The value of open standards and open-source software in government environments

K. D. Simon

Among the most noteworthy topics surrounding the recent widespread adoption of open-source software (OSS) are the convergence by governments worldwide to open standards and the ways in which open source embraces this convergence. There are continuing debates over the future of software and, in particular, the competition between OSS and proprietary software. Many studies by governments and by information technology analysts suggest that OSS and open standards are intimately connected and that the inherent value of open-source adoption may be attributable in large part to the embodiment of open standards in OSS. The government environment is changing rapidly in areas as diverse as homeland security and social services. Given the equally rapid changes in the information technology marketplace, the successful adoption of these new technologies by governments will depend on how well the strengths of proprietary software and OSS are understood and applied—especially with respect to the use of open standards to speed deployments of integrated capabilities that respond to emerging challenges. This paper evaluates the relative strengths of proprietary software and OSS as development techniques that embrace the open standards valued by governments.

Among the most noteworthy topics surrounding the recent widespread adoption of open-source software (OSS)[1] are the convergence by governments worldwide to open standards[2] and the ways in which open source embraces this convergence. Many studies by governments and by information technology (IT) analysts indicate that OSS and open standards are inherently valuable. For example, a 2003 study conducted by the e-Cology Corporation reported that over 70 policy proposals, position statements, or actual government decisions had been made concerning OSS in some 24 countries around the world.[3] The study also found that open source is a good fit for current IT environments with respect to each of the following criteria: data integration, IT architecture, standards compatibility, and coexistence with commercial applications. Furthermore, the study revealed that open source embodies open standards critical to technology advancement and to flexibility of implementation.

Because of the rapid changes in the IT marketplace, the success of governments in applying these technologies will depend on how well the capabilities of proprietary software and OSS are understood and applied. This paper will evaluate the relative strengths of proprietary software and OSS as development techniques that embrace the open standards valued by governments. The first section discusses the benefits of using open standards in governments and describes several prominent open standards valued by governments today. The second section explores the weaknesses and strengths of proprietary software and OSS. The third section provides examples of OSS in governments from various regions of the world and discusses how open standards have emerged from OSS. The paper concludes with a summary of our findings.

WHY OPEN STANDARDS FOR GOVERNMENTS?

Governments worldwide are considering open standards and discovering, in many cases, that the standards offer several significant benefits. In fact, the benefits are so significant that many government agencies have already adopted or are currently considering adopting policies that require adherence to open standards. For example, the state of Massachusetts now gives "preference to open-source software and products that adhere to open standards such as Extensible Markup Language (XML) and Secure Sockets Layer (SSL)."[4]

Then, too, the acceptance and adoption of open standards by major commercial interests such as IBM, Hewlett-Packard, and Oracle have created a growing assortment of open-standard-enabled products that have accelerated the comfort level of governments toward the adoption of these standards. Many of the products involved are either open source, contain open-source modules, or at least interoperate seamlessly with common open-source software such as Linux** and the Apache** Web server. These companies and related systems integrators not only supply such products to governments, but also provide the support that allows for a relatively easy transition to the use of open source and open standards. Governments are also currently engaging outside services, including consultants, non-government organizations, and industry associations, to examine the relevance of open standards and to address issues that would maximize benefits for these governments. Whether governments move aggressively or slowly toward open standards adoption, they have, at the very least, begun to document the benefits open standards can bring to their citizens and business constituents. We now briefly examine a few of the most compelling benefits of using open standards in governments. We also discuss several open standards that are valued by governments today.

Interoperability

Governments consume, manage, and produce massive amounts of information, which in turn delivers the greatest value when it is readily accessible to citizens, businesses, and government counterparts. As IT continues to evolve into ever more complex and globally interconnected networks, governments are under constant pressure to provide efficient and reliable services—including those supporting military operations, trade and travel security, social services delivery, and a range of citizen services. Increasingly, these services cross departmental and even national boundaries. Thus, interoperability is needed vertically and horizontally, within and across all organizational and administrative boundaries. The most challenging areas of system interoperability are found in software. Open standards form the essential framework needed to position government agencies to overcome these challenges. Moreover, the interoperability delivered by open standards is independent of whether OSS or proprietary software is used to implement these standards.

A good example that highlights the importance of interoperability involves e-government services, which have gained momentum in recent years. In using e-government services, citizens and enterprises must be allowed to seamlessly exchange information electronically both within and beyond local boundaries. Citizens and enterprises should never have to waste time trying to comprehend the intricacies of different government agencies in order to complete simple transactions. Governments in Europe, for instance, are insisting that e-government services be interoperable, and open standards are crucial in allowing such services to work together within enterprise systems.[5] Indeed, Manuela Finetti, head of the Interchange of Data between Administrations (IDA) unit for the European Commission, has stated that agreement upon common standards and specifications is necessary to bring administrations within the reach of citizens and enterprises and to support required services.[6]

Vendor lock-in

Government IT buyers and commercial suppliers have both learned that when governments implement core systems that do not use open standards, the supplier chosen for the initial system frequently has a competitive price advantage in subsequent procurements for system expansion. In the worst cases, deployment of solutions that cannot be used with any other vendor immediately locks governments into doing business with one particular vendor. This ultimately forces governments to take high financial risks. The total cost of ownership for vendor products and services may be cheap initially, but is likely to increase as more customizations are needed. Governments typically struggle to operate on limited budgets and simply cannot afford to pay inflated prices for software solutions. In addition, many governments store massive amounts of data, for example, birth certificates and tax returns, as digital files and are reluctant to store such files in a format supported by only a single or even a handful of vendors.[7] A single-vendor product with a specific data format is not accessible for use by each and every citizen and enterprise within governments. Thus, an important requirement is that data must be readily accessible to citizens and enterprises that, in turn, use various products with many different data formats.

Governments are turning to open standards to alleviate the problems of vendor lock-in. The Linux operating system, frequently adopted as a de facto standard, continues to be deployed rapidly in governments and was noted as the fastest growing server and database platform in the first quarter of 2004.[8] For instance, more than 200 IBM government customers from around the world have adopted Linux.[9] In September 2004, Brazil signed an agreement with IBM to open the Knowledge and Technology Center at the University of Brasilia.[10] The center was created to promote the usage of Linux and other open standards. One of the significant reasons for opening the center was to avoid being locked into one vendor and instead to allow freedom of software choice and innovation with lower costs. Additional initiatives are being developed worldwide to avoid vendor lock-in. The Chinese Ministry of Science and Technology (MOST) and the French Atomic Energy Commission (CEA), in association with Bull and STMicroelectronics (ST), formed the Chinese Open Platform Initiative in October 2004.[11] Compatible servers, computer appliances, and mobile systems will be developed under this new initiative, lowering costs and allowing collaborative improvements to ensure that solutions can be provided throughout the world. ST, China, France, and other partners are currently working together to develop a new Linux-based platform to achieve these goals.

■ Many government agencies have already adopted or are currently considering adopting policies that require adherence to open standards ■

Flexibility

The environment in which governments operate is changing more rapidly than ever before. To succeed in this environment, governments must be more flexible and responsive than was previously the case. Nowhere is this seen more clearly than in the global response to terrorism, where every piece of insight gained by government demands immediate response. The dynamic IT marketplace, which offers endless new technologies and expanding services, can help governments make more informed decisions faster. One result is that governments are increasingly conducting operations via the Internet. (Indeed, the Internet itself is a prime illustration of the value of open standards. It simply would not function in the absence of documented standards, such as HTML [Hypertext Markup Language], that have been universally adopted by suppliers and users.) In anticipation of unforeseen situations such as natural disasters or terrorist attacks, governments must position their operations and the resources that support them—including raw materials, information, and personnel—to respond quickly, appropriately, and without excessive costs. To ensure efficient operations, databases must be integrated, and information systems must be consolidated. The unpredictable nature of these emergencies requires flexibility.

Governments are not just talking about a need for flexibility. They are taking actions to ensure flexibility in their procurement processes. A recent article on Web services highlighted the following achievements by governments worldwide:[12]

- The Belgian Federal Government launched a Web-based e-procurement system to replace its paper-based public acquisitions procedures. The Joint Electronic Public Procurement project creates a “network of portals” for a whole-of-government electronic-tendering process.
- The state of Massachusetts’ Executive Office of Health and Human Services deployed a new Web-based system called MassCARES that helps caseworkers and beneficiaries locate resources, determine program eligibility, and coordinate the flow of information among the many agencies acting in shared cases.
- The Australian Taxation Office created the Australian Business Register, a company registry interoperating with all federal, state, and local agencies that serve and regulate the business community, making it easier, faster, and less costly for business to deal with government.
- By 2005, the UK government expects all its departments to offer their services electronically. They launched a project—Government Gateway—that helps central and local governments and devolved administrations get services online faster.

Although each of the preceding government actions is quite different, the common goal is to merge and evolve organizations that are able to provide efficient, flexible, electronic services equal to those provided by private enterprises. Open standards directly support the ability of governments to achieve these results quickly and at minimal cost, results that are certainly valued by governments. For example, the Department of Technology for the Government of Punjab in India has a policy which assures that acquisitions have required flexibility, a policy that is aligned with their strategic direction through the use of open standards.[13] In September 2004, a study by the Center for Strategic and International Studies found that government agencies worldwide have approved 90 initiatives that embrace OSS,[14] a trend which can be expected to continue.

Prominent open standards valued by governments

Shown in *Figure 1* are prominent open standards valued by governments today that demonstrate the claim that open standards provide significant benefits.[15] In particular, these open standards provide governments with solutions that range from interchanging vocabularies via the Web to defining strategies for business. Due to its widespread adoption, HTML is perhaps one of the most recognized open standards among governments and for the IT industry as a whole. Indeed, HTML is used practically everywhere on the World Wide Web. By using open standards such as HTML, governments will no longer suffer from a lack of flexibility or interoperability of their Web services. Moreover, by supporting open standards a government reduces the likelihood of vendor lock-in.

The benefits from using HTML have been demonstrated on such a large scale that other open standards, such as XML,[16] have been based upon HTML. Both a study conducted in 1999 by the U.S. National Association of State Chief Information Officers and a paper published in 2000 by the U.S. National Electronic Commerce Coordinating Council (NECCC) noted XML as a promising open standard for seamless sharing of information among governments.[17] Because XML specifications continue to be developed, governments have been actively involved in ensuring that XML formats for exchanging data remain consistent. Many governments have either launched projects to promote open XML formats or switched to the OpenOffice XML format.[18] OpenOffice is itself an open-source alternative to Microsoft** Office and is currently gaining popularity among governments in Germany, France, China, Brazil, and the U.S.[19]

One such project recently launched by governments is the Electronic Business Extensible Markup Language (ebXML**). Established by the United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) and the Organization for the Advancement of Structured Information Standards (OASIS), ebXML is a set of specifications for agencies that conduct transactions electronically, including procurement and other transactions. Governments in Norway and Finland are currently working on implementations.[20]

Many other open standards, such as Transmission Control Protocol/Internet Protocol (TCP/IP), Hypertext Transfer Protocol (HTTP), Java** 2 Enterprise Edition (J2EE**), and structured query language (SQL), also demonstrate the benefits mentioned here. What is inherently clear is that the more accepted open standards like HTML and XML become, the more governments can benefit as they move toward seamless e-government services.

SOFTWARE DEVELOPMENT TECHNIQUES: PROPRIETARY SOFTWARE VERSUS OSS

In today's IT environment, one of the key factors to successful solution deployment by governments is the assessment of which type of software—proprietary or open source—is a better fit for a given situation. Proprietary software and OSS are differentiated both by their philosophies and by their development methodologies. Development techniques are the foundations of software solutions and can clearly show the strengths and weaknesses of these two approaches. Thus, examination of the development techniques used for proprietary software and for OSS is a good starting point for deciding the best type of software to use in a specific government deployment.

The goal of this section is to review the distinguishing features of the development techniques for proprietary software and OSS that would allow informed decisions to be made about the best type of software for a government deployment. First, the respective development philosophies will be discussed briefly. Next, the development techniques of proprietary software and OSS will be outlined according to the typical sequence of the software development process: design, programming, and testing and maintenance. A comparison of the two techniques will then be provided.

Philosophies of proprietary software and OSS

To understand the differences between the development techniques for proprietary software and OSS, one must first understand the philosophies on which these approaches are based. On the one hand, the development technique for proprietary software is based on a philosophy that users of a software program only need to understand what the software does, not how those results are achieved. Some people think of this as a philosophy based on obscurity. The obscurity of proprietary software is implemented through closed-source techniques. Closed source means that the licensee or user of software does not receive and cannot view the software *source code*, the form of the software that a human can read and understand. Moreover conditions of use, sharing, or possible modifications of the software are typically restricted by copyright law. Closed-source software is delivered only in a form (known as *binary code*) that computers can understand and execute. A closed-source approach keeps a vendor's services and products unique, and frequently protects profitability and trade secrets by preventing competitors from viewing, duplicating, or using a program's source code. The ultimate goal for a vendor of proprietary software is to achieve a sustainable business that protects intellectual property, supports continued investment, and delivers value that keeps customers coming back. Most of the advancements in software from the dawn of the IT industry have emerged using this technique.

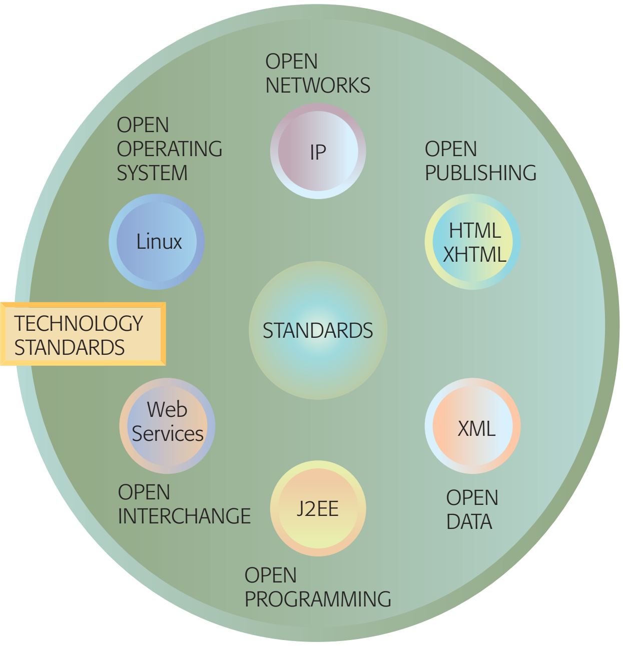

Figure 1
Open standards valued by governments

On the other hand, the development techniques of OSS stem from a philosophy of openness. Most licensing agreements in use for OSS today allow anyone to freely view, share, use, and modify a program's source code. Indeed, the availability and open presentation of source code is the origin of the term OSS. This openness allows anyone technically knowledgeable to understand both what the software does and how it functions. Note that "freely" does not mean cost-free, but rather refers to freedom of use. OSS does require time and money to develop and maintain, especially in commercial environments. However, the open-source approach allows information exchange and creates a level playing field for software producers. Commercial players in the OSS arena rarely see the software itself as the key to achieving business results. Rather they consider the services, integration, and product sales that accompany the OSS as their business. Indeed, some open-source authors derive significant income from the sale of services related to their free handiwork. The paramount objective here is not to achieve profit for vendors, but to build offerings on a base of high-quality software that is available and beneficial to all users.

■ Governments are turning to open standards to alleviate the problems of vendor lock-in ■

Although the philosophies of proprietary software and OSS differ, this is not to say that one philosophy is better than the other. Proprietary software has been and remains the essential means of advancing software technology on the leading edge of innovation. Its ability to attract investment and produce profit from software licenses generates economic growth and creates lucrative employment opportunities.[21] This is certainly evident for software companies such as IBM, Microsoft, and Oracle. On the other hand, OSS grants users the flexibility to customize software solutions. While OSS has produced many software breakthroughs, its strength lies in the delivery of widely deployed, stable tools that promote the goals of flexibility and interoperability discussed earlier. Users of OSS reap benefits such as low costs and avoidance of vendor lock-in.

Of importance here is an acceptance that the philosophies of proprietary software and OSS are different, as well as a knowledge of how each philosophy affects the environments in which proprietary software and OSS are deployed. Proprietary software and OSS can each be used in appropriate environments to achieve desired results, both for the commercial entities that promote them and, more importantly, for the public- and private-sector entities that use them.

Techniques used in a typical software development process

Eric Raymond's highly acclaimed work *The Cathedral and the Bazaar* separates the development techniques of proprietary software and OSS into two categories: "cathedral-like" and "bazaar-like."[22] Analogous to the meticulous construction of cathedrals in the Middle Ages, the development of proprietary software is depicted as being very controlled in terms of design and implementation. In contrast, the development of OSS is like a bazaar in which everyone can choose what he or she desires. Here, communication is informal, and coding is often carried out by volunteers in their spare time.

Indeed, proprietary software and OSS have distinguishing development techniques throughout the software development process, a process which typically consists of at least the steps of design, programming, and testing and maintenance. We will now take a closer look at how each of these development techniques carries out these steps, with the goal of highlighting the characteristics of each technique that determine its respective suitability for government environments.

Design

Proprietary software is designed with a focus on integration.[23] The intention is that customers will select proprietary software because a variety of capabilities are bundled into each product, and multiple compatible products are then bundled into suites. The components of proprietary software are very tightly coupled. For example, a customer who purchases a server-based piece of software from a vendor will likely also purchase client-based tools from the same vendor. Next, the customer may purchase a subscription, upgrades, and related offerings. Each component purchased is typically characteristic of a specific vendor and only operates with other components that are built and supplied by that particular vendor. In addition to providing extended function, integration is used to produce products with better performance. Fewer resources are necessary, and software components work in tandem. This technique contributes greatly to the way in which proprietary software delivers leading-edge capabilities in the marketplace, while generating sustainable revenue and profit streams for its producers. As mentioned earlier, profit provides the benefits of company growth, employment, and investment. In fact, Jack Messman, Novell chairman and chief executive, stated in a September 2004 article that "the industry needs the profits from proprietary software to help fund open-source development."[24] Indeed, proprietary software companies such as IBM, Hewlett-Packard, and Novell (which once primarily produced proprietary software but now also owns SUSE LINUX) also employ people to help produce OSS projects based, for example, on the Apache Web server.

In contrast to proprietary software, OSS is designed to be modular. Linus Torvalds, founder of Linux, has strongly advocated modularity with the statement, "What is needed is as modular a system as possible."[23] Modularity allows components to be loosely coupled or interchanged independent of one another. A user can select many different, small components to form an OSS solution. The components are not directly linked to one provider or developer. Instead, users can obtain these components from various sources located all over the world. There have been studies released to support the modularity of OSS. For example, one study reveals that by implementing modularity the GNU/Linux project achieved the benefits of fast development, reuse of modules, innovation through project competition, and recombination of modules.[25] Modularity is one of the primary reasons OSS is closely connected to open standards. The publication, use, and widespread adoption of open standards allow this modularity to be achieved. Almost as a by-product, the same standards that OSS programmers use can be reused by organizations that deploy OSS to produce interoperation and information sharing.

Programming

The OSS technique involves programming in many parallel efforts that typically do not adhere to strict schedules or market pressures.[26] OSS developers are frequently volunteers with a passion for a particular type of software, who complete coding as time allows. Many efforts are duplicates of one another—inducing collaboration and experimentation with ideas and approaches. The OSS approach also allows users to pick and choose the solutions that they like best. Over time, the universe of users selects the best capability from available implementations, in contrast to the proprietary technique of having a centralized design group make these choices. Solutions are coded in small modules that provide a limited number of features. Such code modules are produced rapidly, and projects are often created by as few as one or two developers at a time. A 2002 study of the top 100 mature products on SourceForge**, an open-source software-development Web site, revealed that "the vast majority of mature OSS programs are developed by a small number of individuals."[27] A downside, however, is that documentation is typically scarce, outdated, or nonexistent.[28] Because developers are not obligated to write documentation, which is often a tedious process, they tend to code solutions without producing adequate documentation.

In contrast, proprietary software is usually programmed by designated teams of developers who follow specific schedules. These schedules are controlled by management and influenced by market pressures.[26] Duplicate efforts are avoided or discouraged to minimize inefficient use of resources. Moreover, there is little pressure to constantly migrate to the newest programming languages and development environments, a very common occurrence in OSS development. Projects usually satisfy many specific requirements that can be time-consuming to refine and program. Thus, proprietary software is often implemented in longer cycles than OSS. Also, documentation is typically extensive due to customer requirements.

Testing and maintenance

The testing technique used in proprietary software development usually involves both internal and beta testing,[23] which together improve the quality of the software by discovering bugs that can be fixed before the software is released to the public. Although problems are detected and fixed by this method, the software configurations tested are limited to the environments chosen by the testers. Therefore, the fixes incorporated may not always be optimal for all environments. In the case of environments not tested, users must often wait until a vendor releases a new version to obtain necessary fixes. Then too, bugs may never be discovered and fixed because the source is not public. However, proprietary software vendors are often reliable in helping users with maintenance and support.

Unlike proprietary software, OSS is made available to the public on such a frequent basis that developers and users can test the software well before a final release. This often creates many more opportunities to eliminate bugs and incorporate improvements than is the case for proprietary software. In addition, software can be tested immediately in real environments with a wide variety of configurations. Users are not forced to wait for a new release because they have the option to fix the software themselves. A user who fixes a problem can then submit a patch[29] to the developers. This encourages extensive collaboration to fix bugs the best way possible. In contrast, though, finding a reliable source of maintenance and support may at times be a challenging proposition, because the developers who contribute to OSS are typically scattered all over the world. Users must also be careful in implementing methodologies for upgrading their environments to avoid being overwhelmed by the high frequency of OSS software releases.

■ It is important to recognize that neither proprietary software nor OSS is a panacea ■

Because of these differences between the techniques of proprietary software and OSS, many studies captured in a popular paper by David Wheeler suggest that OSS is more secure than proprietary software.[30] For example, the Summer 2004 Linux Development Survey by Evans Data found that "92% of their Linux systems have never been infected with a virus and 78% of their Linux systems have never been cracked."[30] Perhaps Eric Raymond sums up the technique of OSS best with the statement, "Given enough eyeballs, all bugs become shallow."[22]

Based on the characteristics outlined above, proprietary software excels at allowing rapid innovation and advancing emerging areas of technology due to requirements for meeting schedule deadlines, market pressures, and a constant push for profits. The profit gained from proprietary software increases company growth, investments, and employment. For example, Dassault Systèmes and IBM announced the first release of their 64-bit-enabled Product Lifecycle Management (PLM) applications for the IBM AIX* 5L* operating system in October 2004. Per Larsen, vice president of IBM eServer* pSeries* Marketing, Systems and Technology Group, stated, "This achievement is a great illustration of how IBM advanced technology combined with leading-edge software developed by Dassault Systèmes contributes to protecting PLM customers' investments."[31]

However, once a technology area becomes saturated with unique proprietary solutions, innovation becomes more difficult. OSS, which is based on large-scale collaboration, levels the playing field and sparks innovation that drives software compatibility and interoperability. A survey in February 2004 found that 1.1 million developers in North America alone were working on OSS projects.[30] With so many developers collaborating of their own free will, innovation is stimulated and grows. This innovation allows solutions to be obtained from a wide variety of sources and promotes interoperability and flexibility. It is important to recognize that neither proprietary software nor OSS is a panacea. Environments determine which type of software is better on a case-by-case basis. However, within government environments where open standards are valued, it can be argued that OSS is a superior and more appropriate development technique, as discussed in the following section.

OSS: A SUPERIOR SOFTWARE DEVELOPMENT TECHNIQUE

Both proprietary software and OSS are necessary to continue improvements and advances in the IT industry. However, for government environments OSS has proven to provide a superior development philosophy and development technique in the sense that it embraces the open standards valued by governments. Through the OSS development technique, solutions have been produced that are often better suited and of higher quality than corresponding proprietary software[30]—especially in government environments. Despite its disadvantages, OSS has provided interoperable, flexible, and secure solutions for governments. One successful application of this development technique is certainly Linux.

In this section, examples of OSS in governments from various regions around the world will be presented. Next, the emergence of open standards from OSS will be discussed, including the key characteristics that make OSS a superior development technique in such instances.

Examples of OSS in governments

OSS has certainly drawn interest from governments worldwide that are eager to reap the benefits of open standards, as discussed in the earlier sections of this paper. The following are some examples of OSS adoption in governments from regions around the world.

- *Australia*—A panel of ten contracted suppliers was formed this year to directly service Australian government agencies that choose to adopt Linux systems. The two-year contract has an option to extend for two additional one-year terms.[32]
- *Brazil*—Government officials in Brazil have committed "to exporting around $2 billion worth in software every year, to replacing Windows with Linux in 300,000 federal computers, to transferring $1 billion from the Telecommunications Fund to the free-software-based Digital Communications System, and to network the country's 200,000 public schools using open source."[33] This migration will make Brazil the largest public sector user of open source in South America.
- *Denmark*—The Ministry of Finance in Denmark has implemented an open-source project to simplify data exchange between systems. The system uses the open-source application server JBoss** running on Red Hat** Linux.[34]
- *Germany*—Despite concerns of possible software patent infringements raised in the debate over new European Union patent legislation, the government of Munich decided to migrate 14,000 desktops to the open-source Linux platform. The migration is one of the largest moves ever from proprietary software to open source.[35,36] Clearly, this example shows that the growing popularity of OSS is a threat to some established commercial interests. Any backlash from this threat can potentially impact the growth of OSS.
- *Malaysia*—Malaysia is promoting open-source software such as Linux to increase IT skills, gain information security through code inspection, and gain bargaining advantages with proprietary software vendors.[37]
- *Philippines*—The Advanced Science and Technology Institute (ASTI) government agency has advocated the use of open-source software in the Philippines and is developing versions of the Bayanihan Linux software, which is currently being used by the University of the Philippines.[38]
- *South Africa*—South Africa has adopted an official policy promoting the use of open-source software. A preference is now given to open-source applications when proprietary alternatives do not offer a compelling advantage.[39]
- *Sweden*—Statskontoret, the Swedish Agency for Public Management, released a feasibility study to promote OSS in public administrations. The study found that OSS is equal to or better than proprietary software is many cases.[40]
- *United States*—In a recent article, government agencies have "implemented open-source solutions that range from Linux-running data-collection computers on Naval Oceanographic Office survey ships to a Web-based tool that allows the U.S. Agency for International Development (USAID) to quickly process the visas of foreign workers scheduled to train in the United States."[41]

How open standards have emerged from OSS

Many software capabilities began as OSS solutions. Over time, these OSS solutions blossomed into open standards that are positively affecting governments worldwide today. To further see the advantages of the OSS development technique, one must understand how open standards have emerged from OSS. Evidence shows that open standards have emerged due to at least three significant factors:

1. Development of a high-quality product that became trusted by many users
2. Competition
3. Demand from users

An example that supports this hypothesis is sendmail, the de facto standard program for Internet mail transfer. Sendmail was developed by Eric Allman in 1981 at the University of California, Berkeley during a period of proliferation of networks and e-mail protocols.[42] At that time the Internet had not yet standardized on the TCP/IP protocol. Therefore, e-mail became difficult to transport across networks because of different protocols. Sendmail was the first software of its kind due to the unique method it used to complete mail transfers. Instead of rejecting incompatible mail from different networks, sendmail modified the mail so that it could be transported to its destination.

With the explosion of e-mail, interest in sendmail grew rapidly. In 1986, Allman ceased his development of sendmail. However, other developers took up enhancement of sendmail's capabilities and released two additional versions of sendmail. Vendors also had their own versions of sendmail with special features including, for example, the Sun Microsystems version which added Network Information System (NIS) support. In 1989, Allman began development of his version of sendmail again. The additional versions of sendmail created friendly competition, which enabled Allman to incorporate some excellent features from these other versions, such as external databases and NIS support. In addition, because many users trusted and used the capabilities of sendmail, they began to demand that specific features be added. In fact, at one point users were requesting three to five new features per week.[42] As a result of trust, competition, user demand, and facilitation by the development technique of OSS, sendmail emerged as the standard for mail transfer. Thus, both the needs of customers and the abilities of developers, and not solely the output of proprietary vendors, shaped the Internet into its current success.

There are several additional OSS solutions, such as Linux, the Apache Web server, Berkeley Internet Name Domain (BIND), and Mozilla**, that have been as successful as sendmail. All have become open standards with the help of the superior OSS development technique, and all support the hypothesis presented here.

CONCLUSION

As the widespread adoption of OSS continues, governments worldwide are increasingly converging to open standards to reap significant benefits, namely interoperability, flexibility, and avoidance of vendor lock-in. This paper has discussed these benefits, as well as prominent open standards that provide such value for governments today. E-government services that must operate seamlessly are gaining momentum among government agencies. To ensure successful operations, governments must be able to determine whether proprietary software or OSS better suits their particular environments.

The development techniques of proprietary software and OSS, including weaknesses and strengths, were outlined to provide a better understanding of how each would suit a given environment. Based on these weaknesses and strengths, evidence and examples were presented that demonstrate how OSS can be a superior development technique for government environments. In addition to the many proprietary software solutions currently deployed in government applications, OSS and open standards are having major, positive impacts on governments in various regions around the world.

It is also important to understand how many open standards have emerged from OSS, facilitated by the development technique and philosophy of OSS, because of the key factors of trust, competition, and user demand. These factors are, in turn, undoubtedly linked to the needs of customers and the abilities of producers. As the IT marketplace continues to evolve, these powerful factors, coupled with the superior development technique of OSS, will not only enable governmental environments to efficiently serve citizens and enterprises, but will also help drive the IT industry in directions which do not rely solely on proprietary vendors.

ACKNOWLEDGEMENTS

Very special thanks to Bradley L. Westpfahl, without whom this paper would not have been possible, for sharing great insight into government industry programs and interests in open source and open standards. The author would also like to thank the following people for their cooperation and support during the creation of this paper: Gary Wright, Sandra K. Johnson, Pamela Sclafani, Sheila Harnett, Angela Allen, Tom Clarke, Christine Lorenz, Danna Rother, Reggie Hill, Sharal Brown, Martin Forte, Tim Sheehy, and Andrea Cotter.

*Trademark or registered trademark of International Business Machines Corporation.

**Trademark or registered trademark of JBoss Inc., Linus Torvalds, Microsoft Corporation, OASIS, Red Hat, Inc., Sun Microsystems, Inc., SUSE AG, A Novell Company, The Apache Software Foundation, The Mozilla Foundation, or VA Linux Systems, Inc.

CITED REFERENCES

1. There is no universally accepted definition for the term “open source.” For the purposes of this paper we use the definition found in *The Open Source Definition*, The Open Source Initiative, http://www.opensource.org/docs/definition.php.

2. Throughout this paper, the term “open standard” is used as defined in *Wikipedia*™, The Wikimedia Foundation, http://en.wikipedia.org/wiki/Open_standards.

3. *Open Source Software in Canada*: *A Collaborative Fact Finding Study*, e-Cology Corporation (September, 2003), http://www.e-cology.ca/canfloss/report/CANfloss_Report.pdf.

4. D. Becker, *Proprietary Software—Banned in Boston?* CNET Networks, Inc. (September 30, 2003), http://news.zdnet.com/2100-3513_22-5084442.html?tag=nl.

5. M. Broersma, *Europe to Push for Open Source*, CNET Networks, Inc. (July 21, 2003), http://news.zdnet.com/2100-3513_22-5047543.html.

6. M. Finetti, *Interoperability Matters*, PSCA International Ltd. (2003), http://www.publicservice.co.uk/pdf/europe/autumn2003/manuelaf.pdf.

7. *Microsoft at the Power Point*, The Economist Newspaper Limited (September 11, 2003), http://www.economist.com/business/displayStory.cfm?story_id=2054746.

8. P. Hochmuth, *Linux Stays Hot as an Overall Server and Database Platform*, Network World, Inc. (June 9, 2004), http://www.nwfusion.com/newsletters/linux/2004/0607linux2.html?fsrc=rss-linux.

9. *Linux Can Work Wonders for India*, Sify Ltd. (September 9, 2004), http://sify.com/finance/fullstory.php?id=13563101.

10. J. Loftus, *Brazil, Meet Linux*, TechTarget, Inc. (September 13, 2004), http://searchenterpriselinux.techtarget.com/qna/0,289202,sid39_gci1005633,00.html.

11. *STMicroelectronics to Participate in China’s Open-Source Computing Initiative*, STMicroelectronics (October 11, 2004), http://www.st.com/stonline/press/news/year2004/t1534h.htm.

12. *Web Services—The Way to Go*, BizHat.com (March 17, 2004), http://forums.bizhat.com/post-5863.html.

13. *Procurement Policy*, Department of Information Technology, Government of Punjab, India, http://www.doitpunjab.gov.in/draft_procurement.htm.

14. S. Shankland, *Study Shows Governments’ Open-Source Embrace*, CNET Networks, Inc. (September 2, 2004), http://news.com.com/Study+shows+governments’+open-source+embrace/2100-7344_3-5345709.html.

15. J. B. Stallings, *Innovation with Standards: It Takes a Community*, IBM Corporation (December 2004), unpublished.

16. The Microsoft “embrace and extend” strategy to encumber XML with proprietary extensions has been widely discussed. See, for example, *MS Open-Source Move Is Straight from Playbook*, Ziff-Davis Media, Inc., http://www.eweek.com/article2/0,1759,1563644,00.asp.

17. *An Introduction to XML’s Potential Use within Government*, National Electronic Commerce Coordinating Council (December 2000), http://www.ec3.org/Downloads/2000/XML_Paper.pdf.

18. M. Broersma, *Denmark Urges Government Support for Open Source*, CNET Networks, Inc. (October 24, 2003), http://www.zdnet.com.au/news/software/0,2000061733,20280102,00.htm.

19. S. M. Kerner, *OpenOffice Finds Sweet Spot with Governments*, Jupitermedia Corporation (January 1, 2004), http://www.internetnews.com/dev-news/article.php/3294431.

20. S. P. McCarthy, *Internaut*: *XML Extensions Ease Sharing of Business Data*, Government Computer News (October 11, 2004), http://www.gcn.com/23_30/tech-infrastructure/27563-1.html.

21. J. Carroll, *In Defense of Proprietary Software*, CNET Networks, Inc. (December 16, 2003), http://news.zdnet.com/2100-3513_22-5125160.html.

22. E. S. Raymond, *The Cathedral and the Bazaar*, (December 6, 2004), http://www.catb.org/∼esr/writings/cathedral-bazaar/.

23. A. Gok, *Open Source versus Proprietary Software*: *An Economic Perspective*, presented at Free Software and Open-Source Days/2004, Istanbul Bilgi University, Istanbul, Turkey, February 27–29, 2004, http://open.bilgi.edu.tr/freedays/papers/Abdullah_Gok.pdf.

24. A. Donoghue, *Novell*: *The Future is ‘Both-Source*,’ CNET Networks UK Ltd. (September 14, 2004), http://uk.builder.com/manage/work/0,39026594,39217373,00.htm.

25. A. Narduzzo and A. Rossi, *Modularity in Action*: *GNU/Linux and Free/Open Source Software Development Model Unleashed*,” Free/Open Source Research Community (May 12, 2003), http://opensource.mit.edu/papers/narduzzorossi.pdf.

26. J. Carroll, *Open Source vs Proprietary*: *Both Have Advantages*, CNET Networks, Inc. (May 27, 2004), http://www.zdnet.com.au/news/software/0,2000061733,39148762,00.htm.

27. S. Krishnamurthy, “Cave or Community? An Empirical Examination of 100 Mature Open Source Projects,” *First Monday* **7**, No. 6 (June 2002), http://firstmonday.org/issues/issue7_6/krishnamurthy/.

28. Michelle Levesque, “Fundamental Issues with Open Source Software Development,” *First Monday* **9**, No. 4 (April 2004), http://firstmonday.org/issues/issue9_4/levesque/.

29. The term “patch” is used as defined in *Wikipedia*, The Wikimedia Foundation, http://en.wikipedia.org/wiki/Patch_%28computing%29.

30. D. A. Wheeler, *Why Open Source Software/Free Software (OSS/FS, FLOSS, or FOSS)? Look at the Numbers!* (February 21, 2005), http://www.dwheeler.com/oss_fs_why.html#security.

31. *IBM and Dessault Systèmes Announce Availability of 64-Bit Application of AIX*, Cable News Network LP, LLLP, A Time Warner Company (October 13, 2004), http://money.cnn.com/services/tickerheadlines/mw/074116.htm.

32. *NSW Government Goes Open Source*, CIO Australia (January 10, 2004), http://www.cio.com.au/index.php?taxid=7&id=570520727.

33. *Brazil to Migrate 300,000 PCs to Linux*, Open Source Industry Australia (June 21, 2004), http://www.osia.net.au/news/brazil_to_migrate_300_000_pcs_to_linux.

34. I. Marson, *Danish Government Agency Opts for Open Source*, CNET Networks, Inc. (September 23, 2004), http://news.com.com/Danish+government+agency+opts+for+open+source/2100-7344_3-5379068.html.

35. J. Best, *Munich to Stick with Open Source*, CNET Networks, Inc. (June 17, 2004), http://news.com.com/

Munich+to+stick+with+open+source/ 2100-7344_3-5237356.html.

36. J. Blau, *Munich Migrates to Linux Despite EU Debate*, Computerworld Inc. (September 30, 2004), http://www.computerworld.com/softwaretopics/os/story/0,10801,96288,00.html.

37. *Open Source Has* "*Strategic Role*": *Malaysia*, CNET Networks, Inc. (August 28, 2003), http://asia.cnet.com/news/software/0,39037051,39148245,00.htm.

38. K. G. Bautista, *BSA Lobbies vs. Pro-Open Source Software Laws*, BusinessWorld Online, Inc (October 20, 2004), http://itmatters.com.ph/news/news_10202004d.html.

39. P. Festa, *South Africa Embraces Open Source*, CNET Networks, Inc. (February 5, 2003), http://news.zdnet.co.uk/software/0,39020381,2129893,00.htm.

40. *Open Source Software Gets Government Approval in Sweden*, Interchange of Data between Administrations (February 21, 2003), http://europa.eu.int/idabc/en/document/880/318.

41. G. Gross and E. Scannell, *U.S. Government Seeks the Open Road*, InfoWorld (March 12, 2004), http://www.infoworld.com/article/04/03/12/11FEopgov_1.html.

42. G. N. Shapiro and E. Allman, *Sendmail Evolution*: *8.10 and Beyond*, Sendmail Inc., http://www.sendmail.org/~gshapiro/Sendmail-8.10.Paper.pdf, also available in *Proceedings of the FREENIX Track: 1999 USENIX Annual Technical Conference*, Monterey, CA, June 6–11, 1999, http://www.usenix.org/events/usenix99/full_papers/shapiro/shapiro.pdf.

Accepted for publication December 22, 2004.
Published online April 7, 2005.

Kimberly D. Simon
IBM Systems and Technology Group, 11501 Burnet Road, Austin, TX 78758 (kdsimon@us.ibm.com). Kimberly Simon is a software engineer for the IBM Linux Technology Center in Austin, Texas. She received a B.S. degree in computer engineering from Auburn University in Alabama. Her primary work and interests are in the area of IT security, ranging from LDAP (Lightweight Directory Access Protocol) and database protection to security, government certifications, identity management, and intrusion detection. She is a member of the Society of Women Engineers (SWE) and volunteers for the Women in Engineering Program (WEP) at the University of Texas. She is also a member of the Exploring Interests in Technology and Engineering (EXITE) program at IBM. ■