

OPEN DOCUMENT FORMAT

ODF ALLIANCE



Achieving Openness:

a closer look at ODF & OOXML

Sam Hiser

June 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
(C) Conclusion.....	4
(II) Open Availability	5
(A) ODF.....	5
(B) OOXML.....	5
(B)(1) Non-Disclosure of Elements of OOXML.....	5
(B)(2) OOXML Elements Require An Application to Emulate Microsoft Office.....	6
(C) Conclusion.....	7
(III) Multiple Implementations	7
(A) ODF.....	8
(A)(1) ODF's Multi-Vendor Support	8
(A)(2) ODF's Reuse of Existing Standards.....	8
(A)(3) ODF's Covenant Not-To-Sue Provides Necessary Assurance	8
(B) OOXML.....	9
(B)(1) OOXML Is Not Fully Implemented in Any Application.....	9
(B)(2) OOXML Fails To Reuse Existing Standards.....	9
(B)(3) OOXML's IPR Covenant Offers Limited Protection	9
(C) Conclusion.....	10
(IV) Interoperability Across Different Systems	10
(A) ODF.....	11
(B) OOXML.....	11
(B)(1) Platform Dependencies of OOXML.....	11
(B)(2) Application Dependencies of OOXML.....	12
(B)(3) Inadequate Specification.....	13
(B)(4) Poorly-Designed XML	13
(C) Conclusion.....	14
Final Conclusion & Recommendation	14

*This document in PDF format is designed
for 2-sided printing. Use your printer's
duplex settings and SAVE PAPER!*

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”) – across widely accepted “openness” criteria has revealed substantial differences, including the following:

- ODF is developed and maintained in an open, multi-vendor, multi-stakeholder process that protects against control by a single organization. OOXML is less open in its development and maintenance, despite being submitted to a formal standards body, because control of the standard ultimately wrests with one organization.
- ODF is the only openly-available standard, published fully in a document that is freely available and easy to comprehend. This openness is reflected in the number of competing applications in which ODF is already implemented. Unlike ODF, OOXML's complexity, extraordinary length, technical omissions and single-vendor dependencies combine to make alternative implementation unattractive as well as legally and practically impossible.
- ODF is the only format unencumbered by intellectual property rights (IPR) restrictions on its use in other software, as certified by the Software Freedom Law Center. Conversely, many elements designed into the OOXML formats but left undefined in the OOXML specification require behaviors upon document files that only Microsoft Office applications can provide. This makes data inaccessible and breaks work group productivity whenever alternative software is used.
- ODF offers interoperability with ODF-compliant applications on most of the common operating system platforms. OOXML is designed to operate fully within the Microsoft environment only. Though it will work elegantly across the many products in the Microsoft catalog, OOXML ignores accepted standards and best practices regarding its use of XML .

Overall, a comparison of both formats reveals significant differences in their levels of openness. While ODF is revealed as sufficiently open across all four

key criteria, OOXML shows relative weakness in each criteria and offers fundamental flaws that undermine its candidacy as a global standard.

Introduction

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

Open standards are at the core of these new interoperable systems. Based largely on the framework of TCP/IP and HTML, both open standards, the Internet's open architecture has enabled new and unimagined ways of communicating, working and innovating. The Internet is the best example of what can be achieved when systems inter-operate around open standards.

With that in mind, this paper analyzes the "openness" of two emerging XML-based document formats. The analysis is timely. ODF achieved approval as an international standard in May 2006 (ISO 26300).ⁱ OOXML was recently submitted to JTC1 of the International Organization for Standardization (ISO and the International Electrotechnical Committee (IEC), triggering a nine- to 12-month process during which OOXML will be reviewed and voted on by national standards bodies.ⁱⁱ Much of the information in this paper has appeared before, but not in a synthesis on the openness theme.

"Openness" in Document Formats

With the emergence of flexible ICT architectures that depend upon 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 data access as well as cost, choice, and innovation in software.

Given the document-intensive nature of their day-to-day business, governments have a special interest in open document 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. This is why interest in open document standards is growing and why the number of governments around the world requiring their use is increasing.ⁱⁱⁱ

Various definitions of an “open standard” have been proposed.^{iv} With document formats particularly in mind, a consensus emerges among the definitions; they gravitate to agreement in four basic areas:

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

Following is an analysis of both formats across each of these four consensus criteria and a measurement of the degree to which ODF and OOXML satisfy each one. In satisfying these criteria thoroughly, a document format can be deemed 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

A format development process having an open life-cycle means the format is evolved in a fashion 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 as well as companies – have a voice in consensus decision-making on the standard's technical make-up. An open standard should be platform and vendor-neutral, so multiple implementors working on multiple platforms is essential.

(A) ODF

ODF was developed and is continuously evolving in an *open, appeal-able, and published* process. ODF was developed at the Organization for the Advancement of Structured Information Standards (OASIS) Technical Committee which operates in view of the public while inviting the participation of any interested party. E-mail of technical committee communications, including meeting notes and documentation, is archived on the OASIS web site. Technical committee meeting participation is not

limited: individual members of OASIS as well as corporate representatives participate equally, with voting eligibility established by the level of individual participation.

ODF's Technical Committee and Sub-Committees include multiple active participants representing both proprietary and open source implementors. Other participants include accessibility advocates, academic and government representatives, and consumer groups.

Originally the default format in the OpenOffice.org application, ODF went through a rigorous, open evolution process starting in 2003 when it was submitted to OASIS. OASIS members improved it over the course of two years before a year-long review process at ISO, where it received more comments and correction, before it was officially published as an ISO standard in November 2006. During this four years of collaborative technical refinement, many software application vendors implemented it to varying degrees of completeness in both proprietary and open source solutions.

(B) OOXML

Ecma International (“Ecma”) Technical Committee 45 (“TC45”), which maintains OOXML, works in an opaque manner with its voting, balloting and appeals policies not published. It is unclear 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 in TC45. Moreover, while there is an after-the-fact reporting by press release, the meeting activities of TC45, the committee's work-in-progress, documents and e-mail are not public.^v

Barriers to participation in the development of OOXML are high. Ecma membership requirements are limiting: individuals are not welcome to participate except by special invitation or through their corporate membership at Ecma. Only senior corporate members have the right to vote on a TC. The OOXML specification's over 6,000 pages were reviewed in less than a year by Ecma and were submitted to ISO in December 2006 without a reference implementation in software.

(C) Conclusion

OOXML is a single-vendor specification that does not have an open life-cycle. Ecma TC45 behaves only as a consultative body. A single vendor, Microsoft, retains control over development of OOXML. Performance on

such key criteria as interoperability (see Section IV, below) therefore remains in the hands of one private entity. This contrasts significantly with ODF's open life-cycle as maintained at the OASIS ODF Technical Committee.

(II) Open Availability

An open format is published in its entirety in a specification document which is freely available and easy to comprehend. Open Availability also means that a format is freely available for implementation in software.

(A) ODF

ODF is published in its complete form in the *.odt* and *.pdf* formats which are downloadable from the OASIS website free of cost.^{vi}

ODF has been implemented in many different vendors' products, under both proprietary and open source software licenses and on numerous operating system platforms. This is possible because the ODF specification is technically explicit, contains references to other open standards, and because its length is reasonable and manageable.

(B) OOXML

The OOXML specification is free and may be downloaded from the Ecma website. It is, however, difficult to manage, coming in such length and in parts so numerous, in a text so complex and inconsistent in its technical terminology and with so many deliberate omissions, that questions arise about its *availability* on a practical level.^{vii} In the following areas, OOXML presents significant questions and challenges regarding full, open availability.

(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 which are not and have never been 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 or developers to create working alternative implementations of the OOXML binary spreadsheet format.

(B)(2) OOXML Elements Require 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 particular only to legacy Microsoft Office and WordPerfect applications.^{viii} Examples from the OOXML specification include:

<i>Function Name</i>	<i>Description</i>
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 the data associated with these features, once it is contained in OOXML files, will not be readable, editable or render-able by software applications which cannot perfectly emulate Microsoft Office or WordPerfect. While the stated purpose of OOXML is to ensure the backwards compatibility with old files, such *deprecated* legacy data creates a dependency upon Microsoft's Windows operating system and office suite applications.^{ix}

Such dependencies fail the criteria of *Open Availability*. The Microsoft OOXML format includes interactions with its earlier unspecified formats. The result is that other vendors, developers or users cannot access data in Microsoft formats to the same degree as Microsoft software nor to the degree expected of standard XML.

These unspecified format characteristics and application behaviors are not explicit in the OOXML technical specification nor are they legally allowed to be duplicated by developers. Microsoft's license for OOXML, the *Open Specification Promise*, prohibits such application behavior emulation and, therefore, blocks access by non-Microsoft entities to the data in OOXML form -- in effect, this makes the specification unavailable while it also defeats the purpose of having an XML document format.^x

(C) Conclusion

As evidenced by its implementation in multiple products which are offered through multiple vendors, ODF achieves open availability. However, the OOXML specification's complexity, its length, omissions and single-vendor dependencies prohibit efficient, cost-effective or fully working implementations of the format 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

An open document format can and will be *designed in* to many different software applications without practical, technical, legal or other impediments.

From a different perspective it is fair to say that an *open* format has the characteristics that attract multiple implementations. If one had no other way to tell, the format specification with the greater number of complete implementations likely follows open principles more rigorously and will better deliver information free-flow between applications and platforms.

Additionally, software patent restrictions provide a drag on a format's re-implementation in software. They stifle software choice and increase the structural cost in the economy of working with documents.

(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 many software applications without limitation as to which party or how many can implement it. One of the principal objectives as specified in the OASIS ODF Technical Committee charter is to have as many implementations as possible and, indeed, multiple implementations of ODF exist today. Currently OpenOffice.org, StarOffice, KOffice, 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 the specification's quality, shortens re-implementation production schedules and permits conformity with the wide range of W3C XML tool sets. This respect for existing standards makes re-implementation of the ODF format attractive, efficient and feasible, and it provides for an efficient standard specification process and a more compact specification 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. The originator of ODF, Sun Microsystems, provides a simple covenant-not-to-sue which covers any of potential Sun patents used in the development of ODF implementations (although none have been asserted yet).^{xi} The ODF specification is allowed to be fully implemented in both commercial and open source software without legal impediment. Additionally, ODF has been certified by the Software Freedom Law Center as free of legal encumbrances that would prevent its use in any free or open source software.^{xii}

(B) OOXML

OOXML is a single-vendor format which presents obstacles to implementation in software other than products offered by Microsoft for which the format was designed.

(B)(1) OOXML Is Not Fully Implemented in Any Application

The Ecma TC 45 charter states the goal 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 partial implementation of OOXML, while no application exists which is a reference implementation for the formats Ecma TC45 has submitted to ISO.^{xiii}

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

OOXML ignores a number of open standards which are available and should be used, including SVG for drawings and MathML for equations.^{xiv} 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 in Microsoft's *Open Specification Promise* only protects what is explicitly specified in the standard. The *Promise* states that the company will not sue anyone for implementing the *explicit* parts of the OOXML specification; however, there are numerous implied, referenced and undocumented facets and behaviors of the OOXML formats which, if implemented by another entity, would risk "intellectual property" (patent) violations against Microsoft software. Additionally, there are serious gaps in the promise not to sue.

- 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 for an implementation, no patent rights extend to the implementer. For example, numerous sections, including those sections which require replicating the behavior of proprietary Microsoft products, do not appear to be described in detail and

therefore are not covered by Microsoft's *Promise*. 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 associated with these technologies will be made available on terms acceptable to ISO.

- the *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 aspects. 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 patent 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). From a practical perspective, all optional aspects of a format are necessary for a full implementation to function effectively across the wide range of possible software behaviors.

(C) Conclusion

ODF has already achieved multiple implementations and has therefore achieved success with this criteria. In contrast, the patent “promise” of OOXML is insufficient. The embedded uncertainty and gaps in coverage hold back the formats' practical and legal implementability. That's why OOXML does not have multiple implementations today, and it should not be expected to have them under Microsoft's present approach to protecting developers against patent violations.

(IV) Interoperability Across Different Systems

Perfect interoperability across different systems means a format can be fully implemented in any application, regardless of the platform or system on which that application operates. Every respective system would be able to access a document's content and layout parameters to provide perfect document fidelity to the original. While neither ODF or OOXML offers perfect interoperability, we can judge each one's performance based on its proximity to perfection as well as its potential to reach a high practical level

of interoperability for business processes.

It is sufficient that an open document format should be easily read, authored and edited from within different system environments and across different applications. Content should be transmitted without loss, and 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, Mac OS, and a variety of web-based on-line document editing applications, like Google Docs and Spreadsheets. Users can create documents with the wide variety of ODF-supporting applications on any of these platforms and exchange them with users working on different platforms.

Simple documents are being exchanged with high levels of fidelity today. For example, OpenOffice.org can open, read and edit simple text documents originated in Koffice or AbiWord. While certain formatting problems can occur (typically with more-complex documents), loss of content is infrequent in single-trip file transfers.^{xv} Content and layout fidelity in roundtrip document exchange in collaborative work group situations (passing a document back and forth repeatedly) is imperfect but can be improved; there are no technical impediments to achieving better if not perfect roundtrip performance between all ODF-ready applications of the same vintage.^{xvi}

(B) OOXML

OOXML presents problems for document interoperability across multiple platforms.^{xvii} Shortcomings exist in four areas: 1) platform dependencies; 2) application dependencies; 3) inadequate specification; and 4) poorly-designed XML. While not meant to be an exhaustive list, the problems highlighted below reflect the dependencies upon the Microsoft Windows operating system(s) and Microsoft Office application(s) that pose the most obvious and significant threat to OOXML's interoperability.

(B)(1) Platform Dependencies of OOXML

Certain platform dependencies of OOXML are features that can only be implemented or optimized for Windows. Document files containing such features will break or not function the same way in non-Microsoft environments. 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. Consequently, printer initialization, display and other settings may not work in an OOXML file that is transferred into a non-Microsoft environment.^{xviii}
- [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.^{xix}
- 3.2.29 "Workbook Protection" (page 2,698^{xx}) 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 5,905^{xxi}) 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.

System dependencies like these make it unappealing, difficult and in most cases impossible to conduct work productively without Microsoft software.^{xxii} These examples are not exhaustive.

(B)(2) Application Dependencies of OOXML

OOXML documents' collaborative functionality and integration with e-mail and other applications depend upon further purchases of additional software from Microsoft.^{xxiii} The following capabilities are not available to OOXML files when they are accessed by software which is not Microsoft software:

- VBA macros contained in OOXML documents will not run when outside Microsoft applications.
 - 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 graphics.^{xxiv}
 - "Disable Features Incompatible with Earlier Word Processing Formats" (page 2,252^{xxv}) explicitly states that OOXML only considers the needs of Word 97-2003.
 - "Disable Features Not Supported by Target Browser" (page 2,120^{xxvi})
-

is designed to optimize for various version of Internet Explorer and disregards Internet Explorer's main competitor, Mozilla Firefox, as well as other alternative browsers.

(B)(3) Inadequate Specification

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

- The OOXML specification does not specify how macros or scripts are embedded in OOXML document.
- "autoSpaceLikeWord95" (page 2,161^{xxvii}) 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.^{xxviii}
- 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 XML practices in the following ways:

- OOXML requires bitmasks (see "Conditional Formatting Bitmask," page 2,478).^{xxix} 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 to access outside the Microsoft fold.
- OOXML carries forward a known legacy bug affecting date and time for spreadsheets (page 3305-6^{xxx}). This standardization of a famous

old mistake (originating in Lotus 1-2-3) for the stated purpose of “backward compatibility” requires that spreadsheets treat the year 1900 incorrectly as a leap year. This gives wrong dates according to our standard Gregorian Calendar and affects other software that interacts with Microsoft documents.^{xxxii}

- 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 its specification causes confusion and increases the difficulty of implementing the format (see specification section 2.15.1.78 "settings (Document Settings)", page 2,020^{xxxii}, where it says, "EOXML has poor XML Element names").^{xxxiii}

(C) Conclusion

ODF is independent of any particular platform or application, whereas OOXML is either dependent upon, or optimized for, a catalog of Microsoft software applications and platforms and does not function fully with non-Microsoft software. The practical effect of such dependencies is that the use of OOXML by individuals or within work groups will require the purchase of licenses of Microsoft operating systems on both the desktop and the server as well as Microsoft Office 2007.

Final 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 as extensively.

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, design and execution block full implementation by entities other

than Microsoft. The great promise of XML, interoperability, cannot be achieved with OOXML. Ultimately, the format's conflicting objectives make it a poor candidate for a global standard.

In light of such fundamental limitations, basic questions need to be resolved before OOXML is considered for use on any basis and certainly as a potential standard. 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 be looking carefully at their own objectives in deciding which document format is appropriate for their users and for the long-term viability of their systems and information culture. It is no longer necessary to accept the one solution offered. And it is important to get this particular decision right, given the practical and strategic importance of the document format today.

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

Endnotes

- i <http://www.iso.org/iso/en/commcentre/pressreleases/2006/Ref1004.html>
- ii “ISO to Fast Track”, Scott Fulton, Beta News, Mar. 17, 2007 (http://www.betanews.com/article/ISO_to_FastTrack_Office_Open_XML_Process/1173731196)
- iii <http://www.odfalliance.org/resources/GlobalViewODFPolicy.pdf>
- iv There are numerous overlapping definitions of open software standards. Proposals exist from the Open Source Initiative (<http://perens.com/OpenStandards/Definition.html>), the European Union's Interoperability Framework (<http://europa.eu.int/idabc/en/document/3761>), the Danish government's definition of open standards (http://www.oio.dk/files/040622_Definition_of_open_standards.pdf), the “Krechmer Requirements” (<http://www.google.com/url?sa=t&ct=res&cd=1&url=http%3A%2F%2Fwww.csrstds.com%2Fopenstds.pdf&ei=itBcRtjJNoP4wQKDvpyvBQ&usq=AfrqEzFBPDJ6VDzjbvILByt9zwcUkReMVw&sig2=BoL6-Cxk20VCdg56OQDiXQ>) among others.
- v <http://www.ecma-international.org/memento/TC45.htm>
- vi ODF specification version 1.1 (<http://docs.oasis-open.org/office/v1.1/OS/OpenDocument-v1.1.pdf>)
- vii “...pushing through an overcomplex proposal in a very short time frame.” (http://press.ffii.org/Press_releases/FFII_opposes_Fasttrack_adoption_of_Microsoft_OOXML_format_as_ISO_standard)
- viii “How to hire Guillaume Portes” by Rob Weir (<http://www.robweir.com/blog/2006/01/how-to-hire-guillaume-portes.html>)
- ix The OOXML specification includes the following note relating to each of the named legacy deprecated feature tags: “Guidance: To faithfully replicate this behavior, applications must imitate the behavior of that application, which involves many possible behaviors and cannot be faithfully placed into narrative for this Office Open XML Standard. If applications wish to match this behavior, they must utilize and duplicate the output of those applications. It is recommended that applications not intentionally replicate this behavior as it was deprecated due to issues with its output, and is maintained only for compatibility with existing documents from that application. end guidance” This is an *apologia* for the lack of documentation in the OOXML for the named items.
- x The author's “Analyzing the OOXML License” (http://fussnotes.typepad.com/plexnex/2007/01/analyzing_the_m.html)
- xi Sun Microsystems' covenant-not-to-sue (<http://www.oasis-open.org/committees/office/ipr.php>)
- xii <http://www.softwarefreedom.org/resources/2006/OpenDocument.html>
- xiii Marbux, General Counsel to the OpenDocument Foundation, Inc., 501(c)3, comments on Jeremy Allison's article, “Crushed by the Wheels of Industry” (<http://talkback.zdnet.com/5208-11048-0.html?forumID=1&threadID=33784&messageID=621428&start=-1>)
- xiv For a detailed accounting, reference the number of entries on OOXML at <http://www.groklaw.net/staticpages/index.php?page=20051216153153504>
- xv Despite lingering technical and approach problems with the Microsoft / Clever Age ODF Add-in project for Word on Sourceforge, the list there of MS Word application features not supported by the ODF specification is accurate (<http://odf-converter.sourceforge.net/features.html#unsupportedODT>). The list indicates ways a document's formatting can be predicted to break when translating from Microsoft (.doc, for example) to ODF (.odt) formats. In addition to non-alignment of the different formats, honest differences of application design create interoperability dysfunction which manifests as file content or layout breakages that disturb productivity in work groups where mixed desktops exist.
- xvi Poor document roundtrip performance also plagues collaboration between users of different vintages of Microsoft Office.
- xvii Groklaw's “EOOXML Objections,” Jan. 2007 (http://www.groklaw.net/index.php/EOOXML_objections)
- xviii Microsoft Developer Network's explanation of the DevMode data structure (<http://msdn2.microsoft.com/en-us/library/ms535771.aspx>).
- xix http://en.wikipedia.org/wiki/Globally_Unique_Identifier
- xx This page reference to the OOXML specification may be obsolete, since ECMA has posted revisions numerous times since the specification first went “final.”
- xxi This page reference to the OOXML specification may be obsolete, since ECMA has posted revisions numerous times since the specification first went “final.”
- xxii Engineering such dependencies with intent is an integral part of software vendors' traditional business models and a source of vendors' ability to influence software purchase decisions. Open source software and open standards – such as ODF – are making it easier for customers to determine what software they need, when they need it, how much they pay and whether license management takes up their time or not.
- xxiii The information on system requirements for using Microsoft Office (its commercial implementation of the OOXML standard) clearly indicates that “Microsoft Office SharePoint Server 2007 is required for certain advanced functionality” (see <http://office.microsoft.com/en-us/suites/HA101668651033.aspx#5>). The OOXML specification lacks the required documentation on what specific elements of OOXML are not independently implementable.

Endnotes

- xxiv Clip-art should not be defined in the file format since clip-art licensing is not uniform or standard on every platform and this creates another operating system and application dependency (http://www.grokdoc.net/index.php/EOOXML_Objections_Clearinghouse#Inappropriate_user-interface_specifications:_Clip_Art) and (<http://lnxwalt.wordpress.com/2007/04/06/to-the-members-of-the-california-state-assembly/>).
- xxv This page reference to the OOXML specification may be obsolete, since ECMA has posted revisions numerous times since the specification first went “final.”
- xxvi This page reference to the OOXML specification may be obsolete, since ECMA has posted revisions numerous times since the specification first went “final.”
- xxvii This page reference to the OOXML specification may be obsolete, since ECMA has posted revisions numerous times since the specification first went “final.”
- xxviii OOXML relies on undisclosed information (http://www.grokdoc.net/index.php/EOOXML_Objections_Clearinghouse#Ecma_376_relies_on_undisclosed_information).
- xxix 'Bitmasks cause significant XML validation, among several other, problems.' (http://www.grokdoc.net/index.php/EOOXML_Objections_Clearinghouse#Ecma_376_uses_bitmasks.2C_inhibiting_extensibility_and_use_of_standard_XML_tools)
- xxx This page reference to the OOXML specification may be obsolete, since ECMA has posted revisions numerous times since the specification first went “final.”
- xxxi Legacy application bug carry-forwards would not occur in a format that is developed by open consensus of multiple parties (http://www.grokdoc.net/index.php/EOOXML_Objections_Clearinghouse#The_Gregorian_Calendar).
- xxxii This page reference to the OOXML specification may be obsolete, since ECMA has posted revisions numerous times since the specification first went “final.”
- xxxiii As part of the expressed goals of W3C XML, it is important to respect standard naming conventions for readability and honor the principle of consistency in a specification. OOXML does not (<http://www.w3.org/TR/REC-xml/#sec-origin-goals>) and (http://www.grokdoc.net/index.php/EOOXML_Objections_Clearinghouse#Poor_names_and_inconsistent_naming_conventions_for_elements_and_attributes) and (http://www.openmalaysiablog.com/2007/01/ooxml_has_poor_html).