Running Code as Part of an Open Standards Policy

Rajiv Shah
Adjunct Assistant Professor
University of Illinois at Chicago
rajiv.shah@alumni.illinois.edu

Jay Kesan
Professor & Mildred Van Voorhis Jones Faculty Scholar
University of Illinois at Urbana Champaign
kesan@illinois.edu

Abstract

Governments around the world are considering implementing or even mandating open standards policies. They believe these policies will provide economic, socio-political, and technical benefits. In this article, we analyze Massachusetts’s open standards policy as applied to document formats. This policy was known as the open formats policy, and Massachusetts believed it would offer substantial economic and technological benefits. However, our analysis shows these benefits never emerged.

The failure of the open formats policy is due to the lack of running code. Running code refers to multiple independent, interoperable implementations of an open standard. With running code, users have choice in their adoption of a software product. Choice brings about economic and technological benefits. Consequently, we urge governments to incorporate a “running code” requirement when adopting an open standards policy.

1. Introduction

Open standards have grown in prominence within the last few years. A simple definition of an open standard is a specification that is publicly available and freely implementable. Examples of open standards include the transmission protocols such as FTP, the language of Web pages HTML, and the image format PNG. Many observers have asserted that the growth, widespread use and popularity of the Internet are largely the result of a reliance on open standards [1].

Open standards are widely considered to have positive economic and technological benefits. When standards are open and freely available, it becomes possible for anyone to develop an interoperable implementation. This reduces the ability of vendors to tie a standard to the purchase of other products, i.e., vendor lock-in [2]. Open standards are able to facilitate multiple interoperable implementations, thus providing users with choice. Choice brings with it lower costs and technological variation.

Open standards are often conflated with “open source”, but they are very different. Open source is a development model for software based on the public availability of the source code. While open source software typically relies upon and uses open standards, it should not be conflated with open standards. Open standards can also be incorporated into software developed through other development models, such as proprietary software, e.g., Microsoft.

The economic, socio-political, and technical benefits of open standards have led to widespread support within industry, academia, and policymaker circles. Numerous reports have called for government policies that strongly encourage or mandate open standards. Some prominent examples include reports by the Berkman Center at Harvard and the Committee for Economic Development as well as the Office of Government Commerce in the United Kingdom [3-5]. Consequently, many governments around the world have enacted legislation that encourages or mandates the use of open standards.

This article argues that conventional open standard policies by government are flawed. The policies overlook the importance of multiple independent interoperable implementations. The basis for this argument rests on our analysis of one of the most prominent open standard policies, Massachusetts. Massachusetts developed an open formats policy as a subset of their open standards policy. We argue that Massachusetts acted prematurely in adopting the OpenDocument Format (ODF), because there was only one implementation, OpenOffice.org. With one implementation, Massachusetts was trading the shackles of Microsoft Office for those of OpenOffice.org. Massachusetts should have waited for multiple interoperable independent implementations if they wish to receive the economic and technological benefits of open standards. As a result, other governments considering open standard policies should incorporate a “running code” requirement before adopting an open standard.
Running code refers to the existence of multiple interoperable implementations of an open standard. This analysis stands apart from the conventional wisdom that holds the Massachusetts policy was a good idea, but rushed, and other governments should consider emulating it [6]. Updegrove has provided a rich history of the policy with a complimentary focus on the policy. His analysis supports the efforts of Massachusetts and reflects his support of open source and open standards. He recently co-founded Digistan, an organization that promotes open standards. Ghosh, the noted open source scholar, believes “Massachusetts has created a blueprint for development of an open standards IT policy that others can follow”[7]. A similar interpretation can be found in the work of the New York state government, which sought to emulate the Massachusetts policy, but avoid the procedural controversies in setting such a policy [8]. The most critical assessment comes from the work of Shah, Kesan, and Kennis. However, they focus on offering a litany of lessons both procedural and substantive that other governments should take away.

This paper relies on the data collected by Shah, Kesan, and Kennis [9] and Updegrove [10]. Based on this data, we offer a unique and compelling critique of the Massachusetts policy than differs from the work of past scholars and commentators.

2. Background on open standards

Standards in IT are important because they allow for interoperability and interconnection. They provide "the digital equivalent of a common gauge for railroad tracks" [11]. In the past, standards were created through two means: de facto and de jure [12]. De facto standards are those created by an industry. These standards are not publicly distributed and often require a licensing fee for others to use them. These standards may be referred to as proprietary standards to emphasize who controls the standard. A simple example of a proprietary standard is the Microsoft Word document format. De jure standards, on the other hand, are those standards endorsed by formal standard-setting organizations, such as the International Organization for Standardization (ISO).

In the area of information technology, however, a new process for developing standards has emerged. Firms, individuals, and governments all recognize the value of standards. The importance of standards has led to a new organizational form, consortia, to collectively develop standards [13]. In developing standards, consortia offer a variety of new approaches for creating and distributing standards. The most prominent is known as open standards.

Governments have typically defined open standard as “specifications for systems that are publicly available and are developed by an open community and affirmed by a standards body” [14]. This is the widely accepted definition of open standards [15].

The benefits of open standards arise from the ability to offer compatibility between implementations, which in turn encourages competition among vendors [16]. These two facets work together to provide adopters of open standards the ability to choose between multiple implementations with low switching costs.

Consider the example of the simple mail transport protocol (SMTP), which is a foundational standard for e-mail. SMTP is an open standard. The market for mail software consists of at least 10 different implementations with at least a 2% market share [17], and over 30 implementations listed in the Wikipedia entry on “comparison of mail servers”. Examples of the available implementations are shown in Figure 1. The multiplicity of implementations provides adopters with varying costs, ease of use, complexity, and integration with other applications. Because of the compatibility of open standards, users can choose the implementations that best fits their needs.

An overlooked characteristic to ensure compatibility is interoperability. Figure 2 illustrates how various implementations of SMTP are interoperable. The consequence is that information flowing from one implementation, e.g., Postfix will be correctly received by any other implementation, e.g., Microsoft Exchange Server. Without interoperability, information could not pass
seamlessly and users would not receive the benefits of open standards.

Figure 2. Interoperability for SMTP

The attraction of open standards is their ability to offer interoperable implementations. Users want choices in their selection of implementations. Additionally, they seek implementations that offer low or no switching costs. After all, users are tired of being locked-into proprietary solutions. They are seeking the choice and flexibility that open standards provide. This motive is elucidated in Massachusetts declaration for an open standards policy [18].

Proponents of open standards focus on these benefits, with no caveats. For example, Louis Suarez-Potts the Community Manager of OpenOffice.org states the following:

ODF, coupled with OpenOffice.org, shakes the foundations of monopoly, the status quo. With an easily usable open standard and Foss technology, one is not limited to a single vendor; there is, as the phrase puts it, no vendor lock-in. . . . It need not be OpenOffice.org. It can be any other application; it just has to be implementing the ODF or otherwise supporting it. Think of it as real consumer choice, or consumer freedom. We call this freedom "no vendor lock-in." And it's a freedom that goes beyond simple consumer choice . . . [19].

As we later point out, these advocates are assuming an open standard will lead to a vibrant market of multiple independent interoperable implementations that removes vendor lock-in. A review of the Massachusetts open formats policy shows how this blind faith in open standards can lead governments awry.

3. Massachusetts open formats policy

This section begins by reviewing the key facts of the Massachusetts open formats policy. It is based on work that documented the Massachusetts open standards policy [9].

In 2003, Massachusetts set a policy of open standards. Massachusetts decided to move toward open standards, because of the cost savings and flexibility open standards provide. In 2005, it extended this policy to open standards for document formats. This part of the open standards policy is referred to as the open formats policy. For background, document formats are standards for maintaining information produced by typical office productivity applications, e.g., word processing, spreadsheet, and presentation programs. The most common are the proprietary document formats by Microsoft, such as DOC for word processing.

The open formats policy lead Massachusetts to focus on the use of the OpenDocument Format (ODF) as an acceptable open standard for government documents based on the following grounds. ODF was developed by OASIS, a non-profit standard-setting organization, through an open process. The ODF standard was published and openly available. Sun, the founder of the principal ODF implementation (OpenOffice.org) and the founder of the standards process, agreed to a royalty free license as well as a covenant not to sue for patent infringement. Additionally, the standard relied on an XML format. Massachusetts was trying to move toward XML data format for documents storage, because of the increased technological flexibility XML provides for the storage and analysis of content.

A deadline of Jan 1, 2007 was set. After that date, all documents created were to be saved in OpenDocument Format. In setting this deadline and mandating the ODF format, Massachusetts became the first government to mandate ODF [20]. This shift to ODF and open formats has been widely debated and discussed, most richly by Updegrove [10].

In the end, Massachusetts never made the full switch to ODF or OOXML (another XML-based document format initially developed by Microsoft and later deemed an acceptable open format by Massachusetts). As of January 2008, both OOXML and ODF have had very little usage by the Massachusetts government. Using a Google Search, the authors identified 17,300 DOC files, 31 ODT files based on ODF, and 0 DOCX files based on OOXML. While this includes only files that are publicly available through the mass.gov Web site, these numbers do reveal a limited use of these standards.

Massachusetts never received the economic and technological benefits of open standards. Despite Suarez-Potts argument that the combination of ODF
and OpenOffice.org would bring consumer choice and freedom, there is little evidence of any significant cost savings or increased flexibility that has arisen with the open formats policy. This lack of benefits remains puzzling and should prompt other governments to proceed cautiously. In the next section, we argue what went wrong.

4. Analyzing the failure of the open formats policy

Our explanation first requires an understanding of the definition of open standards used by Massachusetts:

For the purpose of this policy, open standards are defined as follows: specifications for systems that are publicly available and are developed by an open community and affirmed by a standards body. Hypertext Markup Language (HTML) is an example of an open standard [18]. This sort of definition is widely used. Other governments, such as New York, have relied on a similar definition [8]. Bruce Perens, an advocate of open standards, defines open standards on the basis of six principles: availability, maximum end-user choice, no royalty, no discrimination, extension or subset and predatory practices [21]. A more scholarly analysis that largely overlaps Perens definition can be found in Kretchmer’s “Ten Rights that Enable Open Standards” [15].

The key problem with all of these definitions is that they focus exclusively on the development of an open standard. For example, ensuring the standard was developed publicly and is freely available. This has lead to a conflation of the definition of open standards with their consequences. In his prescient article on open standards, Joel West warns that the definition of open standards may be confused with the consequences of standards, i.e., multiple implementations [22].

In the case of Massachusetts, the ODF mandate was set in 2005. At the time, ODF was an immature standard. ODF 1.0 was approved as a standard within OASIS in May 2005 and ISO in May 2006. Only OpenOffice.org could natively save documents in the acceptable document format (ODF) at the time of the policy. As Figure 3 shows, this situation is markedly different than that for SMTP. Multiple implementations did not exist in 2005. There was no running code for ODF in 2005.

The Massachusetts open formats policy was really an OpenOffice policy!

The process in Massachusetts was akin to switching vendors for office software. While ODF was an open standard, multiple implementations did not exist, so Massachusetts was effectively locked-in to OpenOffice.org. This is why we believe in the case of Massachusetts, the open formats policy as constituted in 2005 was better understood as an OpenOffice policy rather than a policy seeking competition and choice for government.

Figure 3. The lack of implementations for ODF

Our feeling is that when Massachusetts initially mandated ODF, its underlying goal was moving away from Microsoft Office toward an open source Office suite. Our reasoning is that the cost savings estimated by Massachusetts for switching to ODF rest largely on the reduced licensing costs for Microsoft Office and Windows [23]. These estimates suggest Massachusetts was planning to move away from Microsoft to an open source solution. Massachusetts later decided not to adopt a solely open source solution.

The OpenOffice.org lock-in has still not disappeared in 2008. In another study, we have shown significant interoperability issues between various ODF implementations, e.g. OpenOffice, Wordperfect, and KOffice [24]. We found considerable variation among how well each implementation performed. For ODF, the compatibility scores ranged from a raw score of 151 for OpenOffice.org (100% - weighted percent) to 48 (55% - weighted percent). As a result, the study warns that adopters of ODF will effectively be locked-in to OpenOffice.org, unless they are willing to suffer problems in translating documents between implementations. This study shows that despite Suarez-Potts claim, the adoption of an open standard does not guarantee choice and the freedom from vendor lock-in. Even several years later, running code for ODF does not exist due to the significant interoperability problems.
Our view is that there are several factors that can explain the conflation of the definition of open standards and its consequences. First, there is a natural conflation between open standards and open source. They share a long history and share many of the same supporters. Most people could not distinguish the difference between them. With an open source project, it is implicit that an implementation exists; however, the same is not true for open standards. As a result, it becomes easy to forget that open standards depend upon implementations. Second, for simple open standards, implementation can be trivial and taken for granted. However, the document formats example shows that implementation cannot always be taken for granted. The ODF standard is about 700 pages, and OOXML is over five thousand pages. Implementing standards of this length is very difficult, especially when interoperability between multiple implementations is a significant goal. As a result, the consequences of open standards should not be taken for granted when considering complex standards. Third, vendors may try to conflate an implementation with a standard. After all, if a government standardizes one particular implementation, then a vendor has the possibility of locking that government into using its product. It is no accident that the dominant implementations of ODF and OOXML are tied to major IT vendors.

5. Running code requirement

Governments and organizations are intrigued about the benefits of open standards. Cheerleaders like Suarez-Potts are telling governments that open standards are the proverbial silver bullet. Open standards can offer more choices while reducing costs. Our concern is that governments could believe this hyperbole without fully understanding how best to receive the benefits of open standards.

To ensure they receive the benefits of open standards, governments should incorporate a requirement of running code, i.e., multiple interoperable independent implementations, into their open standards policy. Independent implementations ensure that standards can be adopted by anyone (not just the party pushing for the standard). Interoperability is important, because it ensures that multiple implementations will all work together. This is essential from a user perspective, because it ensures minimal costs when switching between implementations.

The idea of running code is not new. The Internet Engineering Task Force requires that a standard be implemented by two independent vendors before it can be considered an open standard [25]. This requirement is informally known as the need for “running code” and is a requirement for all IETF open standard standards. Not all standards organizations require running code. The ISO, for example, does not require implementations during the standards approval process.

Standards organizations have differed in their emphasis on running code during the development process. Their main concern is that an emphasis on running code may influence how a standard develops. The problem here is that if there are flaws in the design or if it has a poor implementation, these flaws get transferred to the standard, instead of being fixed during the development process.

The ISO, for example, does not require implementations during the standards approval process. The W3C has traditionally not focused on implementations during the standards process. However, it has recently introduced a “Candidate Recommendation” stage. This stage focuses on vendors’ successful implementation of the standard. At this point, the W3C has not formally required interoperability but has supported informal interoperability testing [26].

While standards organizations may differ on the desirability of running code, we believe it is imperative for governments to adopt open standards only when there is running code. Without running code, governments are just switching vendors and not receiving the enormous benefits of open standards.

A running code requirement would have led Massachusetts to defer adopting ODF. In 2005 adoption of ODF was premature. Multiple implementations did not exist, hence there was no running code. Only now in 2008 is there support for several independent implementations of ODF. However, there are still interoperability problems to address. The hope is that these will be resolved soon, so there will be multiple interoperable implementations. With the arrival of running code, governments can then choose the implementations that fit their needs, while receiving the benefits of lower costs and increased flexibility.

The drawback to a running code requirement is that it requires additional work for adopters. Evaluating an open standard requires an understanding of how it is actually being implemented. This requires adopters to perform due diligence on how a standard is being used as well as ensure interoperability exists. Moreover, as implementations evolve, adopters will need to reevaluate implementations. Nevertheless, to gain the benefits of open standards, it is necessary to ensure there is running code.
Massachusetts’s early adoption of ODF did help to increase the adoption of ODF by other customers and vendors. Massachusetts can be seen as a critical force in shaping the technological direction of the market. The open formats policy was critical in influencing Microsoft’s decision to make its OOXOXML standard an open standard and to support ODF. While Massachusetts did help shape the market, not all governments wish to influence the technological direction of the market. Some governments prefer to act as passive consumers by not appearing to influence the market through their procurement. For these governments, they need to incorporate a requirement of running code into their standards policy.

6. Conclusions

Open standards play a vital role in software development and adoption. The advantages of open standards make it reasonable that governments will seek to adopt open standards. However, the Massachusetts experience suggests that governments must not judge open standards by how they are written, but by how widely they are implemented. After all, without multiple interoperable independent implementations, i.e., “running code”, governments may find themselves suffering from lock-in to an open standard solution.

The running code requirement is relevant to all organizations adopting open standards. Adding a running code requirement to an open standards policy puts an emphasis on how the standard is actually being used. We believe if adopters of open standards insist on running code, software developers and vendors will further support open standards and their interoperability. The result will be an array of economic and technological benefits.

7. Acknowledgements

This material is based on work supported by the National Science Foundation under Grant Nos. 0429217.

8. References