



## The role of open computing, open standards and open source in the public sector

*An IBM White Paper*

*There is a general consensus in governments around the world that “Open” is both important and good. There are also a great many people worrying about what “Open” means and debating the issue vigorously. We shouldn’t be surprised by this. “Open” is critical to government. The defining of “Open” is also a critical dynamic in the competitive landscape of the information and communications technology (ICT) industry. There are, as a result, many points of view and many definitions. So how we come to terms with “what is Open and how can it be leveraged to support dynamic, responsive, and cost effective government?” is a very important question.*

*Information systems are essential to help governments deal with the complexity of globalization of economies, unanticipated threats, citizen demands and fiscal constraints. Governments need considerable flexibility in the way they configure their information systems. They need to have those systems seamlessly communicate with other systems. They need to be able to reconfigure those systems easily. They need to have the flexibility to source technology from a variety of vendors and leverage innovative emerging technology. They need flexibility and the ability to move information around efficiently. This is where “Openness” comes in. “Openness” is all about the enablement of this flexibility.*

*Governments must also be “open” to their citizens, giving greater access to e-government applications and enhanced responsiveness when citizens and businesses need to interact with government. “Open” implies that public administrations allow access to e-government applications on a choice of platforms and with a variety of technologies so as not to impose a single platform or vendor’s offering on the general public.*

*“Open” is simply a means to an end. It is essential that we do not lose sight of what the goal is. There are various goals of “Openness”. They include:*

- *Ensuring flexibility*
- *Ensuring interoperability*
- *Avoiding vendor lock-in*
- *Avoiding imposing technology decisions on the citizenry*
- *Driving cost effectiveness*
- *Ensuring future access to information*
- *Ensuring a level playing field for competition*
- *Maximizing freedom of action*

There are of course a number of members of the “open” family. These include open computing, open standards and open source software. Some definitions:

Open computing, or simply “openness”, is the philosophical principle that describes architecture and technology procurement policies and practices that align IT with the above stated goals. It permits interoperability by using published specifications for API's, protocols, and data and file formats. The specifications must be published without restrictions that limit implementations, or require royalties or payments (other than reasonable royalties for essential patents).

Open standards are a slightly more refined level of open computing. They are distinct because of the formal role of independent organizations who have adopted the specifications through some democratic process. Many characteristics help to define open standards. Varying degrees of “openness” are possible depending on the number of characteristics which apply to an individual standard and standards may become more open over time. For instance, a proprietary standard may become largely open if it is endorsed by a standards setting body or consortium.

The characteristics of openness when applied to standards are:

1. Published without restriction (other than reasonable royalties for essential patents),
2. Freely available for adoption by the industry,
3. Control by an open industry organization with a well-defined inclusive process for evolution of the standard,
4. Implemented by offerings that are available in the market.

Standards evolve and move through a maturing process driven by pragmatism, speed to market and efficiency. Examples that many people are familiar with include HTTP, HTML, WAP, TCP/IP, VoiceXML, XML, and SQL. They are typically built by software engineers from various IT/software companies who collaborate under the auspices of organizations such as W3C, OASIS, OMA, ISO, and IETF.

Standards evolve and move through a maturing process driven by pragmatism, speed to market and efficiency. Examples that many people are familiar with include HTTP, HTML, WAP, TCP/IP, VoiceXML, XML, and SQL. They are typically built by software engineers from various IT/software companies who collaborate under the auspices of organizations such as W3C, OASIS, OMA, ISO, and IETF.

**Proprietary** describes interfaces that are developed by and controlled by a given company and have not been made freely available for adoption by the industry. Proprietary software uses non-public interfaces or formats. When an interface is non-public, the owner of the proprietary interface controls it, including when and how the interface changes, who can adopt it, and how it is to be adopted.

**Open Source software** is software whose source code is published and made available to the public enabling anyone to copy, modify and redistribute the source code without paying royalties or fees. Open source code evolves through community cooperation. These communities are composed of individual programmers as well as very large companies. Some examples of open source initiatives are Linux, Eclipse, Apache, Mozilla, and various projects hosted on SourceForge.net.

**Free software and software libre** are terms that are roughly equivalent to Open Source. The term "free" is meant to describe the fact that the process is open and accessible and anyone can contribute to it. "Free" is not meant to imply that there is no charge. "Free software" may be packaged with various features and services and distributed for a fee by a private company. The term "public domain software" is often erroneously used interchangeably with the terms "free software" and "open source software." In fact, "public domain" is a legal term that refers to

software whose copyright is not owned by anyone, either because it has expired or because it was donated without restriction to the public. Unlike open source software, public domain software has no copyright restrictions at all. Any party may use or modify public domain software.

**Commercial software** is software that is distributed under commercial license agreements, usually for a fee. The main difference between the commercial software license and the open source license is that the recipient does not normally receive the right to copy, modify, or redistribute the software without fees or royalty obligations. Many people use the term "proprietary software" synonymously with "commercial software." Because of the potential confusion with the term "proprietary" in the context of standards and interfaces and because commercial software may very well implement open, non-proprietary interfaces, this paper will use the term "commercial software" to refer to non-open source software.

## **The open computing environment**

Most major companies and governments have embraced the concept of open computing. They purchase ICT goods and services from a variety of vendors and expect the technologies to work together. They wish to have the flexibility to deploy hardware and software in a specific way in order to address specific problems in an organization. They do not wish to be locked into a specific vendor and subjected to the priorities and schedules of that vendor. Open computing provides them with a way to treat technology components as discrete modules that can be mixed and matched.

There are a number of common beliefs associated with this trend. ICT organizations investing in open computing believe it will maximize their flexibility and, consequently, the amount of business agility they have. They believe that open computing will allow them to rapidly adopt technology innovations and to exploit technology cost reductions. They believe open computing will provide them some degree of vendor independence.

Of course there are a number of tactics implementers of an open computing philosophy will use. Well thought-out architecture and open standards are critical elements of open computing. Furthermore, many governments and companies are leveraging the fact that the community development process, because of its requirement that the technologies from many entities must work together in a complimentary fashion, must discriminate in favor of open standards. As a result governments and companies are using open source software as a means of accelerating the adoption of open standards which subsequently allows them further to implement open computing.

## **Interoperability**

Interoperability should not be looked at solely as a technical issue. It should ultimately be defined by user experience. Interoperability is achieved when users' expectations regarding the exchange and use of information and application functionality between a variety of devices and between the offerings of various vendors are met. Any technical barrier to interoperability should only be tolerated in some special cases involving overriding and legitimate cases such as security requirements.

Interoperability is also a powerful economic force. It is an enabler of a "network effect" which promotes economic development through the linking of services. Trade, competitiveness, GDP growth, higher employment and the facilitation of efficient trade are all benefits of the interoperability of information systems.

## **A brief history of open standards**

Interfaces are the means by which the many elements of an application talk to each other and to other applications. Historically, interfaces were built by companies to allow internal and external programmers to add to the function of an application and to allow the value of an application to be enhanced with the capabilities of other applications. Companies that owned these interfaces

wielded considerable control, which in turn enabled them to sell vertical software "stacks"<sup>1</sup> to customers or to assemble software stacks that drove value into hardware platforms. This control model was very successful and the source of considerable profits for many of the major founders of the ICT industry. When a company said they were "an IBM shop," "an HP shop," or "a DEC shop," it really meant that, for the most part, they ran their business on a particular manufacturer's hardware, software, architecture, and often proprietary interface specifications.

In the 1980s computing technology started to become more stratified with much more distinct horizontal structures. Computer hardware became more commoditized and architectures more open. This led to greater degrees of modularization, interoperability, and the development of a market place for peripherals. The eventual effect was an increase in the rate of innovation, greater value, and a certain degree of loss of account control by hardware vendors. The software side of the equation also saw horizontal stratification. Operating systems became more generic and independent of hardware platforms. The middleware layer evolved, allowing for greater cost effectiveness and greater innovation and speed to market since application vendors were freed from having to worry about "the plumbing."

These developments started to force interface standardization, which became vital in the effort to exploit networking technology and the growing usage of the Internet. The potential for computers to communicate with each other and for great stores of information to be virtualized was predicated on simple and standardized communications and interfaces. Therefore, while it may have been possible for a business to be an IBM, HP, or DEC shop in the past, it had become impossible for any one company to control the interfaces that ran the world's networks.

During the 1990's a number of major companies made strategic decisions to embrace this evolution toward open standards. These decisions were based on simple pragmatism: If we are going to live forever more in a networked world, then that networked world must run on open standards. This development has been good news for customers of ICT and the ICT industry in general. The skill and resources of these industry players have been critical in the development of robust, functional, and highly practical interfaces that are critical enablers of e-business.

The battle of "openness" is still being waged. For the most part businesses have embraced open standards as a means of ensuring degrees of flexibility and vendor independence. Many vendors have also embraced open standards, either because their role in the ecosystem as a provider of horizontal infrastructure or networking capability necessitates it, or because of their desire to participate in markets dominated by other players who use their market position to promote their proprietary interfaces. Some vendors have been successful in exploiting what economists call the "network effect"--the tendency towards adoption of a common platform owing to the intersecting interests and interdependencies of ecosystem participants, including consumers. In turn, these companies have been able to exert control over programming interfaces and document formats to protect their market positions. However, with the increasing momentum towards open standards and development of powerful alternative approaches such as XML, Web services, and J2EE (which isn't so much a standard as a widely adopted programming model), the ability to exploit proprietary interfaces for competitive advantage will likely diminish.

## **Open standards**

Open standards are not born; they evolve and mature. Typically one vendor or initiator will craft a specification related to their perception of some technical or business requirement. These specifications will often go through a second phase where a "core group" works to mature the specification. This group may consist of a small number (in some cases as few as 2) of technology vendors and/or end users. This core group will evolve and mature the specification at a relatively accelerated rate. They often will publish the specification and may produce a reference platform. They may even release the reference platform as open source software. They will often put commercial products in the market that implement the specification once it has reached a certain level of maturity. If the specification has been published many vendors may deliver products that use the specification to the market. The specification then typically moves to a third phase where it is transferred to some independent organization which is usually composed

of many vendors and technology implementers. These standards organizations will further enhance the specification, may develop interoperability testing suites and procedures, and will maintain it over an extended period of time. As time goes on the specification will typically undergo increasingly fewer modifications and become very stable.

The reason that specifications undergo these phases is entirely pragmatic. When a specification is developed by one or a small core of vendors it can evolve at an accelerated pace and the technology can be delivered to the market in a more timely manner. Its subsequent adoption by a standards body increases the likelihood that the specification will be implemented by a large number of vendors in a consistent manner.

One danger here is the propensity for some vendors to “embrace and extend” an open standard. This is to say, they may incorporate a standard into a product but add to it and extend it in a proprietary way that introduces incompatibilities and inhibits interoperability. For example, if there is a published standard for email formatting and encryption and everyone used it, then any email program could create an email that could be opened and read by users of other standards compliant email programs. However, if one vendor adds proprietary or non-standard extensions to the standard so that emails created with their software could not be opened and read by others, it would reduce the customer value of the standard as an enabler of interoperability and of avoiding vendor lock-in. In most situations this strategy is an attempt by a vendor to regain a control point, meaning some technology that enables a vendor to dictate the terms of the evolution of a certain technology and potentially the ecosystems surrounding that technology, which was nullified by the open standard. It is important to apply a critical eye to vendors who “extend” standards in some proprietary way as their actions may be contrary to the goals of open computing.

## **Procurement policy and open standards**

This life cycle of a specification gives rise to a policy dilemma. If governments insist on formal open standards as defined above, they may be limiting their ability to exploit new and useful technologies. There may be specifications that have not made their way through all the phases to the point where they are formally adopted by a standards group but that are published, implemented in commercial offerings, and that could deliver significant value.

Governments need to be pragmatic. They need to make sure they reflect on the top level goals of “openness”. If the specification is published, can be implemented by a number of vendors, the core group sponsoring the specification consists of vendors who represent a sizable market share and have a certain degree of influence in the industry, and the specification is being delivered in commercial products from a number of vendors, then there is a high likelihood that the specification will evolve to become an open standard. This situation largely meets the openness test and governments should feel comfortable implementing products that use these specifications. If, on the other hand, the specification is not published and is not or can not be implemented by a number of vendors, and it is being advocated by a core group that may not have the market strength to ensure its success, then it may not evolve to become an open standard and, thus, fails the openness test.

Procurement policies that insist that products be “open” make eminent sense. Procurement policies that insist products implement formal open standards may be discriminating against very useful technologies unnecessarily. When crafting the language of ICT procurement policy, governments should insist that products meet the test of open computing and that the goal of openness always be taken into account.

## **The role of open source software**

It has become clear that open source software (OSS) has an important role to play in the ICT industry and in business, in general. Yet there is considerable confusion about the strengths and weaknesses of OSS. Some believe it will eventually replace the commercial software model; even that OSS is a critical element of a modern democracy. Others decry OSS as the single

greatest threat to capitalism and the principles of intellectual property--the ruin of Western society. Neither of these extremes is accurate. OSS, for the most part, represents a software development process. It can be leveraged to provide considerable value and complement commercial software products. Many commercial products, a number which is likely to grow, use open source technology under the covers as componentry in what is referred to as a "hybrid model." Commercial software products will continue to play a critical role for the foreseeable future. The rationale for this conclusion will be discussed below.

Open source software isn't usually developed by any one company; it is developed by a community and it comes in many flavors. For example, the Linux movement was started by an individual who was quickly joined by many others who used the Internet to collaborate on the project we know today as the leading open source platform. Others such as Apache are offshoots of academic work. Some, such as Mozilla and Eclipse, were seeded by substantial code donations from major software companies.

## **Open source projects**

OSS describes many different kinds of projects, each with different characteristics. These projects are long term and evolutionary in nature and don't usually have a defined end point. It is useful to break these groups down into four major subcategories, as described below.

1. **"Academic" projects:** University students and professors, and researchers from academic, public, and private research facilities use OSS as a means of peer review. OSS provides a mechanism for others to comment on their work, review its merit, and improve the code. Some researchers and companies publish their work to the public in order to promote discussion and innovation. Some companies use the publication of academic open source as a means of improving relations with other developers and researchers and even to attract talent. The emphasis is on innovative function. The code base typically isn't structured in such a way--and isn't stable enough--to deploy in a commercial setting. IBM Research is an active participant in this community and publishes many research projects for peer review.
2. **"Foundation" projects:** These projects focus on some of the lower layers of the software stack. The major examples are Linux (operating system), Eclipse (development tools framework), Apache (HTTP server), Globus (grid computing), and Mozilla (Web browser). These types of projects represent the bulk of open source community development. They are delivering the bulk of the value of the Open Source movement and have by far the highest level of commercial deployments and support infrastructure. These projects have large and vibrant communities supporting them. Many also have large commercial software companies supporting them, including IBM.
3. **"Middleware" projects:** These projects are focused on areas such as application servers, databases, and portals. This group includes Web application servers such as Tomcat and JBoss, databases such as MySQL, and portal software such as Jetspeed. In most cases, these communities have small private companies at their core who have service business models. These communities have not attracted a critical mass of programmers and are likely to have only a marginal impact on the software industry in the near term. The dynamics of this category will be discussed below.
4. **"Niche" projects:** This group represents the bulk of the 56,000 registered projects at SourceForge.net (a Web-based repository of OSS projects). These projects are typically very small and have a very niche focus. Many are experiments or toolkits developed by industry organizations. They collectively represent a significant influence on the market but individually have negligible impact. Code from these projects is used to test against and as a source of ideas and techniques.

**"Walled Garden" projects** use some of the same community development processes but are not really open source projects. I.e., the source code is not available to be modified by the public at large. Walled garden projects are composed of small groups of companies that put in place a

mechanism to collaborate on the development of some common technology that all participants may access. The source code from this kind of project is only available to members of the walled community whose membership is typically by invitation only and supported by dues. These types of projects are not common. Commercial software companies may also use a walled garden approach as an internal development process.

### **The role of foundation and middleware projects**

Of the open source project groups described above, the two most frequently discussed are the "foundation" projects and the "middleware" projects. These are the areas with potentially disruptive influence in the market.

It is quite normal that mature, common, "foundation" layers of technology will become commoditized over time. Value will migrate to higher and more innovative layers. This has certainly been true of the hardware industry and will also be true of the software industry. We have seen many examples of this already. TCP/IP stacks, compressions tools, compilers, Web servers, and browsers come to mind. As a rule, the adoption of open standards by an industry will accelerate this migration of value as the cost of entry to a market falls and customers have greater choice. There have been some exceptions to this phenomenon, notably desktop operating systems and basic office productivity function. These markets have been protected in large part by the "network effect" discussed earlier. The combination of network effect and control over proprietary interfaces has slowed the rate of commoditization and allowed the realization of considerable control and profitability from those product areas.

Open source software, specifically Linux, has the potential to disrupt the status quo. Already, Linux has presented a serious challenge to Microsoft's strategy to move into the Unix server business with Windows/Intel economics. Linux has become a very capable and scalable server operating system. Many companies and governments have implemented Linux servers. Linux offers them some key advantages. On the low end, Linux on Intel provides attractive economics. In the mid-range, Linux can run on high performance RISC servers and has great affinity with Unix. On the very high end, Linux can run on mainframe business computers and even runs some of the world's most powerful super computers. This kind of scalability and multiplatform support has some very obvious advantages, including the ability to 1) scale ICT systems to match shifting business requirements, 2) optimize programmer productivity and minimize support requirements, and 3) leverage new innovations and take advantage of new cost structures regardless of platform.

To a lesser extent, Linux is also making inroads on the desktop. As companies and governments seek more cost effective ways to deliver ICT services to their user populations, the combination of the efficiencies of Linux, the runtime support of Java, the Web application model supported by a Web browser, and the aggregation and personalization services of a portal combined with high function open source office alternatives such as "Open Office" are providing a viable and financially attractive alternative to the traditional Windows environment.

### **Health and sustainability for foundation OSS projects**

The growing popularity of Linux and open standards leads to an interesting question: How far up the software stack will open standards and open source accelerate commoditization, and at what rate? There are many factors to consider, the two most important being 1) the health and sustainability of a community and 2) the involvement of major industry players.

Regarding the size and relative health of the open source communities, we can distinguish the work of software development communities focused on the foundation/platform projects and the work of major industry players focused on projects at higher levels in the software stack.

Let's take a look at these two factors driving today's OSS "foundation" projects.

## Large and healthy developer communities

As noted earlier, projects such as Linux, Eclipse, Mozilla, Globus, and Apache have relative large and healthy developer communities. There are a number of reasons for this success from both a business perspective as well as a developer perspective.

### *Business perspective*

- These open source projects provide **business value** to end user customers. Because the community development process enforces modularity and standards compliance, these projects yield high dividends and provide significant value for businesses and governments.
- These open source projects represent areas where **one or more significant commercial software vendors** has taken an active sponsorship role and where there is solid overall support. And the rewards are increasingly clear. Commercial software vendors recognize the opportunity to amortize collective investment into commoditized layers in the software stack. Moreover, commercial software vendors believe they can disrupt competitive control points such as critical interfaces or software stack components, eliminate vendor lock-in, and support multiple platforms. This is especially true of Linux.
- **Critical skills** to support ongoing foundation projects are readily available in the software development community. These skills come from the hacker<sup>2</sup> community, from the consumers of the technology, from companies with complementary business models, from commercial software vendors, and from academia. All of these skill sources are fed by academic institutions that use OSS code to teach computer science concepts. Consequently, there is an educated population of programmers emerging out of colleges and universities who are familiar with the open source environment and with the technologies being built in open source projects.
- These communities have **long-term plans for improvements** in design, development, test, documentation, etc. The increasing number of conferences, commercial interest, and worldwide collaboration regarding the future of open source computing is a strong indication of its maturity and reveals the degree of importance associated with it.

### *Developer perspective*

- There is a **passionate interest in developing and enhancing code**. These projects make for a highly visible "artistic canvas" for programmers to demonstrate their skills and technique. Some developers participate, at least in part, to establish a reputation and build credentials to interest potential employers.
- There is **significant overlap between sets of users and developers**. The direct participation of users and developers in the use of open standards and open source software leads to rapid incorporation of domain requirements into the code base and a significant degree of project focus on business-oriented as well as technical problems. This focus supports the relevancy and value of the project's output and subsequently the health of community participation.
- The communities are **diverse and interactive**. The sheer volume and the many diverse perspectives of project participants lead to a rapid rate of innovation, optimization, and bug-fixing.
- Communities have **strong overall project/code leadership**. A strong leadership style of the open source "maintainer" (the person in charge of deciding what goes in and what stays out of any particular release) is essential. Linus Torvalds, the creator of the Linux operating kernel, is a great example of this leadership. Strong leadership promotes focus

and momentum and generally helps keep the project moving in the right direction. This success factor, like many others becomes circular and self promoting. The important successful projects attract strong leadership and the strong leadership, in turn, promotes the health of the project.

The foundation projects mentioned above are healthy and appear to be sustainable.

### **Involvement of major industry players**

The second factor--involvement of major industry players--toward viable OSS solutions is also significant. Linux, Eclipse, Mozilla, Globus, and Apache all have active participation and support from some of the largest and most influential software companies in the industry. The Linux community counts HP, IBM, Sun, and Oracle among its active contributors. Members of the Eclipse community include Borland, IBM, Oracle, Sybase, Montavista, Red Hat and many others. Mozilla has participation by Netscape, AOL, IBM, Red Hat, and Sun. Globus is sponsored by IBM, Microsoft, and Cisco. Apache lists Apple, IBM, Sun, CollabNet, and Red Hat among its active contributors. These lists are, of course, small samples of the large number of companies that are involved in these initiatives. Their members also include contributors from Brown University, Massachusetts Institute of Technology, Stanford University, and many other academic institutions.

Why are major software companies increasingly interested in open standards and the open source movement? Primarily because investing in these communities makes good business sense, for three major reasons:

1. **Drive rapid adoption of open standards.** Those companies that have made a strategic decision to back open standards and whose business model is based on widespread adoption of open standards support open source projects to help business and governments get ready access to high-quality, open source implementations of open standards and speed industry adoption.
2. **Use OSS as a strategic business tool.** OSS can eliminate competitor "lock-in," create a competitive playing field, and open up new business opportunities. Vendors who support OSS make money on OSS services as well as through the sale of components based on open source platforms. And their customer base is satisfied by the ability to extend their systems via additional components and services from alternative vendors, as needed. OSS can effectively level the playing field by removing many of the structural advantages that a company controlling proprietary control points may have.
3. **Extend mindshare.** Active participation in an open source community can enhance partnership relationships and help build relationships with a broad spectrum of developers.

The importance of the participation of these commercial software vendors in the advance of OSS adoption cannot be understated. These companies provide a critical mass of highly skilled programming expertise and competence. The involvement of these companies provides stability to open source communities and reassurance to consumers of the technology as to its long-term viability. The technical resources these companies provide can kick start an open source community and provide critical technical and domain expertise as well. For example, the donations of the Netscape code base to Mozilla by AOL and the Eclipse framework to the Eclipse Project by IBM have enabled two new OSS movements over the past three years. In addition, the financial resources of these companies can fund critical government standards compliance testing. A recent example of this is Oracle and IBM's announcing their commitment--likely to mean millions of dollars--to get Linux certified to meet US government security standards.

### **Challenges for middleware OSS projects**

Middleware OSS represents a potential opportunity to many entrepreneurs--typically small companies seeking to develop alternatives to commercial middleware technology. And as we

know, opportunity invites competition and competition drives innovation. Some middleware OSS projects may migrate into the foundation efforts and be picked up by those communities. We have already seen basic application server functions migrating into the Apache project. This function will be very useful to many business and governments and will likely be used widely in the near future.

However, middleware OSS projects have not yet had much impact in the enterprise space nor have they had much impact in the mid-market space. For a variety of reasons, they do not share the characteristics that have made the foundation projects successful, including the critical mass of a healthy and sustainable community or the support of major vendors.

More importantly, middleware projects currently lack enterprise features and characteristics of commercial alternatives such as scalability, reliability, enterprise support, multiple languages, robust documentation, and integrated tooling. Compared to the OSS foundation level projects, which have attracted support from major software vendors, it has also proven to be difficult for middleware OSS communities to raise the funds required to purchase testing tools and do government standards compliance testing.

Security is another significant concern. The architecture of commercial offerings typically addresses security in ways that OSS middleware projects cannot. The mid-market is dominated by independent software vendors (ISVs) who develop industry solutions that require many 'enterprise' characteristics for software they either embed or require as a prerequisite.

The commercial application server vendors are rapidly innovating and creating function that allows them to competitively differentiate their products and resist commoditization due to high-value capabilities such as portals, business processes integration, automation, nationalization, and other strengths. The middleware market will continue to evolve and very likely be characterized by high rates of innovation.

## **A quick note about licensing**

There are many arguments that go back and forth about the role that licensing plays in the potential success of OSS projects. In the extreme, some industry players have referred to OSS licenses as "cancer." In our opinion, arguments over licensing are overblown. OSS projects use various types of licensing ranging from a very flexible BSD (Berkeley Software Distribution) type of license, which has no obligations other than publication of copyright notice and sometimes an attribution requirement, to a GPL (GNU General Public License) type license, which requires source code for modifications that you distribute to be made available. The GPL follows certain aspects of the modification. Some have suggested that this restriction limits intellectual property rights and have characterized it as "viral." But, note, the requirement to provide source code for modifications only applies if you intend to redistribute the code outside of your organization. To an end user, the requirement to provide source code changes are not very important.

There is considerable experience in the industry working with the various OSS licenses. There is a general consensus that for the most part these licenses do not represent a barrier to the integration of OSS into business solutions and will likely have little, if any, impact on the success of OSS one way or the other.

## **Procurement policy and open source software**

Open source software can be very useful and may be a cost effective way of deploying technology. It certainly is a widely used development approach that can be found inside a large number of products in the market. There may be a rationale in putting in place procurement policy that insures that open source-based offerings are being considered along with commercial offerings as a means of overcoming institutional or historical biases or of ignorance of alternatives.

Procurement policy should not establish preferences or mandates. Preferences and mandates work against the goals of openness. They may force the adoption of suboptimal technologies

that reduce flexibility and interoperability and whose total cost of ownership is more than commercial alternatives. It only makes sense that procurement be directed at the best product at the best value which maximizes interoperability and flexibility and the other openness goals, regardless of the development practices used.

## **What is driving open computing, open standards, and open source success?**

The reasons for the success of open computing and open standards are clear. They are a necessary feature of a networked world and they are essential to the critical factors of organizational flexibility and agility. However, the explanation for the popularity of open source software is a little less clear.

In general, businesses and governments see value in the following OSS features:

- **Flexibility to modify.** Some businesses or governments require specialized modifications to a code base to accommodate specific business or technical requirements. OSS offers this flexibility. The U.S. National Security Agency (NSA) has used this flexibility and created a secure version of Linux
- **Cost effectiveness.** OSS often has some attractive up-front cost advantages, although there is much debate as to the total cost of ownership (TCO). There is anecdotal evidence that some companies have realized considerable license savings. On the other hand, it is argued that scarcity of skills translates to higher support and maintenance costs that nullify the up-front cost advantage. The economic case will vary from geography to geography as the availability of skill and labor rates vary. Customers will need to make an assessment of the TCO advantages or disadvantages of OSS on a case-by-case basis.

### **Specific benefits: the case for Linux**

In the specific case of the foundation projects, and especially in the particular case of Linux, which businesses and governments see value in:

- **Multiplatform support.** Some businesses and governments have realized advantages in deploying a common operating environment across multiple hardware platform architectures. They also see some advantage in being able to scale their applications beyond the confines of one particular hardware architecture. Linux for example is available on systems ranging from cell phones to super computers. An enterprise might deploy an application on an Intel platform and then need to scale the application to midrange systems such as IBM's POWER technology, Sun's SPARC technology, or even to IBM's mainframe zSeries systems. There may be other reasons such as reliability, manageability, corporate or departmental mergers, security requirements, or the exploitation of some specialized capability, to move platforms. This kind of flexibility allows businesses and governments to match their ICT support to their business requirements and have that ICT support change as the business requirements change.
- **Standards enforcement/promotion.** Standards compliance is a natural and inevitable characteristic of community developed software. So some businesses and governments have decided to use OSS as a means of promoting or enforcing open standards and open computing. Implementing Linux or Apache, for example, implies the implementation of many of the most important Internet standards. Strict standards adherence at the lower foundation layers permits considerable flexibility of configuration and of choice of application and vendor.
- **Security/Audits.** Some businesses and governments believe that being able to see the underlying mechanics of code enhances the reliability and security of that code. While this transparency argument does make aesthetic sense and many cite it as a rationale for

selecting OSS alternatives, there isn't any hard data to suggest that OSS is inherently any more secure or stable than commercial software. Linux offers the best case due to its maturity, the involvement of major software vendors, and the sheer volume of reviewers--all of which means that bugs and security holes are discovered and patched very quickly. On the other hand, in many domains it is quite clear that the commercial alternatives are both more reliable and more secure owing to their architecture and the quality control measures that many commercial vendors enforce. Without question, security needs to be designed into a program and a program's security needs to be assessed on a case by case basis. In any case, the platform foundation layer of the software stack is where most of this scrutiny is focused.

- **Linux as an agent of economic development.** This situation is specific to governments and to the case of Linux and occurs typically in developing countries. Governments are using their purchasing power and influence in the economy to create a critical mass of demand to stimulate the development of local skills and local business activity around the ecosystem that evolves around an OSS platform. The hope is to stimulate a domestic software industry. We even see governments sponsoring the establishment of Linux competency centers to accelerate Linux adoption and to act as a focal point for skill development and investment.

## Conclusion

Open computing, open standards, open source software, and commercial software which implements open standards are all succeeding because they are enablers of technological innovation and because businesses and governments recognize value in them. Businesses and governments will strive to attain the flexibility and business agility of the on demand world. Open computing platforms--both hardware and software--are essential underpinnings of the journey towards on demand computing. The role that open computing and open standards have played in the evolution of e-business has been well established. The role that open computing and open standards *will* play as part of the open commercial and open source projects that embrace those specifications and standards is central to the further evolution towards more responsive, focused, and resilient e-business and e-government capability.

Businesses and governments are embracing open computing, open standards, and some open source projects. Procurement policies should be pragmatic, insist on openness and take into consideration the goals of openness.

IBM has made the strategic decision to embrace these concepts and is aligning its hardware, software, services, and consulting businesses to support its customers on the journey toward "on demand".

## Notes

<sup>1</sup> Generally speaking, "software stack" refers to the complete array of software -- from the operating system, through the middleware layer, to various specific components -- used to accomplish a specific business goal; for example, the assemblage of software designed for a financial services solution.

<sup>2</sup> The word "hacker", in this context, is used by its original meaning. Originally, "hacker" referred to a person who is highly skilled and interested in computers and not a person engaged in criminal mischief.