications portability, that have weathered a decade of implementation and use.

References

1. The 1984 /usr/group Standard, published by /usr/group, Santa Clara, California, USA.
2. ISO/IEC 9899:1990, Programming Languages—C,
3. ISO/IEC 9945-1:1990, Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) [C Language], IEEE Standards, NJ, ISBN 1-55937-061-0
4. National Institute of Standards and Technology, Federal Information Processing Standards Publication 151-2, *Portable Operating System Interface (POSIX) - System Application Program Interface [C Language*], 12 May, 1993.
5. X/Open CAE Specification:

- System Interfaces and Headers, Issue 4, Release 2
- Release 2 Commands and Utilities, Issue 4,
- System Interface Definitions, Issue 4, Release 2

   X/Open Company Ltd., 1994
6. Walli, Stephen R., Go Solo: How to Implement and Go Solo with the Single UNIX Specification, Prentice Hall, Englewood Cliffs, NJ, 1995
7. ISO/IEC 9945-2:1993, Information Technology — Portable Operating System Interface (POSIX) — Part 2: Shell and Utilities, IEEE Standards, NJ, ISBN 1-55937-255-9
8. JTC1/SC22/WG15 N622, The Relationship between the X/Open SUS & ISO POSIX, Source: X/Open, 1995-10-01
9. JTC1/SC22/WG15 N639, U.S. Position on Single UNIX Specification, Source: U.S. Member Body, 1995-04-17