

Response to Comments by Mr. Gautam

Clause No. / Subclause No. / Annex (e.g. 3.1)	Paragraph / Figure / Table / Note (e.g. Table 1)	Type of comment	Comment (justification for change)	Proposed change	Response
Office Open XML Overview	-	ge	This document is not listed as part of the Ecma 376 standard in the Forward to Part I “Fundamentals” and its status whether informative or normative is not explicitly stated.	Clarify the status of this Overview document. If it is merely a promotional whitepaper about Ecma 376, then it should not be included in the published standard.	It is agreed that the Overview document is non-normative, however it is a useful paper to the implementer and consistent with similar information found in other standards. An appropriate disposition recommendation will be taken by Ecma as part of the response to this comment.
OfficeOpenXML-DrawingMLGeometries.zip	-	ge	There is no explicit indication given as to whether this annex is informative or normative. See ISO Directives, Part 2, section 5.2.6	Clarify the status of this annex	This annex is non-normative as stated in Part 4, Annex E of the specification document.
OfficeOpenXML-DrawingMLGeometries.zip	-	ge	This annex was not provided in a humanly-readable format as required by JTC1 Directives 8.3.5 and Annex H	The annex should be provided in a humanly readable, lined-numbered format so it can be referenced and cited. Additionally, an electronic machine readable version can be provided according to Annex H	The annexes are text files that contain XML source and/or XML schemas, and are intended for consumption by tools and not by human readers. Zip files containing multiple documents are allowed as described in Annex H4.4.2. Further, Annex H.4.2 clearly allows for alternative formats when documents are provided for process, not human, consumption.
OfficeOpenXML-RELAXNG.zip	-	ge	There is no explicit indication given as to whether this annex is informative or normative. See ISO Directives, Part 2, section 5.2.6	Clarify the status of this annex	This annex is non-normative as stated in Part 4, Annex B2 of the specification document.

OfficeOpenXML-RELAXNG.zip	-	ge	This annex was not provided in a humanly-readable format as required by JTC1 Directives 8.3.5 and Annex H	The annex should be provided in a humanly readable, lined-numbered format so it can be referenced and cited. Additionally, an electronic machine readable version can be provided according to Annex H	This annex is informative as stated in Part 2, Annex E line 2 of the specification document and is provided specifically for electronic consumption. The human readable version of this information is fully contained within the 5 parts of the specification.
OfficeOpenXML-SpreadsheetMLStyles.zip	-	ge	There is no explicit indication given as to whether this annex is informative or normative. See ISO Directives, Part 2, section 5.2.6	Clarify the status of this annex	This annex is normative as stated in Part 4, Annex D of the specification document.
OfficeOpenXML-SpreadsheetMLStyles.zip	-	ge	This annex was not provided in a humanly-readable format as required by JTC1 Directives 8.3.5 and Annex H	The annex should be provided in a humanly readable, lined-numbered format so it can be referenced and cited. Additionally, an electronic machine readable version can be provided according to Annex H	The annexes are text files that contain XML source and/or XML schemas, and are intended for consumption by tools and not by human readers. Zip files containing multiple documents are allowed as described in Annex H4.4.2. Further, Annex H.4.2 clearly allows for alternative formats when documents are provided for process, not human, consumption. This annex is normative as stated in Part 4, Annex D of the specification document.
OfficeOpenXML-XMLSchema.zip	-	ge	There is no explicit indication given as to whether this annex is informative or normative. See ISO Directives, Part 2, section 5.2.6	Clarify the status of this annex	This annex is normative as stated in Part 4, Annex A of the specification document.

OfficeOpenXML-XMLSchema.zip	-	ge	This annex was not provided in a humanly-readable format as required by JTC1 Directives 8.3.5 and Annex H	The annex should be provided in a humanly readable, lined-numbered format so it can be referenced and cited. Additionally, an electronic machine readable version can be provided according to Annex H	The annexes are text files that contain XML source and/or XML schemas, and are intended for consumption by tools and not by human readers. Zip files containing multiple documents are allowed as described in Annex H4.4.2. Further, Annex H.4.2 clearly allows for alternative formats when documents are provided for process, not human, consumption. This annex is informative as stated in Part 2, Annex E line 2 of the specification document.
-----------------------------	---	----	---	--	--

OpenPackagingConventions-RELAXNG.zip	-	ge	This annex was not provided in a humanly-readable format as required by JTC1 Directives 8.3.5 and Annex H	The annex should be provided in a humanly readable, lined-numbered format so it can be referenced and cited. Additionally, an electronic machine readable version can be provided according to Annex H	<p>The annexes are text files that contain XML source or XML schemas, and are intended for consumption by tools and not by human readers. Specification document is already very exhaustive and adding summaries for each schema separately will make this document unreviewable in the specified period. However, where ever there was a need schema's are explained within the specification document as in the case of SpreadsheetMLStyles in sections 3.8.7 and 3.8.40 of Part 4 and DrawingMLGeometries in sections 5.1.11.18 and 5.1.11.19 of Part 4. Additionally schemas are explained in detail throughout part 4 and 5 of the specification document.</p> <p>Zip files containing multiple documents are allowed as described in Annex H4.4.2. Further, Annex H.4.2 clearly allows for alternative formats when documents are provided for process, not human, consumption. This annex is non-normative as stated in Part 2, Annex E line 2 of the specification document.</p>
OpenPackagingConventions-RELAXNG.zip	-	ge	There is no explicit indication given as to whether this annex is informative or normative. See ISO Directives, Part 2, section 5.2.6	Clarify the status of this annex	This annex is non-normative as stated in Part 2, Annex E line 2 of the specification document.

OpenPackagingConventions-XMLSchema.zip	-	ge	There is no explicit indication given as to whether this annex is informative or normative. See ISO Directives, Part 2, section 5.2.6	Clarify the status of this annex	This annex is normative as stated in Part 2, Annex D line 3 of the specification document.
OpenPackagingConventions-XMLSchema.zip	-	ge	This annex was not provided in a humanly-readable format as required by JTC1 Directives 8.3.5 and Annex H	The annex should be provided in a humanly readable, lined-numbered format so it can be referenced and cited. Additionally, an electronic machine readable version can be provided according to Annex H	The annexes are text files that contain XML source or XML schemas, and are intended for consumption by tools and not by human readers. The specification document is the human readable form of the annex, and it is already exhaustive. Adding summaries for each schema separately will make this document unreviewable in the specified period. Wherever there was a need schema's are explained within the specification document as in the case of SpreadsheetMLStyles in sections 3.8.7 and 3.8.40 of Part 4 and DrawingMLGeometries in sections 5.1.11.18 and 5.1.11.19 of Part 4. Additionally schemas are explained throughout part 4 and 5 of the specification document.
Part 1, Appendix		te	The reference given for the Zip format does not provide a date or version, though this specification is frequently revised,	Reference should be made to a particular dated and labeled version.	This may be provided as an editorial change to the specification during the BRM.
Part 1, Forward	line 2	ed	DIS 29500 is a multi-part document, not a multi-part Standard, i.e., the individual parts of this Standard are not themselves standards.	Correct the terminology to correctly reflect the status of DIS 29500.	Issues of an editorial nature will be addressed after the Ballot Resolution Meeting by the specification Editor.

Part 1, Section 10.1.2	line 20	te	Reference is made to material in Part 12, Clause 12. Although a clause of that number does exist, it does not contain the material 10.1.2 references it for.	Correct the reference to point to the correct clause.	Clause 12 of Part 5 is all about inclusion of newer namespaces, which is only intended to describe extensibility. It gives an option of including newer namespaces under the current standards. Line 12 and Examples mentioned indicate the same.
Part 1, Section 11.3.1	lines 15-17	te	This is requiring that a conforming OOXML consumer also be able to understand a specified list of other document formats, including proprietary ones such as MHTML and RTF, and for conforming producers to understand how to convert these formats to OOXML.	Change lines 3-5 to read, "An alternative format import part allows content specified in an application-defined alternate format to be embedded directly in a WordprocessingML document..."	The sentence : "An alternative format import part allows content specified in an alternate format (HTML, MHTML, RTF, earlier versions of WordprocessingML, or plain text) to be embedded directly in a WordprocessingML document in order to allow that content to be migrated to the WordprocessingML format." implicitly means that the consumer or producer defines the alternate file format. While the commenter may not prefer to use the example formats provided, the examples are used to illustrate what is possible, rather than what is required. This is not an issue.

Part 1, Section 12.3.5	-	te	This binary part is said to be used for the storage of “arbitrary user-defined data”. No further detail is given as to what user action would trigger the use of this “user-defined” data. Without further definition, no interoperability of this feature is possible.	Fully define the use of Custom Property Part	The use of custom property part is determined by the user who has decided to use it. This part provides a method for the user to place their own arbitrary binary data alongside their spreadsheet content. It is not an application behavior but a user behavior. It would not be possible to speculate on the various reasons an user would decide to store additional data outside of what is part of the Spreadsheet, which is why the use cases are not detailed in the spec.
Part 1, Section 15.2.12	-	te	There is no reference made to a particular dated version of TrueType or OpenType specifications. And if TrueType and OpenType differ, then there should be different ways to refer to them, rather than calling them both “application/x-font-ttf”	Provide reference to intended specifications for TrueType and OpenType	Open Type font is an extension of True type font. Since Office Open XML is providing backward compatibility, it supports all previous versions of fonts. So explicit reference to any particular version is not necessary. If the application does not understand the font type, than it can fall back to a different font using the panose information.

Part 1, Section 15.2.14	-	te	It is unsatisfactory to store printer settings in OS-dependent binary formats like DEVMODE structures. This is a considerable security concern (DEVMODE structures are loaded directly into device driver memory), as well as lacking cross-platform adaptability. There is also no interoperability with print settings as currently defined.	Alternatives are available for expressing print settings in XML rather than in binary. For example, Microsoft's own XPS specification defines a PrintTicket markup for which the XPS specifications says, "The PrintTicket is XML that provides print settings in a consistent, accessible, and extensible manner. We would like the same qualities in OOXML's print settings, not a binary blob.	The majority of print settings are indeed defined in XML, and anything else is outside of the current scope of the specification. For the settings which are specific to a type of printer where those settings have not been defined in the spec, this part is available for the printer's own data to be stored. It is up to the printer manufacturer to use an interoperable format for those settings. In the future, it may be determined that more common printer settings should be defined, but that would be for a future version.
Part 1, Section 15.2.15	-	te	For there to be interoperability of this feature, it must either specify what size the thumbnail should be or state that the application will scale the image as needed.	Clarify what size the thumbnails should be, or that the images are scaled.	There is no reason a thumbnail's size should be specified in the standard. It should completely be up to the producing application to decide what size thumbnail they create, and the consuming application can decide how they want to scale the thumbnail.
Part 1, Section 15.2.6	-	te	What is meant by "This part shall have no contents"? Does this mean that there shall be nothing in the Zip file with the declared name? Or does it mean that a zero-byte file shall be created with the declared name? Or something else?	Clarify the meaning.	The phrase "This part shall have no contents" means that a part shall be created, but it shall have no contents (meaning it would be a zero byte file).

Part 1, Section 2.1 "Goal"	-	ge	There are no normative statements in this clause, though Section 2 is indicated to be normative	Mark clause as informative using one of the mechanisms of Section 7	Section 2.1: Goal, describes the goal of this Section 2 and does not prescribe implementable behavior, and thus is not properly described as "normative" - but there should be no confusion in any implementer's reading as to whether this Section 2 is "normative" with respect to an implementation of the specification. There is no reason to make such a change.
Part 1, Section 2.2 "Issues"	-	ge	There are no normative, statements in this clause though Section 2 is indicated to be normative	Mark clause as informative using one of the mechanisms of Section 7	Please see the response to 'Part 1, Section 2.1 "Goal" above.
Part 1, Section 2.4	line 22	te	This line require conformance with "Unicode Standard" without specifying a version. XML 1.0 referred to Unicode 2.0, though the informative Appendix A of OOXML Part 1 lists Unicode 4.0. Which is it?	An explicit Unicode version reference should be made in the Conformance section.	Unicode consortium cites that "Since Unicode is an open standard, it is important not to over-specify the version number" (http://unicode.org/versions/), so it is not mandatory to make an explicit reference to Unicode version in the conformance section. However, line 23, Section 2.4 "Document Conformance", Part 1 indicates that "The document character set shall conform to the Unicode standard and ISO/IEC 10646-1 with either UTF – 8 or UTF – 16 encoding form, as required by XML 1.0 standard". Specification document does not explicitly specify Unicode 2.0. The Unicode version 4.0 is mentioned in the "Annex A", not "Appendix A". The Annex A is a bibliography, and is just informative.

Part 1, Section 2.6	-	ed	The use of the word “element” is ambiguous. Is this to mean XML elements (but not attributes, character content, etc.)? Or does this mean an element of the Standard, in the usage of ISO Directives, Part 2?	Clarify the use of the word “element” perhaps by saying “XML element” if that is what is meant.	As the section 2.3 is referring to syntax and semantics of XML schemas, “element” here implicitly refer to the XML element.
Part 1, Section 4 “Definitions”	behavior, implementation-defined	te	“application-specific”, at least in common standards use, is not the same as application-defined, viz. ANSI C Programming Language	Use “application-defined” consistently where the intent is for applications to document their behavior.	The definition “behavior, implementation – defined” is defining the behavior of the application that has implemented the standards. Here the application is not defining the behavior as it is just following the standards. So this behavior will be application – specific rather than application – defined.
Part 1, Section 4 “Definitions”	Office Open XML Document	ed	This definition doesn't hold together. Are these two different definitions? Or two clause of which either will define the term? Or both together define the term?	Clarify the definition	Both these sentences define Open Packaging Concept. Part 1 gives overview of Word, Spreadsheet, Presentation ML and other MLs and part 4 talks about them in detail, which means that they are nothing but packages. The second sentence explicitly calls the three documents as Package

Part 1, Section 9.1.1	-	te	ASCII requires a normative reference since there are several national variations.	Suggest reference be made to ISO/IEC 646:1983 or ANSI X3.4-1986	Part 1, Section 9.1.1, gives a list of characters that are not permitted in the part names. It also states that "All other ASCII characters are permitted only when escaped as an encoded triplet of the form "%HH", where H is a hexadecimal digit." As the entire list of the characters that can and cannot be used are specified explicitly in the section, another explicit reference such as ISO/IEC 646:1983 or ANSI X3.4-1986 is not required.
Part 1, Section 9.1.5	-	te	This sub-clause, buried in introductory material, negates a provision of the more detailed OPC specification in Part 2. This will likely be missed by implementors.	If interleaving is not permitted then it should not be described in Part 2.	The note in section 9.1.5 states that "In order to simplify initial implementations of the Standard, interleaving is not used in this current version of the Office Open XML formats but it may be used in further versions of the standard or by other formats that leverage OPC". The Interleaving in Part 2 is for future purpose and hence need not be removed from specifications.

Part 2, Section 3. Definitions	page 4, line 20	te	This definition of 'package model' is not compatible with the prior definition given in Part 2, Section 1. Scope, page 1, line 5.	Define 'package model' in unambiguous terms and use the resulting definition consistently throughout the OOXML text.	<p>This comment appears to be based on the misunderstanding of the specification.</p> <p>In Part 2, section 1: "Scope" ,page 1, line 5, the definition of "Package Model" is defined as :</p> <p>"The package model defines a package abstraction that holds a collection of parts. The parts are composed, processed, and persisted according to a set of rules. Parts can have relationships to other parts or external resources, and the package as a whole can have relationships to parts it contains or external resources. The package model specifies how the parts of a package are named and related. Parts have content types and are uniquely identified using the well-defined naming guidelines provided in this Open Packaging specification."</p> <p>The above definition is more elaborate than in the Part 1, section 3: "Definitions" page 4, line 20 which is stated as follows:</p> <p>"package model — A package abstraction that holds a collection of parts. "</p> <p>As one can observe, there is no conflict in these two definitions and so no scope for ambiguity.</p>
--------------------------------	-----------------	----	---	--	---

Part 4, Foreword	page vi, line 2	ed	DIS 29500 is a multi-part document, not a multi-part Standard, i.e., the individual parts of this Standard are not themselves standards.	Correct the terminology to correctly reflect the status of DIS 29500.	Issues of an editorial nature will be addressed after the Ballot Resolution Meeting by the specification Editor.
Part 4, Foreword	page vi, line 9	Ge	Explicitly references annexes that are not provided in a humanly-readable format, whereas a human-readable format is required by JTC1 Directives 8.3.5 and Annex H	Annexes should be provided in a humanly readable, lined-numbered format so it can be referenced and cited. The reference to electronic form only annexes should be removed.	The annexes are text files that contain XML source or XML schemas, and are intended for consumption by tools and not by human readers. Specification document is already very exhaustive and adding summaries for each schema separately will make this document unreviewable in the specified period. However, where ever there was a need, schema's are explained within the specification document as in the case of SpreadsheetMLStyles in sections 3.8.7 and 3.8.40 of Part 4 and DrawingMLGeometries in sections 5.1.11.18 and 5.1.11.19 of Part 4. Additionally schemas are explained in detail throughout part 4 and 5 of the specification document. ===== Issues of an editorial nature will be addressed after the Ballot Resolution Meeting by the specification Editor.
Part 4, Introduction	page vii, line 7	ed	Full compatibility of the proposed OOXML with any existing application is demonstrably unreachable (because the proposed OOXML explicitly gives up describing parts of what it aims to describe, e.g. Part 4 page 1378 lines 12-17).	Rephrase the compatibility goal so as to make it realistic.	This can be reviewed and addressed by the specification editors at the BRM Meeting.

Part 4, Introduction	page vii, line 8	te	An XML markup cannot be “fully compatible” with an “investment”	Clarification sought	The goal of Open XML is indeed to have compatibility with the existing base of Office documents. In the cases as those referenced, we are using state of the art capabilities to best enable interoperability between implementing applications, the syntax for identifying these properties are fully specified in the standard. For more information on the specific properties mentioned, you can reference this technical description: http://blogs.msdn.com/brian_jones/archive/2007/01/09/specifying-the-document-settings.aspx (details already given with response to comments earlier)
----------------------	---------------------	----	--	----------------------	---

Part 4, Section 3.17.4.1	-	te	The restriction to only two date bases is arbitrary and based only on one vendor's applications. There are other reasonable values for date bases, including earlier ones for historical dates.	Allow a range of vendor-declared date bases, or explicitly allow negative date serial values to express dates prior to 1900	The OpenXML Date formats are formats designed to best take the legacy of the past as a building block toward more powerful solutions in the future. The OpenXML formats allow for two approaches with dates. The first approach starts the baseline in 1904, and in no way contradicts with whether or not 1900 was a leap year. This means that implementers may use this approach and therefore introduces no contradiction with the Gregorian Calendar in ISO 8601:2004. The second approach uses a legacy behavior where 1900 is the baseline, and in order to maintain compatibility with the existing base of documents. As these two date base systems provide the conformance to the standard and legacy compatibility, there is no need to define a range of vendor-declared date bases. Most users may not use dates before January 1, 1900 and the ISO 8601 standard itself only uses Gregorian time which is only used for valid dates after 1582 as before that time historians use Julian time, so use of spreadsheet date fields is limited for genealogist/historians whatever standard they use. Therefore no need to express date prior to 1900.
--------------------------	---	----	---	---	--

Part 4, Section 3.17.4.1	-	te	The mandated incorrect date calculations for 1900 in the 1900-based date basis is unacceptable. An ISO standard should not be mandating incorrect values for the well-established Gregorian Calendar. To do so will only lead to confusion, poor interoperability and perpetuation of errors.	If needed for legacy reasons with legacy Excel documents, then introduce an additional vendor-specific tag such as "doWrongDateCalculationsLikeExcel" or similar. This is the approach recommended elsewhere in OOXML for legacy Word features.	The OpenXML Date formats are formats designed to best take the legacy of the past as a building block toward more powerful solutions in the future. The OpenXML formats allow for two approaches with dates. The first approach starts the baseline in 1904, and in no way contradicts with whether or not 1900 was a leap year. This means that implementers may use this approach and therefore introduces no contradiction with the Gregorian Calendar in ISO 8601:2004. The second approach uses a legacy behavior where 1900 is the baseline, and in order to maintain compatibility with the existing base of documents. As these two date base systems provide the conformance to the standard and legacy compatibility, there is no need to define a range of vendor-declared date bases. Most users may not use dates before January 1, 1900 and the ISO 8601 standard itself only uses Gregorian time which is only used for valid dates after 1582 as before that time historians use Julian time, so use of spreadsheet date fields is limited for genealogist/historians whatever standard they use. Therefore no need to express date prior to 1900.
Part 4, Section 3.17.4.1	page 2522, lines 14-18	te	The text proposes a dual date base system. There is no clear advantage to having two slightly different systems, and this brings significant costs and confusion, as illustrated by the need to specify a default base system, etc.	Choose and keep a single date system.	Maintaining the separate date systems is important to preserve compatibility with legacy documents.

Part 4, Section 3.17.4.1	page 2522, lines 16&18	te	The documented upper limits for serial date times match 9999-12-31 00:00:00, which is most probably not what was intended. The expected upper limits would match 9999-12-31 23:59:59.	Clarify the upper limits.	In the 1900 date base system the lower limit of 1 maps to 1900-1-1 00:00:00, and increases by a value of 1 every 24 hours. The specification is correct in that the upper limit of 2,958,465 does indeed map to 9999-12-31 00:00:00.
Part 4, Section 3.17.4.1	page 2522, lines 19	te	The proposed date system does not cope with dates prior to 1900-01-01.	Propose a better date system.	The specification is both forward looking while also taking on as a goal compatibility with legacy documents. The date system proposed in section 3.17.4.1 has been in use for over 20 years, and there have been no significant complaints from customers. The ecma TC did not feel it was necessary to change from this approach, as it would be outside of the scope of the goals for version 1.0 of the spec.
Part 4, Section 3.17.7.341	-	te	As written this function mandates an incorrect calculation for day of week for certain dates in the year 1900. An ISO standard should not be mandating incorrect values for the well-established Gregorian Calendar. To do so will only lead to confusion, poor interoperability and perpetuation of errors.	Remove the text that defines behavior that results in incorrect date calculations.	The point raised about spreadsheet formulas like the WEEKDAY() function is a great example of why this behavior was designed into the spec. Today, there exist billions of formulas in use, and all those formulas calculate based on this behavior. If the behavior were to change, the affect on existing spreadsheets and their formulas would be very unpredictable. As this off-by-one error applies only to the first 60 days of the year 1900 and no other dates in history or the future, it would be best to not risk breaking existing documents and there is no negative impact on future documents either.

Part 4, Section 3.2.29	-		A hash algorithm is provided, likely based on a legacy algorithm used in Excel. This legacy algorithm is known to be a weak algorithm and has effectively been cracked. One could argue that no hash algorithm would be effective in OOXML, since a user could simply unzip the document and hand edit the XML to remove the hash or to set it to some known value. However, some application types such as online editing via Google Docs, or other similar applications, can secure physical access to the document via other means. Editing access to the document does not necessarily presuppose physical access to the document's XML. So there is a necessity for a secure & interoperable hash algorithm, such as SHA-256 for document protection.	Use a standard, FIPS-180 compliant hash algorithm as the default. Legacy hash algorithms should be supported via the described extension mechanism.	This is a good point, and in fact many algorithms are supported. As called out in the cryptAlgorithmSid attribute as defined in section 2.15.1.28 of Part 4, you'll see that while the hashing algorithm used is completely extensible, the following have been explicitly called out: MD2 MD4 MD5 SHA-1 MAC RIPEMD RIPEMD-160 HMAC SHA-256 SHA-384 SHA-512
Part 4, Section 3.2.29	p. 1917-1922	te	No normative description of the password hashing algorithm is provided, so interoperability of this feature cannot be assumed. In an informative section, 5-pages of C-language source code is provided as "an example", and this appears to involve machine-dependent bit manipulations.	Provide a normative, cross-platform definition of the hashing algorithm. Cross-platform source code can be given as an example, but the normative text should be in English, not in a programming language.	The normative description of the hashing algorithm used by OOXML, along with the flowchart is defined in section 2.15.1.28, Page 1158 to Page 1164. The C-Language Source Code provided in the section 3.3.1.69 is an example for easier understanding of the algorithm.
Part 4, Section 3.2.29	pg. 1916	te	This seems to imply that if a password is entered in a script like Armenian or Ethiopic then the characters will be replaced all by a single character 0x3F, making the protection feature useless. This is unacceptable.	Remedy so password hashes can be calculated on any Unicode password.	This issue can be corrected during the ballot resolution phase

Part 4, Section 3.2.29	pg. 1916	te	The conversion from input password to single byte string is ambiguous. Certainly the input password could contain characters from more than one script, say some Korean, some Chinese. Do we process via multiple DBCS code pages? Or just one and then replace the unmapped characters with 0x3F? If only one DBCS code page is used, how is that determined in this case?	Clarify this processing, especially for passwords that use characters from more than one script.	This issue can be corrected during the ballot resolution phase
Part 4, Section 3.3.1.61	-	te	The pageSize attribute allows a set of enumerated values which does not encompass all of the page size values permitted by ISO 216, ANSI Y14.1 and similar DIN and JIS standards.	Rather than trying to maintain a paper size registry, a more flexible approach would be to simply record the dimensions of the paper size selected.	The paper size enumerations in OpenXML are not in contradiction with ISO 216 as the enumerations include values for common use ISO 216 paper sizes as well as paper, envelope, fanfold, and specialty dimension media in common use and applicable within the specifications' scope and context. The paper size enumerations are documented within, and maintained as part of OpenXML. The ISO 216 page sizes are represented in OOXML, as well as others. OOXML simply uses a number to represent each page size and what number represents which size is documented in Sections 3.3.1.61.
Part 4, Section 3.3.1.69	-	te	No normative description of the password hashing algorithm is provided, so interoperability of this feature cannot be assumed. In an informative section, C-language source code is provided as "an example", and this appears to involve machine-dependent bit manipulations.	Provide a normative, cross-platform definition of the hashing algorithm. Cross-platform source code can be given as an example, but the normative text should be in English, not in a programming language.	The normative description of the hashing algorithm used by OOXML, along with the flowchart is defined in section 2.15.1.28, Page 1158 to Page 1164. The C-Language Source Code provided in the section 3.3.1.69 is an example for easier understanding of the algorithm.

Part 4, Section 3.3.1.69	-	te	he securityDescriptor attribute, “defines user accounts who may edit this range without providing a password to access the range”. It is a string. But no information is given as to what user accounts are referred to here, or what the delimiter is. Are these comma-delimited local machine user accounts? Or semi-colon delimited LDAP DN's? There will be no interoperability if this is not defined.	Fully define this attribute.	Open XML will not define a universal standard for digital identity, but will provide a place to identify the name of the user in question, for which applications can choose to integrate their own digital identity.
Part 4, Section 5.1.3.2	-	te	No mention is made of what audio formats or codecs are permitted.	An interoperable set of formats should be specified.	This comment appears to be based on the misunderstanding of the specification. Some of the audio formats or codecs that it supports are defined in Part 4, section 15.2.2
Part 4, Section 6.1	page 4343, line 8	te	The reference to 'millions of documents' is an unsupported assertion. Furthermore, it is irrelevant in the context of a standard proposal.	Remove the marketing fluff from the text.	“millions of documents” refer to all the legacy office documents used prior to the OfficeOpenXML. This is relevant in the context specified. Issues of an editorial nature will be addressed after the Ballot Resolution Meeting by the specification Editor.
Part 4, Section 6.1	page 4343, line 9	ed	The reference to the specific commercial product 'Office 2000' brings no value to the proposal.	Remove the reference to Office 2000.	This content is informative and is provided to give background into the history of VML. It is believed that this is actually useful information.

Part 4, Section 6.4.3.1	-	te	<p>The allowed values of this enumeration, EMF, WMF, etc., are Windows-specific formats. No allowance seems to have been made for use by other operating systems. For example, in Linux images are typically copied on the clipboard in an open standard format like PNG.</p>	<p>Several options here, but the desire is to allow cross platform interoperability.</p>	<p>This specifies a suggested application behavior, but nothing that is required for conformance. There is no impact whatsoever on the document itself, and this setting only applies to a suggestion on what format to put on the clipboard if a user clicks on the object at edit time and chooses to copy it. There is a list of a number of image types, including bitmap or printer specified. It's also possible though to use any other image format the consuming application cares to use. The behavior is already completely extensible.</p>
-------------------------	---	----	---	--	--

Part 4, Section 7.1	-	te	<p>This is the specification of Office Open Math Markup Language, a specialized XML vocabulary for the describing the layout of mathematical equations. This solves the same problem as MathML, a long-established W3C standard and an ongoing activity in the W3C. Since the equation editing feature of Word was entirely rewritten in Word 2007, there doesn't not seem to be the argument that an additional equation language must be introduced for the sake of legacy documents.</p>	<p>It is recommended that this section be removed from OOXML and that the proposers of OOXML work within the W3C's MathML activity, where MathML 3.0 is currently being drafted, to produce a single standard for equations that can be used later referenced by a future version of OOXML.</p>	<p>Word needs to allow users to embed arbitrary scan-level material (basically anything you can put into a Word paragraph) in math zones and MathML is geared toward allowing only math in math zones. A subsidiary consideration is the desire to have an XML that corresponds closely to the internal format, aiding performance and offering readily achievable robustness. Since both MathML and OMML are XMLs, XSLTs have been created to convert one into the other. To support MathML as an interoperability language between apps, Office read and write Presentation MathML on the clipboard (leveraging those XSLTs). MathML, could have been used as it was already fully designed and documented. It would have been much easier, but it would have also meant to either cut back the functionality, or extend it in such ways that it was no longer as usable. Instead, use MathML as a guide, and try to leverage as much of the design as could. MathML works great for isolated math islands, it didn't give everything needed at the document-level. Although MathML does have space for annotations so it could have been extended, that would not have worked well with document-level features like comments, track changes, Word styles, etc. So we have an XML format that supports all of the features, and that format is fully documented and free for anyone to use.</p>
---------------------	---	----	---	---	--

Part 4, Section 7.4.2.5	-	te	This element defines values for use on Windows and Macintosh platforms, but not for Linux or any other operating system.	Several options here, but the desire is to allow cross platform interoperability.	Section 7.4.2.5, specifies the format of the clipboard data and must be: -3, -2, -1, 0, or any positive integer. The enumeration value "any positive integer" can be used for registering a particular format. The format is registered in ST_CF simple types. Any positive value in the above sentence means : A null-terminated string that contains a Windows clipboard format name, one suitable for passing to the RegisterClipboardFormat function. This function registers a new clipboard format. If a registered format with the specified name already exists, a new format is not registered and the return value identifies the existing format. This enables more than one application to copy and paste data using the same registered clipboard format. The format name comparison is case insensitive and is identified by values in the range from 0xC000 through 0xFFFF. The code page used for characters in the string is according to the code-page indicator. The "positive value" here is the string length, including the null byte at the end. When register clipboard formats are placed on or retrieved from the clipboard, they must be in the form of an HGLOBAL data-type value, which provides the handle to the object. So it is interoperable with other operating systems.
-------------------------	---	----	--	---	--

Part 4, Section 7.4.2.5	-	te	Even within a single platform, there is not enough information given to achieve interoperability. For example, what are the allowed values and meanings for a “built-in Windows clipboard format value”?	Specify this so interoperability may be achieved.	Section 7.4.2.5, specifies the format of the clipboard data and must be: -3, -2, -1, 0, or any positive integer. The value -1 is for built-in Windows format. Therefore, it is interoperable.
Throughout	-	te	The name "Office Open XML" is often mistakenly called 'Open Office XML' implying a connection to the OpenOffice project which does not exist. This naming confusion has been documented and has occurred numerous times, including by analysts and even in Microsoft press releases and blogs. Since “Open Office” is the pre-existing name, by 6 years, Ecma should choose a new name, less apt to continue this confusion.	Change the name of Office Open XML to a name which is not confused with OpenOffice.	The formal name of DIS29500 is "Ecma Office Open XML File Format" and is descriptive of what the formats are meant to accomplish. ECMA owns and publishes the specification. Office is not a trademarked term, it is used by many business productivity suites and products, including primary Microsoft Office competitors. Open describes the process by which the specification was created and reviewed within Ecma International. So there is no scope for confusion. Issues of an editorial nature will be addressed after the Ballot Resolution Meeting by the specification Editor.