	A	B	C	D	E	F	G	H
1	**National Comittee**	**Line number**	**Clause/ Subclause**	**Paragraph / Figure / Table**	**Type of comment** (General/Technical/ Editorial)	**Comments**	**Proposed change**	**OBSERVATIONS OF THE SECRETARIAT** on each comment submitted
2	**ES**		Part 4, Section 3.17.4.1	Page 2522	**te**	OOXML only support two time-bases or date-bases (the 1900 base and the 1904 date). None of those two reperesent corrrectly the Gregorian Calendar (ISO 8601) accepted in all the world. Also, 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. Most SQL databases, which frequently exchange data with spreadsheets, support a much greater range of dates.	Allow a range of vendor-declared date bases, or explicitly allow negative date serial values to express dates prior to 1900	
3	**ES**		Part 4, Section 3.17.4.1	Page 2522	**te**	OOXML can not represent important dates and generates wrong data calculations. . 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 used elsewhere in OOXML for legacy Word features.	
4	**ES**		Part 4, 3.17.4.1		**te**	OOXML specifies that in the year 1900, “for dates between January 1 and February 28, WEEKDAY shall return a value for the day immediately prior to the correct day” and also assigns a “serial value” to the non-existing day February 29, 1900. This is wrong. Software bugs should be fixed, not exported by means of ISO standards to the programs of competitors.	Make the calendar system configurable, with a default of the Gregorian calendar, allowing alternatives to be specified by means of providing code for the computation of year, month, weekday and day-of-the-month. .	
5	**ES**		Part 3 , 3.16.9 Part 4, Section 3.17.4.1	Page 2522, lines 15 and 16:	**te**	In the 1900 date base system, the lower limit is January 1, 1900, which has serial value 1. There are people alive today born before 1900. Historical studies often consider dates before 1900.	Remove this restriction and allow a greater date support. Allow negative values in the serial number.	
6	**ES**		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.	
7	**ES**		Part 3, 3.16.9.1 – 3.16.9.3 Part 4, 3.17.4.1 3.17.6.7		**te**	Having two different date systems with different base dates side-by-side in the same standard document format makes no sense. Rather, it is appropriate to fix a single base date. Applications which use a different base date can convert from the date representation used in the standard document format to the application's preferred date representation, and vice versa.	Delete all references to the “1904 base date system”.	

	A	B	C	D	E	F	G	H
8	**ES**		Part 4, Section 2.8.2.16	-	**te**	Use of bit masks is not the right way in XML. This element of the specification uses a set of bitmasks to specify which code pages a given font supports. The use of bitmasks rather than an XML Schema derived type makes this data almost impossible to work with standard XML tools like XSLT which lack bit-level operations. One of the advantages of XML is that we don't need to encode data like this any more.	Rewrite this subclause to express the feature using XML constructs rather than bitmasks.	
9	**ES**		Part 4, Section 7.4.2.4	-	**te**	The presence of non-XML characters, escaped, or not escaped in an OOXML document, is contrary to interoperability of XML and XML-based tools. The W3C's Internationalization Activity confirms this interpretation, saying “Control codes should be replaced with appropriate markup. Since XML provides a standard way of encoding structured data, representing control codes other than as markup would undo the actual advantages of using XML. Use of control codes in HTML and XHTML is never appropriate, since these markup languages are for representing text, not data.”	Remove the bstr type from OOXML	
10	**ES**		Part 1, introduction (page xii) and entire document		**te**	It is not acceptable for an international standard to be designed primarily around the goal of compatibility with a particular company's products. This is particularly inappropriate where, as in the present case, compatibility with an existing international standard is neglected in favor of the one-sided goal of maximal compatibility with document file formats introduced by one company, and where the proposed standard does not provide equal opportunities for compatibility to that company's present and future competitors. Unless this shortcoming of OOXML/DIS 29500 is fixed, accepting this specification as a national or international standard would be a violation of Spanish and international law.	Change the goal from being „fully compatible“ with „existing investments in Microsoft Office documents“ to seeking at attain the same high level of compatibility not only with Microsoft's formats, but also with the international norm ISO/IEC 26300 (OpenDocument). Review the entire draft standard and modify it corresponding to this revised goal.	
11	**ES**		Part 1, Section 15.2.14		**te**	Security hole. OOXML allows the inclusion of arbitrary binary blobs of data in ways that could be abused my malicious document authors**. For example:** Part 1, Section 15.2.14 recommends that print settings be stored in the binary DEVMODE format used by Windows printer drivers. However, if someone were to change this DEVMODE binary data it would be loaded into the printer driver the next time a user tried to print. Since a printer driver operates at a higher level of privilege than a user, this may allow a hacker to take control of a user's machine by crafting a specific document.	The current procedure could be a good approach to keep interoperability with past legacy tools, but an ISO standard must provide a clear specification for future implementations which does not perpetuate a security hole.	
12	**ES**		Part 4, Section 7.1		**te**	OOXML is not compatible with the industry standard language for displaying mathematical equations — MathML — used by the research community and the most important publications as Science, Nature, etc... . No interoperability with MathML is in the OOXML specifications.	Add the pertinent interoperability with MathML	

	A	B	C	D	E	F	G	H
13	**ES**		Part 4, Section 6.1.2.19	pg. 4653 “equationxml”	**te**	This describes the "equationxml" attribute of "shape" elements, "used to rehydrate an equation using the Office Open XML Math syntax". However, the "actual format of the contents of this attribute are application-defined", which makes them impossible to exchange between applications. If we're going to have a new math markup language in XML, and ignore the existing MathML, let's at least use the new markup in its elemental form, as well-formed XML (not stuffed into an attribute value), and without extending it in application-dependent ways.	Define equations in an interoperable way.	
14	**ES**		All the document		**te**	The MS-OOXML spec contains patented elements in a way not conforming the "ISO/IEC Directives (part 1, section 2.14) . Using patents is compatible with ISO procedures, but must follow the ISO/IEC directives (Part1, section 2.14), and this does not happen: because some patented funtionalities are outside the OOXML specification, some optional funtionalities not covered by the OOXML specification have also patents, and, finally, the statment about patents in OOXML specification is so vague than it has no legal security for companies implementing eventually OOXML specification.	Follow ISO/IEC Directives (part1, section 2.14) for patents issue.	
15	**ES**		Page 1754	**line 1**	**te**	Language specification: OOXML is ignoring ISO 639 and 639-2, and instead is using Microsoft Locale ID page	Accept using ISO 639 , 639-2 specifications	
16	**ES**		Part 4, Section 2.18.51	-	**te**	The use of 255 enumerated language codes, in addition to ISO 639-1 codes, adds no expressive value and only increases the work required of any application that would process an OOXML document.	Drop the use of the redundant ST_LangCode	
17	**ES**		Part 4, 2.18.52		**te**	Having two different language representation systems, one based on ISO standards and one with arbitrary hexcodes for a subset of the languages makes no sense. Applications which use arbitrarily-chosen numeric values to represent some languages can convert from the standard language representation system to the codes they use internally, and vice versa.	Delete all references to ST_Langcode.	
18	**ES**		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.	
19	**ES**		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.	

	A	B	C	D	E	F	G	H
20	**ES**		Part 4, Section 3.2.29	-	**te**	A hash algorithm is provided to secure passwords, 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, such ISO/IEC 10118-3:2004, compliant hash algorithm as the default. Other country specific standards could be acceptable such as FIPS-180 from NIST, but additionaly to glbal standards . Legacy hash algorithms should be supported via the described extension mechanism.	
21	**ES**		Part 4, Section 2.15.1.28		**te**	The algorithm given there has not undergone any peer review. Why are they not using a recognized standard cryptographic hash algorithm like SHA-256 or Whirlpool? Since the algorithm specified has not undergone review, we do not know that it is secure. It can be argued that most Office application do not need strong hashing, but a specification for standard must have general purpose and take into account all the situations. It is not clear, how this feature can accept another hashing alternatives.	Using a recognized standard cryptographic hash algorithm like SHA-256, Whirlpool, or so.	
22	**ES**		Part 4, 3.17.4.2 3.17.6.7		**te**	In the internet age, it is inappropriate to represent times simply as a numeric value without timezone information.	Specify that when stored in OOXML files, dates and times are always expressed in terms of UTC. Add a mechanism for storing in the file information on what timezone should be used to represent the time in human-readable form.	
23	**ES**		Part 4, 3.17.4.3		**te**	The “combined date and time representation” is broken in Spain and all other locales which switch to daylight-saving time and back, on those days which are 23 hours or 25 hours long instead of the usual 24 hours (summer and winther timeframes)	This problem can be easily avoided by always using UTC for the “combined date and time".	
24	**ES**		Part 4, Section 2.15.2.32	-	**te**	This feature has been defined in a way which ignores the existence of current browsers other than Internet Explorer. What about Firefox? What about Safari? What about Opera? None of these can be set as target browsers. This section requires that “all settings which are not compatible with the target web browser shall be disabled.” But what if I want my application to produce standards-compliant output? So yes to PNG, no to VML, yes to MathML and SVG? I can't seem to specify this.	ECMA should rethink the entire optimizeForBrowser subclause. It looks very much like it is mapping directly to the arbitrary choices of a single vendor's application. This clause should be rewritten to express this feature in an application and platform neutral way.	
25	**ES**		Part 4, Section 2.15.1.28	-	**te**	This says that document protection “shall be enforced”. “Shall” indicates required behavior. But then a few sentences later it says that document protection “may be ignored”.	Clarify this contradiction.	

	A	B	C	D	E	F	G	H
26	**ES**		Part 4, Section 2.15.1.28	line 13	**te**	This algorithm description fails to specify the encoding of the input password. Presumably it is Unicode, but in what encoding? UTF-16BE? UTF-16LE? UTF-16 with a BOM (Byte Ordering Mark)? The described algorithms make use of byte-level manipulations which depend on the machine architecture (big endian versus little endian). So it is necessary that all byte ordering assumptions be made explicit.	Make the byte ordering assumptions explicit, both for the input password and the processing steps, so as to allow cross-platform interoperability. Keep in mind that the hash may be calculated on a different machine architecture than the password was entered with.	
27	**ES**		Part 4, Section 2.15.1.29	-	**te**	This element allows the classification of the document into one of three types: “letter”, “email” or “general”. Although the description says that this feature can be used by, “hosting applications to facilitate customized user interface and/or automatic formatting behaviors based on the 'type' of a given WordprocessingML document”, the taxonomy provided is so weak as to be practically useless.	Either provide a reasonable document type taxonomy, or loosen the declared type to an xsd:string to allow applications to provide their own classifications.	
28	**ES**		Part 4, Section 2.15.1.86	-	**te**	This element uses a bitmask to specify a style display filter. The use of bitmasks rather than a set of boolean types makes this data almost impossible to work with standard XML tools like XSLT which lack bit-level operations.	Rewrite this subclause to express the feature using XML constructs rather than bitmasks.	
29	**ES**		Part 4, Section 2.15.1.87	-	**te**	This element uses a bitmask to specify style display sorting parameters. The use of bitmasks rather than a set of boolean types makes this data almost impossible to work with standard XML tools like XSLT which lack bit-level operations.	Rewrite this subclause to express the feature using XML constructs rather than bitmasks.	
30	**ES**		Part 4, Section 2.15.3.26	-	**te**	This is the “footnoteLayoutLikeWW8” element, which is defined in terms of mimicking a legacy application's behavior. The standard contains insufficient detail on how to replicate this behavior.	Define the intended behavior.	
31	**ES**		Part 4, Section 2.15.3.31	-	**te**	This is the “lineWrapLikeWord6” element, which is defined in terms of mimicking a legacy application's behavior. The standard contains insufficient detail on how to replicate this behavior.	Define the intended behavior.	
32	**ES**		Part 4, Section 2.15.3.32	-	**te**	This is the “mwSmallCaps” element, which is defined in terms of mimicking a legacy application's behavior. The standard contains insufficient detail on how to replicate this behavior.	Define the intended behavior.	
33	**ES**		Part 4, Section 2.15.3.41	-	**te**	This is the “shapeLayoutLikeWW8” element, which is defined in terms of mimicking a legacy application's behavior. The standard contains insufficient detail on how to replicate this behavior.	Define the intended behavior.	
34	**ES**		Part 4, Section 2.15.3.51	-	**te**	This is the “suppressTopSpacingWP” element, which is defined in terms of mimicking a legacy application's behavior. The standard contains insufficient detail on how to replicate this behavior.	Define the intended behavior.	

	A	B	C	D	E	F	G	H
35	**ES**		Part 4, Section 2.15.3.54	-	**te**	This is the “ uiCompat97To2003” element, which is defined as: “Disable UI functionality that is not compatible with Word97-2003”. But what use is this if I am using OOXML in OpenOffice or WordPerfect Office? What if I want to disable UI functionality that is not compatible with OpenOffice 1.5? Or WordPerfect 8? Or any other application? Where is the ability for other implementations to specify their preferences?	Define this an application neutral way. If it is truly a Word-only feature, then remove it from OOXML and express as an application-defined extension.	
36	**ES**		Part 4, Section 2.15.3.54	-	**te**	This is the “truncateFontHeightsLikeWP6” element, which is defined in terms of mimicking a legacy application's behavior. The standard contains insufficient detail on how to replicate this behavior.	Define the intended behavior.	
37	**ES**		Part 4, Section 2.15.3.6	-	**te**	This is the “autoSpaceLikeWord95” element, which is defined in terms of mimicking a legacy application's behavior. The standard contains insufficient detail on how to replicate this behavior.	Define the intended behavior.	
38	**ES**		Part 4, Section 2.15.3.63	-	**te**	This is the “useWord2002TableStyleRules” element, which is defined in terms of mimicking a legacy application's behavior. The standard contains insufficient detail on how to replicate this behavior.	Define the intended behavior.	
39	**ES**		Part 4, Section 2.15.3.64	-	**te**	This is the “useWord97LineBreakRules” element, which is defined in terms of mimicking a legacy application's behavior. The standard contains insufficient detail on how to replicate this behavior.	Define the intended behavior.	
40	**ES**		Part 4, Section 2.15.3.65	-	**te**	This is the “ wpJustification” element, which is defined in terms of mimicking a legacy application's behavior. The standard contains insufficient detail on how to replicate this behavior.	Define the intended behavior.	
41	**ES**		Part 4, Section 2.15.3.66	-	**te**	This is the “wpSpaceWidth” element, which is defined in terms of mimicking a legacy application's behavior. The standard contains insufficient detail on how to replicate this behavior.	Define the intended behavior.	
42	**ES**		Part 4, Section 2.16.4.3	page 1501, line 0	**te**	The definition for BATHTEXT references 'the given Thai format', which makes no sense in the context of that definition. What “given Thai format”?	Clarify the definition of 'BATHTEXT'.	
43	**ES**		Part 4, Section 2.16.5.33	-	**te**	This field says that it merely retrieves the picture contained in the named document. Is nothing else to be done with the picture? For example, should it be displayed?	Define what is to be done with the picture once it is retrieved.	
44	**ES**		Part 4, Section 2.16.5.33	-	**te**	This does not define how a picture is named. Is it by a URI? By a local file system path? Either? The example given has a DOS file path, a construct which is not portable.	Define how pictures are named.	

	A	B	C	D	E	F	G	H
45	**ES**		Part 4, Section 2.16.5.34	-	**te**	This does not define how a document is named. Is it by a URI? By a local file system path? Either? The example given has a DOS file path, a construct which is not portable.	Define how documents are named.	
46	**ES**		Part 4, Section 2.16.5.34	-	**te**	This subclause defines an INCLUDETEXT field which “Inserts all or part of the text and graphics contained in the document named”. However, no mention is made of what formats are permissible for the retrieved text.	There should be specified at least a small set of interoperable formats.	
47	**ES**		Part 4, Section 2.16.5.34	-	**te**	The \t flag will apply a named XSLT transform to the input XML file and insert the resulting output. However, no proper reference is given to XSLT, so we do not know what version XSLT transform is permitted here.	Provide a proper external normative reference for the XSLT and Xpath versions which are allowed here.	
48	**ES**		Part 4, Section 2.16.5.5	page 1512, lines 11-12	**te**	According to the text, the AUTONUM field is deprecated. A new standard should not contain deprecated parts.	Remove all references to AUTONUM from the OOXML text.	
49	**ES**		Part 4, Section 2.16.5.77	-	**te**	The example that illustrates USERINITIALS section instead shoes USERNAME.	Correct the example..	
50	**ES**		Part 4, Section 2.18.106	-	**te**	Length is said to be “exactly 1 character”. This is inconsistent with the earlier language and the schema fragment given which defines it as being 1 octet long or two characters.	Clarify the definition. In particular note that xsd:hexBinary measure length in octets, not characters.	
51	**ES**		Part 4, Section 2.18.4	-	**te**	No mechanism for expanding the set of art borders is provided. Since the specified art borders are heavily Western-oriented, it would be good to provide a way for an application to supplement these styles with graphics that provide more regional flavor.	Provide an interoperable extensibility mechanism for a document author or application to specify their own art border graphics.	
52	**ES**		Part 4, Section 2.18.45	-	**te**	Length is said to be “exactly 3 characters”. This is inconsistent with the example given which has a length of 6 characters.	Clarify the definition. In particular note that xsd:hexBinary measure length in octets, not characters.	
53	**ES**		Part 4, Section 2.18.52	-	**te**	This type is defined as containing, “a two digit hexadecimal language code”. It is fruther stated that, “This simple type's contents must have a length of exactly 2 characters”. However, two hex digits can count up to 255 and the values enumerated in this clause go far beyond that.	Reconcile the description of the type with the enumerated values.	
54	**ES**		Part 4, Section 2.18.57	-	**te**	The description of this type says it contains four hexadecimal digits, four hexadecimal octets and exactly four characters. These definitions are not compatible. A hexadecimal octet is two hexadecimal digits.	Clarify the definition. In particular note that xsd:hexBinary measure length in octets, not characters.	

	A	B	C	D	E	F	G	H
55	**ES**		Part 4, Section 2.18.66	-	**te**	The formatting system described here is not comprehensive, lacking, for example, support for Armenian, Tamil, Greek alphabetic, Ethiopic and Khmer numerations, all in use today, as well as the various historical systems still used by scholars.	Use a more flexible, extensible, generative approach to numeration, such as that used by the W3C's XSLT standard in their xsl:number support	
56	**ES**		Part 4, Section 2.18.66	-	**te**	There is nothing in this section which is normatively defined except some enumeration values. No normative meanings to these values are given. An informative example is insufficient in all but the most trivial cases. For example, where is “Korean Legal Counting System” defined?	Give explicit definitions of these numbering styles or proper external normative references.	
57	**ES**		Part 4, Section 2.18.66	“chicago”	**te**	Format is defined in reference to the “Chicago Manual of Style”, but no edition or page reference is provided.	Either include the entire definition in the standard, or provide a proper external reference.	
58	**ES**		Part 4, Section 2.18.66	“decimalEnclosedFullstop”	**te**	The example given does not show enclosed characters and so contradicts the normative text.	Reconcile the text and the example.	
59	**ES**		Part 4, Section 2.18.66	“lowerLetter”, etc.	**te**	Several counting systems are defined to use letters of the alphabet, but nothing is mentioned about how counting continues once the letters of the alphabet are exhausted.	Clarify the text to explicitly cover this case.	
60	**ES**		Part 4, Section 2.18.66	“numberInDash”, etc.	**te**	Format requires use of “dash” to surround the number, but no indication of which Unicode dash is intended, en-dash, em-dash, hyphen-minus, figure-dash, quotation-dash, etc.	Specify the intended dash explicitly.	
61	**ES**		Part 4, Section 2.18.72	-	**te**	Length is said to be “exactly 10 characters”. This is inconsistent with the example given which has a length of 20 characters.	Clarify the definition. In particular note that xsd:hexBinary measure length in octets, not characters.	
62	**ES**		Part 4, Section 2.18.85	-	**te**	The fill patterns lack definitions. The illustrations given are insufficient. An application needs to know what in these illustrations are required behaviors and what are not. For example, is the exact dithering pattern used in the illustration required?	Provide full normative definitions for these graphical elements.	
63	**ES**		Part 4, Section 2.18.86	-	**te**	The description of this type says it contains two hexadecimal digits, two hexadecimal octets and exactly two characters. These definitions are not compatible. A hexadecimal octet is two hexadecimal digits.	Clarify the definition. In particular note that xsd:hexBinary measure length in octets, not characters.	
64	**ES**		Part 4, Section 2.2.1	page 28, line 1	**te**	The sentence 'or auto to allow a consumer to automatically determine the background color as appropriate.' does not define the appropriate behavior of the consumer, whereas the definition of the corresponding simple type, found in Part 4, page 1737, explicitly states that 'This value shall be used to specify an automatically determined color value, the meaning of which is interpreted based on the context of the parent XML element.'	Define the characteristics of the auto value for the color attribute of the background element properly.	
65	**ES**		Part 4, Section 2.2.1	page 29, line 0	**te**	There are several instances of the word 'border' that are meaningless in this context (the text is supposed to describe the 'background' element at that location and no “border” has been defined).	Clarify which border the text refers to (if any notion of border must be introduced here) or else rewrite the text so that it makes sense.	
66	**ES**		Part 4, Section 2.2.1 background (Document Background)	page 27, lines 8&21	**te**	Contradicting use of accent3 and accent5 – the text says one thing, but the example says another.	Fix the contradiction.	

	A	B	C	D	E	F	G	H
67	**ES**		Part 4, Section 2.3.1.8	-	**te**	This element uses a bitmask to specify various paragraph conditional formatting properties.. The use of bitmasks rather than a set of boolean types makes this data almost impossible to work with standard XML tools like XSLT which lack bit-level operations.	Rewrite this subclause to express the feature using XML constructs rather than bitmasks.	
68	**ES**		Part 4, Section 2.3.3.19	-	**te**	This says that "The layout properties of this embedded object are specified using the VML syntax". However, in Part 1, Section 8.2.6 says, "VML should be considered a deprecated format included in Office Open XML for legacy reasons only and new applications that need a file format for drawings are strongly encouraged to use preferentially DrawingML" Certainly a new document creating an OLE embedding should not be using VML. Otherwise, all OOXML consumers will need to support VML, even where legacy documents are not present.	Define layout properties of embedded objects using DrawingML rather than VML	
69	**ES**		Part 4, Section 2.4.51	-	**te**	This element uses a bitmask to specify various table style formatting properties.. The use of bitmasks rather than a set of boolean types makes this data almost impossible to work with standard XML tools like XSLT which lack bit-level operations.	Rewrite this subclause to express the feature using XML constructs rather than bitmasks.	
70	**ES**		Part 4, Section 2.4.52	-	**te**	This element uses a bitmask to specify various table style formatting exceptions. The use of bitmasks rather than a set of boolean types makes this data almost impossible to work with standard XML tools like XSLT which lack bit-level operations.	Rewrite this subclause to express the feature using XML constructs rather than bitmasks.	
71	**ES**		Part 4, Section 2.4.7	-	**te**	This element uses a bitmask to specify various table cell formatting properties.. The use of bitmasks rather than a set of boolean types makes this data almost impossible to work with standard XML tools like XSLT which lack bit-level operations.	Rewrite this subclause to express the feature using XML constructs rather than bitmasks.	
72	**ES**		Part 4, Section 2.4.8	-	**te**	This element uses a bitmasks to specify various table row formatting properties.. The use of bitmasks rather than a set of boolean types makes this data almost impossible to work with standard XML tools like XSLT which lack bit-level operations.	Rewrite this subclause to express the feature using XML constructs rather than bitmasks.	
73	**ES**		Part 4, Section 3.18.86	-	**te**	Length is said to be "exactly 4 characters". This is inconsistent with the schema fragment given which defines it as being 4 octets long or 8 characters.	Clarify the definition. In particular note that xsd:hexBinary measure length in octets, not characters.	
74	**ES**		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.	
75	**ES**		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.	

	A	B	C	D	E	F	G	H
76	**ES**		Part 4, Section 3.2.29	pg. 1916	**te**	This algorithm description fails to specify the encoding of the input password. Presumably it is Unicode, but in what encoding? UTF-16BE? UTF-16LE? UTF-16 with a BOM (Byte Ordering Mark)? The described algorithms make use of byte-level manipulations which depend on the machine architecture (big endian versus little endian). So it is necessary that all byte ordering assumptions be made explicit.	Make the byte ordering assumptions explicit, both for the input password and the processing steps, so as to allow cross-platform interoperability. Keep in mind that the hash may be calculated on a different machine architecture than the password was entered with.	
77	**ES**		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.	
78	**ES**		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.	
79	**ES**		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.	
80	**ES**		Part 4, Section 3.3.1.69	-	**te**	The 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.	
81	**ES**		Part 4, Section 5.1.12.28	-	**te**	Length is said to be “exactly 3 characters”. This is inconsistent with the schema fragment given which defines it as being 3 octets long or 6 characters.	Clarify the definition. In particular note that xsd:hexBinary measure length in octets, not characters.	
82	**ES**		Part 4, Section 5.1.12.37	-	**te**	The Panose value is said to be used, “so that generating applications using this Office Open XML Standard may determine the closest font type if necessary”. However, no font distance metric or font matching heuristic is described.	Describe the intended font matching procedure.	
83	**ES**		Part 4, Section 5.1.12.37	-	**ge**	Why are there several different definitions for a Panose value, both in Word Processing ML as well as Drawing ML?	Since they are exactly the same they should be defined once in a shared schema.	

	A	B	C	D	E	F	G	H
84	**ES**		Part 4, Section 5.1.12.37	-	**te**	Length is said to be “exactly 10 characters”. This is inconsistent with the schema fragment given which defines it as being 10 octets long.	Clarify the definition. In particular note that xsd:hexBinary measure length in octets, not characters.	
85	**ES**		Part 4, Section 5.1.3.4	-	**te**	This describes the attachment of a QuickTime video to a presentation object. No description of the QuickTime format is provided. Without specifying a version and supported codecs, there will be no interoperability.	Provide an external reference for the version(s) of QuickTime format intended here as well as an interoperable codec.	
86	**ES**		Part 4, Section 6	-	**te**	OOXML specifies here a markup language called Vector Markup Language (VML) which, in addition to DrawingML, specifies a vocabulary for describing graphical objects. Section 6.1 says, “The DrawingML format is a newer and richer format created with the goal of eventually replacing any uses of VML in the Office Open XML formats. VML should be considered a deprecated format included in Office Open XML for legacy reasons only and new applications that need a file format for drawings are strongly encouraged to use preferentially DrawingML.” The need to support VML by OOXML consumers, in addition to DrawingML, would come at great implementation expense (the VML specification is over 600 pages) , would disadvantage all vendors but Microsoft, and would hurt interoperability.	Remove VML from OOXML. Vendors who have access to the legacy binary format documentation, such as Microsoft, are free to convert the VML to the “newer and richer” DrawingML at the same time they convert the document to OOXML.	
87	**ES**		Part 4, Section 6.1.2.19	pg. 4655, “gfxdata”	**te**	Describes a "gfxdata" attribute for the "shape" elements, which "contains DrawingML content" that is "base-64 encoded". However, the "contents of this package are application-defined", so even though they "shall use the Parts defined by this Standard whenever possible" there is not sufficient information for an independent implementation to read this data or display the "DrawingML content" contained therein. If we're going to have a new graphics markup language in XML, and ignore the existing SVG, let's at least use the new markup in its elemental form, as well-formed XML (not stuffed into an attribute value), and without extending it in application-dependent ways.	Define this in an interoperable way.	
88	**ES**		Part 4, Section 6.1.2.7	page 4444, “tableproperties”	**te**	This element uses a bitmask to specify VML table properties. The use of bitmasks rather than a set of boolean types makes this data almost impossible to work with standard XML tools like XSLT which lack bit-level operations.	Rewrite this subclause to express the feature using XML constructs rather than bitmasks.	
89	**ES**		Part 4, Section 6.4.2.10	-	**te**	This element is defined as providing a, “general-use element for objects that use an image representation, such as OLE objects, embedded controls, cameras and signature lines.” However, the allowed values, EMF, WMF, etc., refer to formats for which no reference has been given.	Provide a proper external normative reference for the allowed formats containable within this element.	

	A	B	C	D	E	F	G	H
90	**ES**		Part 4, Section 6.4.3.1	-	**te**	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.	Several options here, but the desire is to allow cross platform interoperability.	
91	**ES**		Part 4, Section 7.1	-	**te**	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.	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.	
92	**ES**		Part 4, Section 7.4.2.4	-	**te**	This defines a new XML string type which allows the inclusion via an escape mechanism of Unicode characters which are otherwise impermissible in XML documents. However, any escape mechanism must also specify a mechanism for "escaping the escape". So, how does one represent the literal example given in 7.4.2.4 in a bstr?	Complete the definition of the escape mechanism.	
93	**ES**		Part 4, Section 2.3.2.8	page 178	**te**	It is desired to have improved interoperability between ODF and OOXML. However, OOXML's "vert" attribute only allows text to be rotated 270 degrees, whereas ODF's equivalent allows text rotation by 90 or 270 degrees.	Include ability to specify 90 degree text rotation in addition to existing 270 degree rotation.	
94	**ES**		Part 4, Section 2.3.2.1	page 160	**te**	It is desired to have improved interoperability between ODF and OOXML. However, OOXML text runs only support two font weights, bold or normal, where ODF supports the fuller range of font weights from XSL-FO	Include support for additional font weights, 100, 200, 300, 400, 500,600. 700, 800 and 900.	
95	**ES**		Part 4, Section 2.4.46	page 421	**te**	It is desired to have improved interoperability between ODF and OOXML. However, OOXML lacks the ability to specify a multi-row header that repeats across pages, where ODF does.	Include in this section the ability to specify that the first N rows of a table can be selected as a header.	
96	**ES**		Part 4, Section 7.4.2.5	-	**te**	It doesn't make sense for us to be specifying strings as null-terminated C-style strings and then to base-64 encode that. That is avoiding XML and will cause the markup to interoperate poorly with XML-based tools.	ECMA should rethink the entire Clipboard Data representation. It looks very much like it is mapping directly to the arbitrary internals of a single application. This clause should be rewritten to express this feature in an application and platform neutral way.	