

OPEN DOCUMENT FORMAT

ODF ALLIANCE



Taking the Measure of Openness:

a closer look at ODF & OOXML

Sam Hiser

May 2007

[This page is intentionally left blank.]

Table of Contents

Summary.....	1
Introduction	2
“Openness” in Document Formats.....	2
(I) Open Life-Cycle	3
(A) ODF.....	3
(B) OOXML.....	4
(II) Open Availability	4
(A) ODF.....	5
(B) OOXML.....	5
(B)(1) Non-Disclosure of Elements of OOXML.....	5
(B)(2) OOXML Elements Requiring An Application to Emulate Microsoft Office.....	5
(III) Multiple Implementations	6
(A) ODF.....	7
(A)(1) ODF's Multi-Vendor Support	7
(A)(2) ODF's Reuse of Existing Standards.....	7
(A)(3) ODF's Covenant Not-To-Sue Provides Necessary Assurance	7
(B) OOXML.....	7
(B)(1) OOXML Is Fully Implemented In One Application.....	8
(B)(2) OOXML Fails To Reuse of Existing Standards.....	8
(B)(3) OOXML's IPR Covenant Offers Limited Protection	8
(IV) Interoperability Across Different Systems	9
(A) ODF.....	9
(B) OOXML.....	10
(B)(1) Platform Dependencies of OOXML.....	10
(B)(2) Application Dependencies of OOXML.....	10
(B)(3) Partial or Non-Specification.....	11
(B)(4) Poorly-Designed XML	11
Conclusion & Recommendation	12

Summary

An open, XML-based standard for displaying and storing data files (text documents, spreadsheets, and presentations) offers a new and promising approach to data storage and document exchange among office applications. A comparison of the two XML-based formats – OpenDocument Format (ODF) and Office Open XML (OOXML) – has revealed substantial differences in their “openness” evaluated on the basis of widely-accepted criteria.

- Only ODF is developed and improved in an open, multi-vendor, multi-stakeholder process not dominated by the interests of a single company or its products. OOXML, while submitted to a formal standards body, is less open in its development and maintenance.
- ODF is the openly-available standard, published fully in a document that is freely available and easy to comprehend, as demonstrated by the number of competing applications in which it is already implemented. OOXML's complexity, length, omissions and single-vendor features have prohibited any full implementations in other software to date.
- ODF is the only format unencumbered by any intellectual property right (IPR) restrictions as certified by the Software Freedom Law Center. Numerous elements designed into but undefined by the OOXML specification require actions and behaviors upon document files that are specific only to Microsoft Office applications. Microsoft's intellectual property policies and the company's *Open Specification Promise* prohibit such application behavior emulation and, therefore, prohibit document interoperability to the degree required in the marketplace.
- ODF is the only standard offering true interoperability today. OOXML's dependencies on Windows and Office mean that its full implementation will require the purchase of licenses for the Microsoft advanced operating systems for both the desktop and the server and Microsoft Office 2007.

What's in a name? Though any one breach of universally-agreed criteria would raise serious questions as to whether a proposed document format is “open,” OOXML fails all criteria. Choosing OOXML would have long-term implications for the future of software application choice, access to information, competition and innovation. Choose wisely. Choose ODF.

Introduction

In today's knowledge economy, information and communication technology (ICT) architectures need to be flexible – modular, pluggable and easy to set up, fast to integrate and fast to take down and re-purpose if necessary – for governments and businesses alike to meet the demands of their citizens and customers. The architectures need to be built around agreed protocols, and data needs to flow seamlessly across different applications and platforms.

Open standards are at the core of these new interoperable architectures. The Internet is the best example of what open standards can achieve. Based largely on the framework of TCP/IP and HTML, both open standards, the Internet's open architecture enabled new and unimagined ways of communicating, working and innovating.

This paper analyzes the “openness” of two emerging XML-based document formats – OpenDocument Format (ODF) and Office Open XML (OOXML). The analysis is timely. ODF achieved approval as an international standard in May 2006 (ISO 26300). OOXML was recently submitted to JTC1 of the International Organization for Standardization (ISO and the International Electrotechnical Committee (IEC), triggering a 9-12 month process during which OOXML will be reviewed and voted on by national standards bodies.

“Openness” in Document Formats

With the emergence of flexible ICT architectures that depend on interoperability, a document format's degree of openness will affect the free flow of information across the world's computer systems. ODF and OOXML each promise different results for access, cost, choice, and collaborative innovation.

Given their document-intensive nature, governments have a unique interest in the growing debate about open formats. An open standard for documents that is widely available in many software products would allow agencies and departments to exchange and collaborate on office documents, store them for long periods of time, ensure public access to them, and enable electronic communication with citizens – without forcing on themselves or their citizens any particular brand of software. Hence the growing interest in open document standards, and the increasing number of governments around the world requiring their use.¹

Various definitions of an “open standard” have been proposed.(cite) It

becomes quite clear in looking at the definitions that there are four areas the definitions have in common. They can be characterized broadly as:

- Open Life-Cycle
- Open Availability
- Multiple Implementations
- Interoperability Across Different Systems

Let us now take a closer look at these consensus criteria and the degree to which ODF and OOXML satisfy each one. By satisfying these criteria thoroughly, a document format can be sufficiently open to bring us fully into the Internet era of low-cost, collaborative computing based on modular services and architectures.

(I) Open Life-Cycle

An Open Life-Cycle is a setting for the development and evolution of a standard that is open to public participation, where meetings are held in the open, where meeting artifacts (notes, minutes, e-mail correspondences and documentation) are published, and where all participants – individuals and companies – have a voice in consensus decision-making on the standard's technical development. If a standard is intended to be platform and vendor neutral, then the participation of multiple implementors working on multiple platforms is essential.

(A) ODF

ODF was developed and is continuously evolving in an *open, appealable, and published* process. ODF was developed at the Organization for the Advancement of Structured Information Standards Technical Committee that operates in full public view of the public while inviting the participation of any interested party. Email of all technical committee communications and archives, including meeting notes and documentation, is made publicly available on the OASIS web site. Technical Committee meeting participation is unlimited: individual members of OASIS as well as corporate representatives participate equally, with voting eligibility established by level of individual participation.

ODF's Technical Committee includes multiple active participants representing both proprietary and open source implementors. Other participants include accessibility advocates, academic and government

representatives, and consumer groups. Fifty-two people in all participate in the ODF Technical Committee and its three technical sub-committees (Formula, Metadata and Accessibility), with most participants contributing to more than one committee. A full 33 percent are individuals who do so without corporate affiliation.

ODF began as an OpenOffice.org specification and was submitted to OASIS in 2003, where it spent more than two years being improved and reviewed, followed by a year-long review process in ISO, receiving more comments and corrections, before it was officially published as an ISO standard in November 2006. It underwent a combined 4 years in technical committees being refined by standards bodies. During this time ODF was implemented in dozens of applications, both proprietary and open source.

(B) OOXML

Ecma Technical Committee 45, which maintains OOXML, works in private. Ecma TC 45's proposal, voting, balloting and appeals process policies are not known outside of Ecma. Such policies are not published. Further, it is not clear if voting, balloting or appeals processes are used in the development of OOXML, since the formats were pre-developed within Microsoft's Office software development group and Microsoft retains veto power over any ongoing changes that are proposed. Moreover, while there is an *ex post facto* reporting by press release, the meeting activities of Ecma TC 45, the committee's work-in-progress, documents and e-mail are not public.

ECMA membership requirements are limiting: individuals are not welcome to participate except special invitation or through their corporate organization. Only senior corporate members have the right to vote on a TC. OOXML's 6000+ pages were reviewed in less than a year by Ecma, and were submitted to ISO in December 2006 without a single implementation.

Most importantly regarding its "open life-cycle," OOXML is a single-vendor dominated specification. Ecma TC 45 behaves only as a consultative body. A single vendor, Microsoft, retains ultimate control over development of OOXML, with serious implications for interoperability and single-vendor functionality.

(II) Open Availability

An open format is published in a document that is freely available, easy to comprehend and fully published so that third party developers can implement it entirely, thereby ensuring interoperability of their products.

(A) ODF

ODF is free of cost and fully published in ODF or PDF form from the OASIS web site. There are no hidden or missing annexes or parts. ODF has been implemented in many vendors' products – both proprietary and open source – on both Windows and other operating systems. This widespread adoption is only possible because ODF is fully disclosed, making it easy for competing applications to exchange documents with fidelity.

(B) OOXML

While the OOXML documentation itself may be free of cost and available in multiple parts, the complexity, length and omissions of OOXML raise questions about its *availability* on a practical level.

(B)(1) Non-Disclosure of Elements of OOXML

OOXML contains numerous undocumented elements. For example, OOXML preserves certain file data in binary form based upon legacy formats not disclosed to outside developers. This means it is impossible for any entity besides Microsoft to create effective alternative implementations of the formats.

A second example is the implementation of OOXML for spreadsheets in Office 2007 (Excel 2007), which also makes use of data in binary form. As these binary formats have not yet been shared openly, it is presently impossible for other vendors and developers to create working alternative implementations of the OOXML binary spreadsheet without further seeking and retaining this related technical data.

(B)(2) OOXML Elements Requiring An Application to Emulate Microsoft Office

Numerous elements designed into but undefined by the OOXML specification require actions and behaviors upon document files that are specific only to Microsoft Office applications. Specific examples include:

- lineWrapLikeWord6 (Emulate Word 6.0 Line Wrapping for East Asian Text)
 - mwSmallCaps (Emulate Word 5.x for Macintosh Small Caps Formatting)
 - shapeLayoutLikeWW8 (Emulate Word 97 Text Wrapping Around Floating Objects)
 - truncateFontHeightsLikeWP6 (Emulate WordPerfect 6.x Font Height)
-

- Calculation)
- useWord2002TableStyleRules (Emulate Word 2002 Table Style Rules)
- useWord97LineBreakRules (Emulate Word 97 East Asian Line Breaking)
- wpJustification (Emulate WordPerfect 6.x Paragraph Justification)
- shapeLayoutLikeWW8 (Emulate Word 97 Text Wrapping Around Floating Objects)

The practical effect is that features of OOXML files will not be readable, editable or renderable by software applications which cannot perfectly emulate Microsoft Office. While the stated purpose of OOXML is to ensure the backwards compatibility with old files, such *deprecated* legacy application features threaten to create a dependency on Microsoft's Windows operating system and office suite applications.

Such dependencies are not consistent with Open Availability, because other vendors cannot emulate the behavior of Microsoft Office or Windows, as these features are not incorporated into the OOXML technical specification or legally allowed to be duplicated by developers. Microsoft's intellectual property policies and the company's *Open Specification Promise* prohibit such application behavior emulation and, therefore, block true document interoperability.

In short, OOXML's complexity, length, omissions and single-vendor features prohibit efficient, cost-effective or fully working implementations in other software. OOXML is therefore unlikely to ever be fully implemented by any application other than Microsoft's Office for which it was created.

(III) Multiple Implementations

Multiple implementations means the existence today of wide-ranging choice among applications to create and share documents efficiently and without added cost. Only an open, portable document format can allow for interoperability between productivity applications. An *open* format attracts multiple implementations in software. That is, the format can and will be *designed in* to many different software applications without practical, technical, legal or other impediments. This means no IPR restrictions for its implementation. If one had no other way to tell, the format specification with more numerous complete implementations indicates that it follows open principles more rigorously and will better deliver the desired results for enabling information free-flow between applications and platforms.

(A) ODF

Although ODF originated as the native format of OpenOffice.org, it has been adapted to an open process and is designed for open re-implementation in any software.

(A)(1) ODF's Multi-Vendor Support

ODF was conceived for implementation in multiple software applications without limitation as to which party or how many can implement it. One of the principal objectives as specified in the OASIS charter is to have as many implementations as possible, and indeed multiple implementations of ODF exist today. Currently OpenOffice.org, StarOffice, K Office, Lotus Notes, AbiWord, Google Docs & Spreadsheets, Zoho Writer, AjaxWriter and other applications work with ODF documents. As pointed out previously, multiple implementations of ODF existed prior to its submission to ISO.

(A)(2) ODF's Reuse of Existing Standards

ODF makes use of as many existing open software standards as possible, which both improves quality and shortens production schedules, and conforms to XML toolsets. This makes re-implementation of the ODF format attractive, efficient and feasible, and it provides for an efficient and smaller standard specification process and document.

(A)(3) ODF's Covenant Not-To-Sue Provides Necessary Assurance

ODF is developed and maintained by a Technical Committee whose members are obligated to a royalty free policy. One of the major contributors to ODF, Sun Microsystems, provides a simple covenant-not-to-sue that covers any of its patents used in the development of ODF implementations. The ODF specification is allowed to be fully implemented in both commercial and open source software without legal impediment. ODF has been certified as free of legal encumbrances that would prevent its use in free and open source software.

(B) OOXML

OOXML is a single-vendor format which presents obstacles to implementation in software other than Microsoft's.

(B)(1) OOXML Is Fully Implemented In One Application

The ECMA TC 45 charter offers in its Programme of Work, "To Produce a formal Standard for office productivity documents which is fully compatible with the Office Open XML Formats." Microsoft Office 2007 is currently the only application that provides a full implementation of OOXML.

(B)(2) OOXML Fails To Reuse of Existing Standards

OOXML ignores a number of open standards which are available and should be used, including SVG for drawings and MathML for equations. [For a detailed accounting, reference the number of entries on OOXML at <http://www.groklaw.net/staticpages/index.php?page=20051216153153504>]. Failure to re-use existing standards increases the cost and difficulty of third-party implementation and the frequency of the document formats' code-level interactions with proprietary features in Microsoft operating systems and applications.

(B)(3) OOXML's IPR Covenant Offers Limited Protection

The patent-protection pledge only protects what is explicitly specified in the standard. Microsoft's Open Specification Promise states that the company will not sue anyone for implementing the explicit/visible parts of the OOXML specification; however, there are numerous undocumented facets and behaviors of the OOXML formats which, if implemented by another entity, would risk "intellectual property" violations against Microsoft software. Additionally, there are serious gaps in the promise not to sue.

First, the Promise does not cover any material that is referenced, but not described in detail, within the specification. Even if the referenced material is required (or normative), no patent rights extend to the implementer. For example, numerous sections, including those sections which require mimicking the behavior of proprietary Microsoft products, do not appear to be described in detail and therefore might not be covered by Microsoft's commitments. Additional necessary Microsoft proprietary technologies not described in detail include OLE, macros/scripts, encryption, and DRM. Microsoft has not stated a position on whether any patent rights implicated by these technologies will be made available on terms acceptable to ISO. Second, the Microsoft promise is limited to claims, "...that are necessary to implement only the *required* portions of the Covered Specification.." [emphasis added]. The Promise does not cover optional or non required

portions. To the extent that the implementer includes “excluded optional portions (or non required elements of optional portions)” that are in OOXML, the implementer would be unlicensed to any Microsoft patents covering those items and vulnerable to infringement allegations. For example, password features for WordProcessingML may not be required but are described in the specification (2.15.1.28, page 1,158).

The patent promise of OOXML has a level of uncertainty and a gap of protection by the originator of the proposal that could adversely impact the openness and availability of the standard to implementers. Accordingly, OOXML does not have, nor will it under existing IP restrictions, multiple implementations.

(IV) Interoperability Across Different Systems

Interoperability across different systems means a format can be fully implemented in any application, regardless of the platform/system on which that application operates. A truly open document should easily read, authored and edited from within different system environments and across different applications. Moreover, an open document's presentation layout should be rendered with fidelity by alternative applications operating on different platforms.

(A) ODF

ODF supporting applications are available on all major computing platforms, such as Windows, Linux, Solaris, AIX, MacOS, and a variety of web-based on-line editing environments, such as Google Docs. Users can create documents with the wide variety of ODF supporting applications on any platform and exchange them with users working on any other platform. These documents are being exchanged with high levels of fidelity today while some formatting issues have been raised, loss of content is infrequent in single-trip file transfers. Round-trip file interoperability with multiple passes between more than one different ODF-compliant application can, in fact, be poor.

However, content loss is becoming increasingly rare for single trips and formatting issues and content fidelity are being addressed as ODF-conforming applications improve their conformance to the specification.

(B) OOXML

There are numerous areas where OOXML falls short of what is required of an interoperable, multi-vendor, multi-platform, multi-application format. [Link to Grokdoc]. These shortcomings fall under five general categories: 1) platform dependencies; 2) application dependencies; 3) partial or non-specification; and 4) poorly-designed XML. While not meant to be an exhaustive list, these OOXML dependencies on Windows and Microsoft Office pose the greatest concern in terms of achieving interoperability across platforms and applications

(B)(1) Platform Dependencies of OOXML

These refer to features that can only be implemented on Windows, or optimized for Windows without regard to access by other vendors. Examples include:

- [DevMode](#), a method Windows uses for handling information about printer or display settings, is one such operating system dependency carried within OOXML documents. DevMode will not work in an OOXML file in a non-Microsoft environment.
- [GUID](#), a proprietary Microsoft Windows and .Net implementation of the [UUID](#) standard for applications to coordinate and identify resources within an operating system, is another hidden system dependency tying OOXML files to the Microsoft environment.
- 3.2.29 "Workbook Protection" (page 2698) defines an encryption algorithm by including several pages of C-language source code, code which appears to have byte-ordering dependencies and will produce different results on different machine architectures
- "Clipboard Data" (page 5905) defines a schema type that can encode clipboard format values for Windows and the Macintosh, but doesn't seem to allow for other operating systems

(B)(2) Application Dependencies of OOXML

OOXML documents' collaborative functionality and integration with e-mail and other applications depends upon further purchases of additional software from Microsoft. Such capabilities are not available with these files with software which is not Microsoft software:

- VBA macros contained in ECMA/MOOXML documents will not function outside the Microsoft environment.
- an enumerated list of border art means that every application that

wishes to fully comply with the standard must somehow license the use of those graphicsⁱⁱ

- "Disable Features Incompatible with Earlier Word Processing Formats" (page 2252) explicitly states that it only considers the needs of Word 97-2003.
- "Disable Features Not Supported by Target Browser" (page 2120) is designed to optimize for various version of Internet Explorer and disregards entirely I.E.'s main competitor, Mozilla Firefox, as well as other alternative browsers

(B)(3) Partial or Non-Specification

To the extent a feature is only partially specified or not specified at all, other vendors' products will not be able to interoperate with it. Examples include:

- the OOXML specification does not specify how macros or script are embedded in OOXML document
- "autoSpaceLikeWord95" (page 2161) merely defines semantics in reference to a legacy application whose behavior is nowhere specified
- OOXML preserves certain file data in binary form based upon legacy formats not disclosed to outside developers (see Section)
- the implementation of OOXML for spreadsheets in Office 2007 (Excel 2007) also makes use of data in binary form

(B)(4) Poorly-Designed XML

Engineers in the field agree there are certain practices with respect to the use of XML from which it appropriate not to depart. Otherwise, the notion of a standard format for accessing data across disparate systems is defeated once unique techniques are put to use in particular environments which do not translate to others. OOXML disregards sensible practices in the following ways:

- OOXML requires bitmasks (see "Conditional Formatting Bitmask," page 2,478).ⁱⁱⁱ Commonly agreed proper XML implementation would never employ bitmasks. For example, XSL Transformation ("XSLT"), a useful method for translating from one legitimate XML dialect to another, lacks bitwise functionality, making the use of bitmasked data impossible outside the Microsoft fold.

- OOXML carries forward a known pernicious legacy bug from the Lotus 1-2-3 era (see "Date and Times" (page 3305-6). Such obnoxious standardization of old mistakes for the purpose of backward compatibility with a single vendor's software products requires that spreadsheet dates treat the year 1900 as a leap year; this renders dates incorrectly according to our Gregorian Calendar and would not be an excepted practice in an XML format with an open, multi-party development process. Tools interacting with such products would be adversely affected too.
- OOXML's naming conventions are sub-standard. Proper XML demands adherence to established conventions in the naming of *Attributes* as well as *Child* and *Parent Elements*. OOXML's inconsistency throughout the specification causes confusion and increases the difficulty of implementing the format (see specification section 2.15.1.78 "settings(Document Settings)", page 2,020, where it says, "EOOXML has poor XML Element names").

In short, OOXML is either dependent on, or optimized for, a catalog of Microsoft software products and does not function fully with non-Microsoft software. The practical effect of such dependencies is that the full use of OOXML by individuals or within work groups will require the purchase of licenses for the Microsoft advanced operating systems for both the desktop and the server and Microsoft Office 2007.

Conclusion & Recommendation

Pressure from customers, including many governments has pushed technology companies toward openness and toward ODF. Microsoft has responded with its new format, OOXML. However, a close examination of the origins, technical specifications and follow-on implementations of both formats reveals significant differences. Where ODF meets the four objective criteria of open standards handsomely, OOXML does not satisfy any of the four neither as extensively nor sufficiently in its own right to justify acceptance as a useful format.

ODF showcases how an inclusive, consensus-driven, transparent development process can produce a standard that is available to everyone.

OOXML's weaknesses begin at the fundamental level: its goals conflict. While the format proposes itself as a solution to backward compatibility, its approach to backward compatibility – through including application-specific references with questionable IP restrictions within the specification – blocks full implementation by entities other than Microsoft. Interoperability can not be achieved with OOXML.

In light of such fundamental limitations, basic questions need to be resolved before OOXML is considered for use as a standard format for documents. The questions resonate: How can OOXML with its lack of an open life-cycle, lack of complete documentation in the specification, lack of multiple software implementations, and lack of interoperability across diverse platforms meet the legal as well as practical needs in the organization for long-term document archiving and for accommodating the flow of information correctly through business processes across different types of systems?

ICT executives and policy-makers will need to look carefully at their objectives in deciding which document format is appropriate for their users and for the long-term efficacy of their organization's systems and information culture. It is no longer necessary to accept the one solution offered; and, by the same token, it is important to get this decision right – given the importance of the document format for influencing the many other areas of system procurement.

*The author is Vice President & Director of Business Affairs
at the OpenDocument Foundation, Inc., 501(c)3.*

Endnotes

- i <http://www.odfalliance.org/resources/GlobalViewODFPolicy.pdf>
- ii <http://lnxwalt.wordpress.com/2007/04/06/to-the-members-of-the-california-state-assembly/>
- iii What is a bitmask?