

UPGRADE is the European Journal for the Informatics Professional, published bimonthly at <http://www.upgrade-cepis.org/>

Publisher

UPGRADE is published on behalf of CEPIS (Council of European Professional Informatics Societies, <http://www.cepis.org/>) by **Novática** (<http://www.ati.es/novatica/>), journal of the Spanish CEPIS society ATI (*Asociación de Técnicos de Informática*, <http://www.ati.es/>)

UPGRADE monographs are also published in Spanish (full version printed; summary, abstracts and some articles online) by **Novática**

UPGRADE was created in October 2000 by CEPIS and was first published by **Novática** and **INFORMATIK/INFORMATIQUE**, bimonthly journal of SVI/FSI (Swiss Federation of Professional Informatics Societies, <http://www.svifi.ch/>)

UPGRADE is the anchor point for UPENET (UPGRADE European NETWORK), the network of CEPIS member societies' publications, that currently includes the following ones:

- **Informatik-Spektrum**, journal published by Springer Verlag on behalf of the CEPIS societies GI, Germany, and SI, Switzerland
- **ITNOW**, magazine published by Oxford University Press on behalf of the British CEPIS society BCS
- **Mondo Digitale**, digital journal from the Italian CEPIS society AICA
- **Novática**, journal from the Spanish CEPIS society ATI
- **OCG Journal**, journal from the Austrian CEPIS society OCG
- **Pliroforiki**, journal from the Cyprus CEPIS society CCS
- **Pro Dialog**, journal from the Polish CEPIS society PTI-PIPS

Editorial Team

Chief Editor: Llorenç Pagés-Casas, Spain, pages@ati.es

Associate Editors:

François Louis Nicolet, Switzerland, nicolet@acm.org
Roberto Carniel, Italy, carniel@dgf.uniud.it
Zakaria Maamar, Arab Emirates, Zakaria.Maamar@zu.ac.ae
Soraya Kouadri Mostéfaoui, Switzerland, soraya.kouadrimostefaoui@gmail.com
Rafael Fernández Calvo, Spain, rfcvalvo@ati.es

Editorial Board

Prof. Wolfried Stucky, CEPIS Former President
Prof. Nello Scarabottolo, CEPIS Vice President
Fernando Piera Gómez and Llorenç Pagés-Casas, ATI (Spain)
François Louis Nicolet, SI (Switzerland)
Roberto Carniel, ALSI - Tecnoteca (Italy)

UPENET Advisory Board

Hermann Engesser (Informatik-Spektrum, Germany and Switzerland)
Brian Runciman (ITNOW, United Kingdom)
Franco Filippazzi (Mondo Digitale, Italy)
Llorenç Pagés-Casas (Novática, Spain)
Veith Risak (OCG Journal, Austria)
Panicos Masouras (Pliroforiki, Cyprus)
Andrzej Marciniak (Pro Dialog, Poland)
Rafael Fernández Calvo (Coordination)

English Language Editors: Mike Andersson, Richard Butchart, David Cash, Arthur Cook, Tracey Darch, Laura Davies, Nick Dunn, Rodney Fennemore, Hilary Green, Roger Harris, Michael Hird, Jim Holder, Alasdair MacLeod, Pat Moody, Adam David Moss, Phil Parkin, Brian Robson

Cover page designed by RFCalvo, © Rafael Fernández Calvo 2006

Layout Design: François Louis Nicolet

Composition: Jorge Llácer-Gil de Ramales

Editorial correspondence: Llorenç.Pagés-Casas@ati.es

Advertising correspondence: novatica@ati.es

UPGRADE Newsletter available at

<http://www.upgrade-cepis.org/pages/editinfo.html#newsletter>

Copyright

© Novática 2006 (for the monograph)

© CEPIS 2006 (for the sections UPENET and CEPIS News)

All rights reserved under otherwise stated. Abstracting is permitted with credit to the source. For copying, reprint, or republication permission, contact the Editorial Team

The opinions expressed by the authors are their exclusive responsibility

ISSN 1684-5285

Monograph of next issue (February 2007)

"Next Generation Web Search"

(The full schedule of UPGRADE is available at our website)



The European Journal for the Informatics Professional
<http://www.upgrade-cepis.org>

Vol. VII, issue No. 6, December 2006

- 2 Editorial: UPGRADE in Top Position of Google PageRank for ICT Journals — *Geoff McMullen*

Monograph: Open Document Format (ODF) (published jointly with Novática*)

Guest Editors: *Jesús Tramullas-Saz, Piedad Garrido-Picazo, Marco Fioretti*

- 3 Presentation: OpenDocument Standard for Digital Documents — *Jesús Tramullas-Saz and Piedad Garrido-Picazo*
- 5 Open by Design: The OpenDocument Format Standard for Office Applications — *Erwin Tenhumberg, Donald Harbison, and Rob Weir*
- 14 Is OpenDocument an Open Standard? Yes! — *David A. Wheeler*
- 25 OpenDocument Hidden Traps and their Side Effects on Free/Open Source Software — *Marco Fioretti*
- 29 ISO-26300 (OpenDocument) vs. MS-Office Open XML — *Alberto Barrionuevo-García*
- 38 Interoperability: Will the Real Universal File Format please Stand Up? — *Sam Hiser and Gary Edwards*
- 47 ODF: The Emerging Document Format of Choice for Governments — *Marino Marcich*
- 50 Promotion of the Use of Open Document Formats by the IDA and IDABC Programmes — *Miguel A. Amutio-Gómez*
- 53 A Brief History of Open Standards in Denmark — *John Gøtze*
- 57 Standard Open Formats and Libre Software in the Extremadura Public Administration — *Luis Millán-Vázquez de Miguel*

UPENET (UPGRADE European NETWORK)

- 60 From **Mondo Digitale** (From AICA, Italy)
Programming Languages
Programming Languages: An Introduction — *Carlo Ghezzi*
- 64 From **Pro Dialog** (PTI-PIPS, Poland)
E-Commerce
Organization and Economics of Entertainment Services Networks Exchanging Virtual Goods — *Andrzej P. Urbański*

CEPIS NEWS

- 70 Harmonise Project: Assigning Responsibilities among the Partners and Planning Next Steps — *François-Philippe Draguet*
- 71 News & Events: European Funded Projects and News Updates

* This monograph will be also published in Spanish (full version printed; summary, abstracts, and some articles online) by **Novática**, journal of the Spanish CEPIS society ATI (*Asociación de Técnicos de Informática*) at <http://www.ati.es/novatica/>.

Interoperability: Will the Real Universal File Format please Stand Up?

Sam Hiser and Gary Edwards

This article compares the two solutions that are currently competing to become the universal file format standard: ODF (OpenDocument Format), from the OpenDocument Foundation, and EOOXML (ECMA Office Open XML) from Microsoft. The article analyses the two XML-based formats both from a technical point of view and in terms of their future impact on enterprises, focusing on their true interoperability and their ability to achieve maximum fidelity in the conversion of legacy files.

The large print giveth and the small print taketh away.
Tom Waits, *Small Change*, 1976

Keywords: EOOXML, File Conversion, Interoperability, ODF, Standards, Universal Format, XML.

1 Introduction

Here we are at a crossroads in standards, with the connectivity future of desktops, devices and enterprise systems up for grabs. There are two implementations of XML that can span the horizons of this emerging universe, and the world is waiting to see how these solutions shake out.

It's decision time.

Enter our two contenders: **OpenDocument Format (ODF)** and **ECMA¹ Microsoft Office Open XML (EOOXML)**. Microsoft states they are interoperable. XML's promise of fluid transformation will be fulfilled. The ODF crowd says no way, ODF is the only universal file format worth considering.

There is no need to make a decision, Microsoft insists. If you're using Microsoft applications, running the EOOXML stack, then EOOXML will be your format of choice. For the 485 million users of MS Office on Windows, locked into decades of legacy binaries, building out their daily business processes based on those binaries, then EOOXML is for you. And only EOOXML.

If you're using anything else, ODF will be your choice. Microsoft is pleased to accommodate the choice between document standards [sic] because Microsoft is working as we speak on ensuring the two formats interoperate. After all, both ODF and EOOXML are XML. Right?

Wrong! And therein lies the tale behind a decision that will rule the future. Interoperability and easy transformation is the hallmark of XML. But that's not what's happening here.

Microsoft of course will do everything possible to make the interoperability dream seem real, including forge an

Authors

Sam Hiser is Vice President and Director of Business Affairs at the OpenDocument Foundation. He was OpenOffice.org marketing project leader and along with Tom Adelstein he is the author of "Exploring the JDS Linux Desktop". He contributes regularly to the Financial Times' Digital Business <<http://www.ft.com/technology/digitalbusiness>> section. Sam holds an MBA is from Duke University's Fuqua School of Business. <sam.hiser@opendocument.us>.

Gary Edwards is President of the OpenDocument Foundation, and founding member of the OASIS Open Office XML Technical Committee. He has 20 years of network technology experience. He has spent 12 of those years in the financial services and real estate industries, specializing in the design and implementation of new business models based on emerging Internet technologies. His technology consulting work is that of designing and deploying business process reengineering solutions. <gary.edwards@opendocument.us>.

elaborate set of agreements with a weak Linux and OpenOffice vendor to demonstrate Microsoft's commitment to customer requirements. Look closely and you'll see that the only customer requirement Microsoft is focused on is extension of the old document lockins into the new world of the Internet, XML and SOA. Step back, take it all in, and you'll see that the ultimate objective is to extend the desktop monopoly into servers, devices and globally connected information systems. This time, it's for all the marbles.

Are there important technical and legal issues with portions of the Microsoft-Novell technology-sharing deal? Unequivocally, yes. Different constituencies each have distinct reservations about different facets of the deal. Open source developers and the GPL license are adversely affected by the patent stand-still agreement; competing Linux distro and open source developers are understandably sore about the spurious patent protections as well as the virtualization and sales cross-collaboration arrangements; and users and governments are rightly worried that the Microsoft-Novell of-

¹ ECMA stands for the European Computer Manufacturers Association.

office suite "interoperability" commitments will limit their options.

Our concerns today focus in fine detail on the latter aspect of the Microsoft-Novell deal, and place things into the larger context of ISO, SOA, and the future of highly advanced and extremely productive information processing chains that span desktops, servers, and devices.

Office Open XML ("*The companies' joint project to develop a facility for converting between OpenDocument, ODF, files and Microsoft/Ecma, EOOXML, files*") is fraught with problems. It offers itself as an "interoperability" solution between Microsoft Office and OpenOffice.org. It is not². Even on the face of it, Microsoft's specification for EOOXML is gravely flawed and impossible for non-Microsoft developers to fully implement. So it will have serious problems being accepted by ISO as an international standard. But if you understand only a few of the severe problems with EOOXML, the anti-competitive underpinnings of this "interoperability" project, and the deal between Microsoft and Novell, show through.

2 Migrating Microsoft's Legacy Files to XML

Everyone agrees that XML is certain to be the next-generation format for documents. XML is after all both the future of the Internet, and, the heartbeat of any SOA. It's not just "the future", it's also the bridge between where we are today (our legacy infrastructure) and what we need to do to get to that future.

Microsoft wants to "pull up" customers into their own proprietary version of XML ("proprietary XML" is undoubtedly a *non sequitur*, but we'll let that slide for the time-being); the ODF community wants to get everyone using its universal & portable implementation of XML; and customers are rightly confused because the Microsoft option looks acceptable, if expensive.

Besides, the "EOOXML or ODF" non-decision decision (as in both are acceptable) might preserve a CIO's career. Assuming people trust Microsoft, and while people agree Microsoft is only looking after customers' interests (as usual), this non-decision decision looks safe. Even if people are skeptical of Microsoft's motives, the well-entrenched belief exists that the Microsoft option is simply unavoidable due to the market deployment power of the Monopolist. And, on the other side, the ODF option (despite its two year lead in the ISO standardization process) still looks a little under-articulated and lacking integration across the business processing chain. Fair enough.

But ODF is credible and offers early adopters today (governments in particular) some really appealing benefits of

perpetual data access, lower cost, and new flexibility in systems procurement (more on SOA below). The benefits really light up for an organization when it successfully gets through a migration and starts originating documents in an open implementation of XML.

The case for ODF is well-understood by those who have announced commitments to it (they know who they are). Yet their first concerns include a) how do I get my legacy Microsoft documents into ODF; b) how do I cope with mixed formats and mixed systems during and even long after I transition my organization; c) how do I cope with irregularities introduced through document round-tripping in regular business processes across different systems in- and outside my organization; and d) how can I use open standard XML documents to shift my development efforts away from application-layer APIs — particularly those owned by a sociopathic company known for cooking a second set of books on the APIs it owns?

This is why the "interoperability" word is out and about. With XML on the horizon, customers are all over interoperability. They are hammering on Microsoft and the ODF community to solve the file format incompatibility problems that have plagued document behaviors inside the Microsoft family of formats and out. Accordingly, Microsoft has fallen in love with the word "interoperability" if not the actions to achieve it.

The mandate driven by customers is that they want server and device side systems to integrate fully with the Windows | MS Office desktop. At the exact point of integration, they want an open and application- and platform-independent XML file format connecting those MSOffice desktops to server and device side systems. In short, information technology customers want full and uncompromised ownership over both their information and their information processes, control over the day to day information and business processes on which their workflows and services are based.

But Microsoft is reluctant to make it easy for customers to leave formats it controls. In truth, Microsoft knows its own secret formats perfectly. *Perfectly*. Any company that could sync perfect conversion fidelity between the billions of binaries and EOOXML, either natively or using a plugin, could easily provide a 100% fidelity with ODF within a few weeks of development. The comparative structural advantages of ODF over EOOXML are so compelling that, if not for the business case of advancing and extending the monopoly, there is technically every reason for Microsoft to have chosen ODF over EOOXML.

For customers "interoperability" means 100% file fidelity from one application to another, either of the same or different kind, regardless of underlying platform. The requirement is not complicated and has been articulated clearly in Massachusetts in the ITD Request for Information about an ODF Plugin for MS Office.

One of the trickiest jobs in Christendom is Microsoft's challenge to "pull up" (as they euphemistically call it) all Microsoft half-a-billion-odd customers into a modern, sup-
portable version or two of their OS, application and file

² *Interoperability* normally imports a sense of a complete absence of barriers. See e.g., ISO/IEC 2382-01, as quoted in Wikipedia: ("*[t]he capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units*").

format technologies. Due to lusty appetites, hard work, unscrupulous leadership and a sociopathic competitive culture, the company has revved the customer base on so many versions that their principal competitor is *themselves*. (A sub-text of the main challenge, that of selling, is the need to master the intricacies of EOL-ing (end-of-life-ing) older versions without drawing attention from regulators or infuriating customers beyond a certain tipping point.)

Maybe Redmond feels it's pushed this game once to often, but we're now witnessing a change in strategy. MSOffice 2007 is a transitional application, very ambidextrous in that it has one EOOXML for legacy Win32 API bound BoBs (billions of binary documents), and a different EOOXML for native, Vista - .NET 3.0 API-bound documents. In fact, it's possible to have the exact same document in MSOffice 2007, one version a native and the other a legacy, with different EOOXML internals and dependencies. What gives? While no one knows for sure, it does look like the point of lockin has moved from that of MS Office-bound documents and business processes to an information processing chain comprised of MSOffice 2007 <=> EOOXML <=> IE 7.0 <=> and the Exchange/SharePoint Hub. It is the E/S Hub where the new lockin occurs because all EOOXML documents are converted here to the Vista - .NET 3.0 API-based dependencies. MS Office is just the head-point of this processing chain.

From the E/S Hub, expect accelerated EOOXML connectivity to MS Live, MSN, MS-ERP, SAP, MSSQL Servers and backend transaction processing systems, Active Directory, collaboration Server and MSOffice Server systems (to name but a few). Expect also that the EOOXML ready VSTO 2005 will grease the skids for migrating existing MS Office-bound business processing systems to the E/S Hub, as well as provisioning the rapid development of EOOXML-ready line of business applications able to fit nicely into the EOOXML-dominated information processing chains. Now that's *integration!* Or, as they say in Redmond, "*Interoperability by Design*".

At this point we have to ask ourselves if competitive, non-Microsoft directed, software systems and services can interoperate within this extraordinary processing chain design? I mean, imagine an OpenOffice desktop, or MAC Office desktop participating in this chain? How about Google, Yahoo, or Amazon Web 2.0 mashup services? Or Oracle backend ERP, relational databases, and transaction processing systems? Or IBM's Lotus Notes? Or maybe BEA's SOA model? Now, do you understand what's at stake?

Oh yeah. Bye bye Adobe.

So what's at work here anyway?

Interoperability (real round-trip interoperability, such as that provided by a universal file format) is important to business processes of all kinds based on the way people move documents across a widening variety of systems. The business process of greatest relevant concern from the standpoint of antitrust law and the effects on competition is an organization's migration of its legacy documents stored in Microsoft Office binary formats to the OpenDocument For-

mat ("ODF"). If such a migration can be accomplished then it can be said that competition with Microsoft's new business processes stack is feasible. But if such migrations are infeasible, Microsoft maintains its vendor lock-in on existing customers - not only in the office suite market but also in the business processes arena.

Yes, the BoBs, again (those "Billions of Binaries"): those billions of legacy Microsoft Office documents stored in binary formats, the BoBs that Microsoft reminds us of so frequently when arguing for adoption of EOOXML as a standard. The Novell-Microsoft-CleverAge file converter is limited to EOOXML-to-ODF conversions and does not directly convert between ODF and the secret Microsoft legacy binary file formats, still used by Microsoft Office for internal processing.

3 The Novell-Microsoft-Clever Age ODF Translator is Intentionally Half-Baked

For at least four reasons, the file converter being developed by Clever Age, paid for by Microsoft, and inserted into the Novell-only version of OpenOffice, will never be capable of full interoperability between Microsoft Office and applications supporting ODF like OpenOffice.org. The reasons have to do with fundamental characteristics of both the EOOXML format as well as the Clever Age-Microsoft technical approach, XSLT. Microsoft knows in advance that XSLT is inadequate.

This comes as no surprise. Bill Gates never bets on anything unless he's holding all the cards. And as long as people are thinking EOOXML is just XML, and that any XML can be easily and universally transformed with XSLT, his three ace's in the hole are safe as can be, (95% desktop marketshare, perfect fidelity conversion BoBs, and a EOOXML bound to both the legacy Windows API and the Vista -.NET 3.0 API internals).

3.1 Other Developers Cannot Feasibly Implement all of EOOXML

Alongside the problem of EOOXML's dependence on obscure RTF instructions, there are a host of tags in the EOOXML specification that can not feasibly be implemented by anyone but Microsoft. Here are only a few examples collected by Ben Langhinrichs <<http://www.geniisoft.com/showcase.nsf/archive/20061027-0829>>:

- **autoSpaceLikeWord95** (Emulate Word 95 Full Width Character Spacing) - pages 1378-1379.
- **footnoteLayoutLikeWW8** (Emulate Word 6.x/95/97 Footnote Placement) - pages 1416-1417.
- **lineWrapLikeWord6** (Emulate Word 6.0 Line Wrapping for East Asian Text) - pages 1426-1427.
- **mwSmallCaps** (Emulate Word 5.x for Macintosh Small Caps Formatting) - pages 1427-1429.
- **shapeLayoutLikeWW8** (Emulate Word 97 Text Wrapping Around Floating Objects) - pages 1442-1443.
- **suppressTopSpacingWP** (Emulate WordPerfect 5.x Line Spacing) - pages 1462-1464.
- **truncateFontHeightsLikeWP6** (Emulate

WordPerfect 6.x Font Height Calculation) - pages 1467-1468.

■ **useWord2002TableStyleRules** (Emulate Word 2002 Table Style Rules) - pages 1481-1482.

■ **useWord97LineBreakRules** (Emulate Word 97 East Asian Line Breaking) - pages 1482-1483.

■ **wpJustification** (Emulate WordPerfect 6.x Paragraph Justification) - pages 1483-1485.

These are deprecated features from old Microsoft Office versions that are being carried forward only for the purpose of backward compatibility. Nearly all such tags in the EOOXML specification are accompanied by a bit of repeated "guidance" that only a cynic might view with some sense of hilarity or which, in fact, prompts belly laughs from the file format *cognoscenti*: *"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."*

The EOOXML specification and file format unquestionably contain hundreds of processing instructions that in the aggregate can only be processed feasibly by Microsoft Office. It is a single-vendor file format, a file format that treats software as an endpoint rather than as a router of information in any business process software stack other than Microsoft's own tightly-integrated stack.

Those application-specific tags might as well be binary blobs as far as any non-Microsoft application is concerned. The tags' functionality is concealed in the unspecified Microsoft Office application layer rather than being specified in the EOOXML standard adopted by ECMA. The existence of such tags belies any Microsoft claim that EOOXML was designed for interoperability, a claim expressly made in the Ecma final draft specification (http://www.ecma-international.org/news/TC45_current_work/TC45-2006-50_final_draft.htm), Part 1: *"The goal is to enable the implementation of the Office Open XML formats by the widest set of tools and platforms, fostering interoperability across office productivity applications and line-of-business systems, as well as to support and strengthen document archival and preservation, all in a way that is **fully compatible** with the large existing investments in Microsoft Office documents."* (Bold letters added by the authors).

Right, just blink past the fact that neither the specifications for the legacy binary formats nor the APIs used for the conversions will be found in the EOOXML specification; they remain a closely-guarded Microsoft secret. All of

that "compatibility" with the binary formats is reserved for the exclusive use by the products of a single vendor, Microsoft.

You can begin to see the *modus operandi* resolving clearly. Through systematic obscurity and audacity's sleight of hand, Microsoft would have ISO grant it an exclusive monopoly on migrating legacy Microsoft Office files to XML. To what flavor of XML? Why, to Microsoft's own. Never mind that Microsoft refused to participate in developing the OpenDocument XML international standard that ISO has already adopted. Never mind that Microsoft refused to provide native support for OpenDocument in Microsoft Office. Never mind that since February of 2003, ODF has had the capability of rendering perfectly any and all BoBs, application specific processing instructions and binary blobs included.

Blink and you'll end up back in 1995: software as an end-point.

Obviously, if one follows the logic employed by Microsoft and Ecma, software consumers will need yet a third and maybe a fourth file format implementation of XML for office documents. One to translate WordPerfect Office files to XML, and yet another for IBM's Lotus Notes. How then is the new age of XML office formats any substantial improvement over the cacophony of binary formats that have stymied full interoperability these many decades?

The inclusion of hundreds of vendor-specific tags is not EOOXML's only weakness, of course. As only three more examples, the EOOXML specification also suffers from inappropriate reliance on Windows operating system bitmasks, on Microsoft application-specific bug work-arounds (<http://www.robweir.com/blog/2006/10/leap-back.html>), and a Microsoft proprietary custom XML parser (<http://www.schwieb.com/blog/2006/12/05/conversion-factors/>).³

There is scant wonder that Microsoft found it necessary to pay Novell a large sum of money for even a partial implementation of EOOXML in OpenOffice.org.

3.2 EOOXML is a Vendor Lock-in Format

The sheer complexity of the EOOXML specification, some 6,000 pages, atop its reliance on application-specific tags, Windows bitmasks, and Office bug work-arounds translates to vendor lock-in. It means that no one but Microsoft will be able to support the specification anywhere near completely. That has obvious consequences, as explained by IBM's Bob Sutor⁴ (<http://sutor.com/newsite/blog-open/?p=1145>): *"Fully and correctly implementing [EOOXML] will require the cloning of a large portion of Microsoft's product. Best of luck doing that, especially since they have over a decade head start. Also, since they have avoided using industry standards like SVG and MathML, you'll have to re-implement Microsoft's flavor of many things. You had better start now. So therefore I conclude that while Microsoft may end up supporting most of [EOOXML] (and we'll have to see the final products to see how much and how correctly), other products will likely only end up supporting a subset."*

³ For a more detailed critique of EOOXML within the context of a Service-Oriented Architecture, see the article "IBM's potential MS-Office killer to roll out by year's end" (<http://talkback.zdnet.com/5208-10532-0.html?forumID=1&threadID=13561&messageID=273162&start=-4>) by Gary Edwards.

⁴ IBM's Rob Weir has since published a satirical counterpoint to Sutor's earlier article (<http://www.robweir.com/blog/2006/12/how-to-write-standard-if-you-must.html>).

That means that other products and software, in practice, will NOT be able to understand arbitrary [EOOXML] that might be thrown at them. There is just too much. Therefore they will only create a bit that they need and send that off. Send it off to whom? The only software that might understand it, namely Microsoft Office.

So this is how I see this playing out: [EOOXML] will be nearly fully read and written by Microsoft products, but only written in subset form by other software. This means that data in [EOOXML] form will be largely sucked into the Microsoft ecosystem but very little will escape for full and practical use elsewhere.

(See also Sutor's later article including graphics depicting the above concepts, discussed in more general terms and exploring the difference in meaning between, "interoperability" and "intraoperability". <<http://sutor.com/newsite/blog-open/?p=1260>>).

It is accurate only in theory for Microsoft to claim that the EOOXML file format can be implemented by anyone. As a practical matter, full implementation by other developers will not occur. Unless perhaps, Novell will commit to reverse engineering all of those features of previous Office versions invoked by those hundreds of tags, Windows bitmasks, and Microsoft Office bug work-arounds.

3.3 Novell and Microsoft Do Not Intend to Achieve Interoperability

In light of the gratuitous complexity of the EOOXML specification, it is especially significant that Microsoft's Steve Ballmer said of EOOXML-to-ODF file converter being jointly developed by Novell and Microsoft <<http://www.eweek.com/article2/0,1895,2050848,00.asp?kc=EWEWEMNL103006EP17A>>: "Nor will the collaboration team attempt to build file converters that can make files 100 percent compatible between the two file formats", he said. "But it will achieve the level of interoperability that customers can work with", he said.

Novell's OpenOffice.org development team leader Michael Meeks agrees: "what Ballmer says is true in a way; sure - it's likely that 100% interoperability between any two non-trivial apps is never going to be possible. That of course is misleading, as long as interop is -good-enough- people are unlikely to miss the last 2% (or whatever). At some people the return on investment of adding the next 1% becomes so low that people don't bother :-)".

However, at least one developer who has actually grappled with converting Microsoft's binary file formats into OpenDocument disagrees. The OpenDocument Foundation's Gary Edwards, who also represents the OpenOffice.org Community in the OASIS OpenDocument Technical Committee, says that Mr. Meeks' "or whatever" is substantially larger than 1 or 2 percent when migrating Microsoft binaries to OpenDocument using EOOXML as an intermediary format:

"The ODF transformers Ballmer and Novell both refer to are actually the same CleverAge/Microsoft ODF Translator filters that are based on XSLT with supplemental C #

routines - routines that prove beyond doubt that EOOXML is not XSLT-ready. The CleverAge converter will be the central part of the plug-in being developed for Novell OpenOffice. Ballmer is right in that no one will ever get anywhere near 100 percent fidelity that defines 'interoperability' using XSLT methods."

The reason why XSLT will never work in this situation is that XSLT needs highly structured 'XPath-perfect' XML to perfect a transformation. ODF was written to be this XPath-perfect XML structure that XSLT needs. However, EOOXML is anything but.

"The structural deficiencies in EOOXML that render XSLT next to useless (say an optimal 60% fidelity as compared to the lossy 85% fidelity or so of the traditional reverse engineered filters in OpenOffice.org considered by all who have ever tried to be an extraordinary achievement), center on the loosely structured 'styles' model (presentation) in EOOXML. And this loose half structure is in turn directly related to the simple layout engine of MSOffice. (Simple in comparison to the rich and very complex layout engine in OpenOffice)."

"The Ballmer statement that neither Novell, Microsoft, nor CleverAge will attempt to hit 100% fidelity is in effect a statement that they intend on sticking to XSLT as the transformation method."

Moreover, when it comes to software as a router of information in a business process, where the same data must be read and written by a series of different applications, only full interoperability is workable. A business process that allows loss of data on every trip from one app to another is not "interoperability." Saying "the level of interoperability customers can work with" makes about as much sense in context as saying "partially pregnant." Indeed, the Ecma white paper on EOOXML <<http://openxmldeveloper.org/archive/2006/10/24/764.aspx>> tells us in section 4.6 that only "high fidelity" translations will work when migrating the legacy Microsoft binary formats to XML:

"High Fidelity Migration

OpenXML is designed to support all of the features in the Microsoft Office 97-2003 binary formats. It is difficult to overstate the difficulty of accomplishing this goal, and the consequent uniqueness of OpenXML in doing so. ... OpenXML is intended to permit future editing or manipulation at the same level of abstraction available to the original creator[.]"

But only if Microsoft Office is the editing tool. See also *id.*, section 4.7 ("Integration with business data").

3.4 Market Requirements Will Not Be Fulfilled by the Joint Development Project

Despite Mr. Ballmer and Novell's claim that lossy "good enough" migration from Microsoft formats to ODF is sufficient, laws around the world effectively mandate 100 percent fidelity in migrating binary documents to XML. See e.g., E-SIGN Act, 15 U.S.C. 7001(d)(1)(B) (electronically preserved records must "accurately reflect[] the information set forth in

the contract or other record" and be "in a form that is capable of being accurately reproduced for later reference, whether by transmission, printing, or otherwise"); Sarbanes-Oxley Act, 15 U.S.C. 7261(b) (financial information must "not contain an untrue statement of a material fact").

The "good enough" lossy migration of legacy records to ODF contemplated by the Novell-Microsoft deal is not nearly good enough in such a legal environment. A single user may be able to convert a single file and then manually check to see that no data was lost. It is an entirely different situation when documents must be successively round-tripped or in the case of mass- or wholly-automated conversions of files.

4 The Market Requirement for Interoperability Can Be Fulfilled

Novell's Miguel de Icaza evaded rather than answered the following important question that Pamela Jones asked:

"Also, please tell us all what is wrong, in your view, with the Sun or the ODF Foundation [sic] solutions and can you please explain to us the differences between what yours will do and what they do?"

Icaza answered: *"Michael [Meeks], the head of our OpenOffice team—which happens to be visiting town—says: 'the ODF Foundation Solution is not free software; The Sun one is unpublished' "*

Contrary to Mr. Meeks' statement, the OpenDocument Foundation has not yet made a final decision on licensing of its file conversion products under development, but is decidedly leaning toward, and has entered into discussions with ODF community leaders, licensing under the free and open source software GPL license.

As to the differences among the solutions, Sun's solution is a C# wrapper around OpenOffice.org that uses OOo's traditional file import-export facilities. According to Edwards, it is able to achieve only around 85 percent fidelity at best in translating between Microsoft binaries and OpenDocument. At the heart of this method is the same filter process now used in OpenOffice 2.0. There is no edge where "improvement" might occur. The Microsoft/Clever Age group is working on XSLT-C # filters that translate between EOOXML and OpenDocument. Given the structural challenges for any XSLT based method, CleverAge ought to get a technology of the decade prize if they can hit at best 60 percent fidelity.

The Foundation tools however are designed to achieve high-fidelity by bypassing the EOOXML trap, working directly with the Microsoft Office modified RTF conversion formats. One Foundation tool is the ODF Plug-in for Microsoft Office, a plug-in dubbed "da Vinci" that uses the same Office internal APIs Microsoft uses for file formats it supports natively. The other is a standalone ODF XML Infoset API that can be used in applications (on the server-side, too) that support ODF to import and export Microsoft binary formats. A third is a da Vinci plugin for OpenOffice.org. And a fourth is a "interoperability wizard" designed to guarantee that OpenOffice workstations

caught up in a MSOffice driven business process - workflow will produce perfectly interoperable documents.

Edwards said that the da Vinci plug-in and OpenDocument Infoset API will be able to achieve 100 percent fidelity in translating between Microsoft binary formats and ODF for applications that correctly implement OASIS OpenDocument Format specification version 1.2.

Version 1.1 of the ODF specification focuses on assistive technologies, and was recently accepted under a vote of the OASIS OpenDocument Technical Committee. ODF version 1.2 is the equivalent of interoperability on steroids, and will have the attention of the TC in January and February of 2007. Edwards advises that Microsoft Office 2007 users change the suite's default settings and save their work in the traditional binary formats rather than EOOXML in order to simplify migrations to ODF after the Foundation's tools are released. But he says there are even stronger reasons for avoiding EOOXML: *"Once Microsoft Office-bound business processes are moved over to EOOXML, those processes are ready for migration to the Microsoft Exchange/Sharepoint Hub centered information processing chain. Organizations who fall into this 'business process migration' trap will not be coming off the Microsoft processing chain anytime soon. They might as well cut a long-term leasing deal with Microsoft good for at least the next fifteen years. They should also make certain the deal covers the expense of the eight different desktop productivity applications and nine different server-side systems needed to expand and enhance these highly productive, but EOOXML chain-bound, business processes."*

The Foundation's tools are the only ones thus far brought to public attention that promise full interoperability between Microsoft Office (and its binary formats) and applications that support ODF. Foundation developers began with the last public documentation of the Microsoft Office file conversion APIs before Microsoft decided all future enhancements would be secret, then from there the Foundation deduced the nature of the secret tags. Our da Vinci plug-in adds native support for ODF to Microsoft Office, something that none of the other forthcoming tools will offer. Microsoft Office can then be used in batch mode as a "pump" to migrate documents from the legacy Microsoft binary file formats to OpenDocument format with unmatched fidelity.

The salient point in the Foundation's strategy is this; boost the creation of alternative ODF information processing chains that can compete with the EOOXML dominated chains. MSOffice bound business processes are going to migrate to highly productive E/S style hubs. Nothing is going to stop or alter this migration. The only question is, will these hubs be EOOXML or ODF ready? So it doesn't make sense to totally disrupt current workflows and services to re write these business processes to ODF ready desktop alternatives. What does make sense is to first, get these MSOffice bound business processes into ODF. Get the documents and document/data flows into ODF. This is the job of da Vinci. Second, migrate those MSOffice bound business processes to ODF ready Hubs! This is the primary

purpose of our ODF InfoSet Engine - API; a lightweight server and device side engine that can ODF automate applications instantly.

Full fidelity in such migrations is feasible only because ODF was intentionally designed for interoperability and includes features specifically designed for interoperability with Microsoft Office. ODF implements a system whereby unrecognized and foreign elements and attributes are preserved by a conforming application. *See generally* OpenDocument specification section 1.5 <<http://develop.opendocumentfellowship.org/spec/>>. (But note that occurrences of "should" and "may" in that section are expected to change to "must" in version 1.2 and further interoperability refinement of the specification is planned.).

Those requirements (generally referred to by the ODF development community as *foreign elements* and *unknown elements*, or simply as the *Microsoft tags*) moot Microsoft's claim that a separate standard is needed to ensure compatibility with those billions of legacy Microsoft Office binary files.

The da Vinci plug-in has been developed well beyond proof of concept stage and has been demonstrated for multiple government bodies around the world, including one demonstration in Europe that was attended by Microsoft officials; who publicly announced their lossy ODF XSL Transformer project the very next day, a somewhat inconsistent action for a company that claims to be a champion of software interoperability.

5 The Novell-Microsoft Deal Raises Antitrust Issues

When the Novell-Microsoft deal was announced, and to this day, both companies proclaimed that customers were demanding interoperability between the Microsoft and ODF stacks of software. That, and the need to share virtualized operating system technology, are the major justifications for the deal, both companies said. However, it is also beyond question that both companies realized their deal will not deliver what the market requires as to interoperability. Instead, the deal cements non-interoperability into place for five more years, actually degrading the level of fidelity presently achievable by OpenOffice.org's import/export filters by diverting customers from conversions of the binary formats onto even more lossy XSL Transformation tools.

Any candid discussion of the Novell-Microsoft deal must

begin by asking both companies: [i] why they are refusing to supply what the market requires; and [ii] why they have cemented their joint refusal into a binding contract. Both companies are very apparently aware that full interoperability is feasible. But the Novell-Microsoft deal so far looks less like an "interoperability deal" and far more like a prohibited agreement not to compete and an unlawful allocation of the office productivity software market, a Sherman Act combination in restraint of trade.⁵

Under the deal, Novell is allocated the market for OpenDocument-based office productivity software and Microsoft is allocated the market for corresponding EOOXML-based software. The apparent conspiracy has each company's agreed share of the market shielded from competitors by an undefined patent thicket publicly asserted by the other company in the form of a covenant not to sue the other's paying customers. The highly-publicized covenants not to sue implicitly threaten anyone who uses other products with expensive patent infringement litigation.

Novell's knowledge that far better interoperability is feasible is amply demonstrated by the fact that two days before the Novell-Microsoft deal was signed and announced, Novell head-hunted the OpenDocument Foundation's chief technology officer, and world re known document conversion - document processing expert, Florian Reuter. Novell unquestionably knew what he was working on before it signed on the dotted line with Microsoft. Moreover, Microsoft made no secret of the fact that it could easily provide full native support for OpenDocument in Microsoft Office. Former Massachusetts Secretary of Administration & Finance Eric Kriss: "... *said technical people at Microsoft told him it would be "trivial" to add support for ODF to the new Office 2007. The resistance to doing so came from the vendor's business side, according to Kriss*". < <http://www.computerworld.com/action/article.do?command=viewArticleBasic&articleId=273815&pageNumber=2>>.

It thus appears that both Novell and Microsoft knew that full interoperability was achievable and knew it was a market requirement, but conspired to ensure that software consumers would not be provided with the full interoperability they required, knowingly using joint development of inadequate XSL transformation tools as antitrust "cover" for an agreement to maintain separate non-interoperable software

⁵ *Palmer v. BRG of Georgia, Inc.*, 498 U.S. 46, 50 (1990) (per curiam) <<http://caselaw.lp.findlaw.com/scripts/getcase.pl?court=US&vol=498&invol=46>> ("*[s]uch agreements are anticompetitive regardless of whether the parties split a market within which both do business or whether they merely reserve one market for one and another for the other*"); see also 15 U.S.C. 1: <http://caselaw.lp.findlaw.com/cascode/uscodes/15/chapters/1/sections/section_1.html>: "*Every contract, combination in the form of trust or otherwise, or conspiracy, in restraint of trade or commerce among the several States, or with foreign nations, is declared to be illegal. Every person who shall make any contract or engage in any combination or conspiracy hereby declared to be illegal shall be deemed guilty of a felony, and, on conviction thereof, shall be punished by fine not exceeding \$10,000,000 if a corporation, or, if any other person, \$350,000, or by imprisonment not exceeding three years, or by both said punishments, in the discretion of the court.*"

The courts have imposed a requirement that restraints of trade thus prohibited must be "unreasonable." A finding that a Sherman Act restraint of trade is unreasonable may be "*based either (1) on the nature or character of the contracts, or (2) on surrounding circumstances giving rise to the inference or presumption that they were intended to restrain trade and enhance prices.*" *NCAA v. Board of Regents of Univ. of Okla.*, 468 U.S. 85, 103 (1984), quoting *National Society of Professional Engineers v. United States*, 435 U.S. 679, 692 (1978).

stacks and not to compete in the divided market. The Microsoft-Clever Age-Novell EOOXML-to-ODF transformer admittedly will not achieve full interoperability and therefore is not what the market requires. An agreement not to supply what the market requires in order to divide a market between two companies is blatantly anti-competitive.

Novell could still salvage some of its shredded goodwill among the free and open source software community by solidly — and publicly — backing the OpenDocument Foundation solutions. Novell might also thereby avoid an otherwise predictable wave of antitrust litigation both in the U.S. and in Europe that Novell is far less likely to survive than Microsoft. Novell does, after all, have a freshly acquired set of deep pockets that will attract the sharks. But regardless of its course of action, Novell should stop spinning its deal with Microsoft as an "interoperability deal." The real basis of the deal is embarrassingly transparent and the company can only shed further credibility by continuing the charade.

It should be remembered that in a business process, a file format is every bit as much a communications protocol as a method of storing data. The European Commission's DG Competition previously found <http://europa.eu.int/comm/competition/antitrust/cases/decisions/37792/en.pdf> that Microsoft's refusal to disclose its communications protocols for Windows and Windows Server to competitors was an antitrust violation. But DG Competition did not limit its remedial order to just the Windows communications protocols; it ordered Microsoft to refrain "*from any act or conduct having ... equivalent object or effect*," an issue raised by the European Committee for Interoperable Systems when it filed its complaint <http://www.groklaw.net/article.php?story=20060224151804474> with DG Competition because of Microsoft's refusal to support ODF and its refusal to disclose the specifications for its Office binary file formats.

As Novell was one of the companies that instigated and participated as a party in the European Windows antitrust litigation, it seems more than somewhat incongruous for Novell to subsequently become a co-conspirator in Microsoft's bid to use the secrecy of its office file formats and file conversion APIs to thwart full interoperability with software that supports ODF. That is especially so in a deal also intended to share technology for virtualization of each companies' operating system. What good is virtualization of operating systems if the business applications running on those concurrent operating systems can not interoperate? And is that not an "equivalent object or effect" of Microsoft's refusal to disclose its Windows communications protocols? EOOXML is open in name only.

Novell knows full well what the law has to say on this issue. Indeed, Microsoft paid Novell 536 million to withdraw from further participation in the European Commission's antitrust case. Having successfully advocated to establish the injunction, Novell is in a poor position to contest its implementation. An argument that "we couldn't afford to turn down a second pay check" is not a strong defense to an antitrust conspiracy case, where a defendant is responsible not only for its own but also for its co-conspirator's acts and omissions.

6 Conclusion - The XML Cold War in a Changing Market for Software

To fully grasp the vendor lock-in and legal bind Microsoft and Novell are devising, it is important to understand how ISO/IEC ("ISO") adoption of EOOXML one or two years from now would result in a legally sanctioned extension of Microsoft's monopoly in office document formats. Through its technology-sharing arrangement with Novell and its elaborate messaging around "interoperability," Microsoft audaciously seeks to reassert the old lock-ins while draping its behavior in the "open" language that is today *in vogue*.

The fundamental legal problem is that ISO adoption of EOOXML as an international standard would make its use mandatory in many situations, in both the private and government sectors. By the same token, its use would be prohibited in many of the same situations if not approved by ISO. (*See generally*, Article 2 of the Agreement on Technical Barriers to Trade <http://www.wto.org/english/res_e/booksp_e/analytic_index_e/tbt_01_e.htm#article2> and Article VI of the Agreement on Government Procurement <http://www.wto.org/english/res_e/booksp_e/analytic_index_e/gpa_02_e.htm#article6>).

Those treaties are intended, *inter alia*, to stimulate competition by promoting the development of *de jure* open standards, eliminating multiple standards where a single standard will suffice, and requiring use of adopted standards. Therefore, the ability of non-Microsoft developers to implement EOOXML in their software is a crucial issue in assessing the suitability of EOOXML as a candidate international standard.

But to comprehend the war raging over OpenDocument and EOOXML, one must first understand that the office productivity software market's ground rules are in a state of flux. We live in the age of the Internet. The age of universal access and exchange. The age of universal connectivity and collaborative computing. Things are changing. In the era we are leaving behind, office software was designed as an *end point*. The way to achieve interoperability was for everyone in an office and those exchanging files with that office to use the same software. Different vendors' programs used different and incompatible file formats. Exchanges and info streams were entirely API bound. It was a Tower of Babel situation that ultimately resulted in one vendor's software — Microsoft Office — attaining a monopoly largely through a combination of file format incompatibilities and bundling of the software with new computers.

But even as Microsoft achieved dominance in the office suite market, forces were at work that would undermine its dominance. One such factor was the ascendancy of the open standards-based Internet and ubiquitous networking. Another major factor was a growing crisis of complexity <http://www-128.ibm.com/developerworks/library/wsmigratesoa/>: "*Over the last four decades, software architectures have attempted to deal with increasing levels of software complexity. But the level of complexity contin-*

ues to increase, and traditional architectures seem to be reaching the limit of their ability to deal with the problem. At the same time, traditional needs of IT organizations persist; the need to respond quickly to new requirements of the business, the need to continually reduce the cost of IT to the business, and the ability to absorb and integrate new business partners and new customer sets, to name a few. As an industry, we have gone through multiple computing architectures designed to allow fully distributed processing, programming languages designed to run on any platform, greatly reducing implementation schedules, and a myriad of connectivity products designed to allow better and faster integration of applications. However, the complete solution continues to elude us..."

Now you find more complex environments. Legacy systems must be reused rather than replaced, because with even more constrained budgets, replacement is cost-prohibitive. You find that cheap, ubiquitous access to the Internet has created the possibility of entirely new business models, which must at least be evaluated since the competition is already doing it. Growth by merger and acquisition has become standard fare, so entire IT organizations, applications, and infrastructures must be integrated and absorbed. In an environment of this complexity, point solutions merely exacerbate the problem, and will never lead us out of the woods. Systems must be developed where heterogeneity is fundamental to the environment, because they must accommodate an endless variety of hardware, operating systems, middleware, languages, and data stores. The cumulative effect of decades of growth and evolution has produced severe complexity. With all these business challenges for IT, it is no wonder that application integration tops the priority list of many CIOs.

To meet the challenge posed by that crisis of complexity, a new Service Oriented Architecture (SOA) began to emerge. SOA is in large part built around XML, the open and human-readable eXtensible Markup Language as the building block for future expansion. As to data stored in relevant legacy file formats, SOA requires endless repetitions of the following workflow: [i] identify the location and form of the data specified by the requesting application; [ii] convert the data from the legacy file formats into a *validXML* format; [iii] programmatically extract the portions of the data to be repurposed; [iv] input the data to an XML transformation process; [v] output the data in the requested XML format; and [vi] serialize that data to the specified application in the workflow for subsequent processing.

Notice in that somewhat over-simplified workflow that applications are waypoints, i.e., routers of information, rather than end points. Applications architected in the days of the sneaker net such as Microsoft Office have distinct disadvantages. Flawless migration of data between formats is integral to a SOA process. Workflows incorporating such steps are commonly referred to as business processes.

Microsoft is able to read the tea leaves of industry trends as well as anyone else. Not surprisingly, Microsoft is developing its own stack of proprietary business processes software. That stack, the infamous information processing

chain, uses EOOXML not only as a file format, but also as a communications protocol among the various applications. So EOOXML is far more than just a file format for an office suite. Like OpenDocument, it is designed as an interoperability tool for business processes within solutions such as a SOA. As will be shown below, the difference is that EOOXML is a proprietary closed specification. On the other hand, ODF is completely open.

It is also no accident that the war between proponents of OpenDocument XML and of Microsoft Office 2003 XML (an antecedent of EOOXML) first came to wide public attention <<http://www.groklaw.net/article.php?story=20050330133833843&query=donnybrook>> during the design of a Service Oriented Architecture by the both the EU Valoris Report, and the Commonwealth of Massachusetts with its Information Technology Division's Enterprise Technical Reference Model. The process of designing that architecture created the need to choose which XML file format would be the destination format for Microsoft Office legacy binary file formats. For a variety of reasons, Massachusetts ITD chose OpenDocument as that file format. Of course, SOA is only part of the story. A growing host of Web and Web-distributed applications support OpenDocument, including Software as a Service and next-generation Web 2.0 <<http://www.opendocumentfellowship.org/applications>>.

So here we are. It's decision time. Microsoft is offering a very compelling, fully populated, EOOXML based information processing chain that greatly leverages the existing monopoly base of installed MSOffice desktops, bound business processes, and locked in legacy BoBs. It is no doubt a business objective designed exactly to extend the monopoly from the desktop to servers, devices, and beyond. Awesome to say the least. Incredibly audacious.

ODF on the other hand is designed and destined to be a universal file format transcending applications, platforms, archival needs, and information technology advances unknown. It is a universal file format servicing the needs of information domains as diverse yet begging interoperable connectivity and exchange as: desktop productivity environments, enterprise publication, content and archive management systems, SaaS, SOA, the Web 2.0, and beyond.

EOOXML decidedly breaks the promise of XML; the ease of universal transformation and cross-generation interoperability. But fear not, perfect interoperability between ODF and the billions of binary documents is available in ODF 1.2. And then there is the ODF information processing chain promise of the Foundation's da Vinci plugin (its *ODF Plugin for MS Office*) and its *ODF InfoSet API*.

ODF is ready. It's about 2 years ahead of EOOXML in the ISO standardization process. Let the battle begin!

Acknowledgment

Marbux, a retired lawyer, who is a volunteer member of the OpenDocument Fellowship, contributed this article's legal analysis.