Digital Preservation Testbed White Paper

XML and Digital Preservation

The Digital Preservation Testbed was founded by the National Archives and the Ministry of the Interior and Kingdom Relations. It was established in October 2000 to research different methods of digital preservation over the long term. The Digital Preservation Testbed is part of the ICTU, a government initiative which houses different research projects concerned with varying aspects of e-government.

ICTU
Nieuwe Duinweg 24-26
2587 AD Den Haag

Tel. 070 888 77 77
Fax: 070 888 78 88

E-mail testbed@ictu.nl
www.digitaleduurzaamheid.nl

Digital Preservation Testbed White Paper *XML and Digital Preservation.*

Den Haag, September 2002.

Contents

1 *Introduction*[1]

Digital Records are fragile. The debate as to the best means of preserving digital records over the long term has been underway for many years and will no doubt continue for years to come[2]. Various theoretical solutions have been proposed, and research is currently underway around the world to identify ways in which digital records can be authentically maintained whilst remaining accessible and usable over the long term. This paper focuses on the use of XML in digital preservation. We place XML in context with contemporary thinking and practice about digital preservation, identify the issues involved, and provide a summary of current knowledge and research into XML.

1.1 Digital Preservation Defined

Digital Preservation is concerned with ensuring that records which are created electronically using today's computer systems and applications, will remain available, usable, and authentic in ten to one hundred years time, when the applications and systems which were used to create and interpret the record will, more likely than not, no longer be available. Digital preservation consists of preserving more than just the record's bit stream. We must also be able to *interpret* the bit stream in order for the *record* to survive. Without interpretation, the bit stream is nothing more than a meaningless series of 0's and 1's. During preservation, questions of record context, content, structure, appearance, and behaviour must also be taken into account. Appearance and behaviour are aspects that are peculiar to digital records. These may therefore require the most attention to authentically preserve the record over the long term.

There is a wide range of digital formats available and, to make matters more complicated, different digital objects have different preservation requirements. These can depend on the reason the record is being preserved, how long it needs to be preserved, the context and history of the record, and its original format[3]. Digital Preservation does not mean the same thing for each digital object. Whilst it is often considered that digital preservation means preserving the object so that it is identical to its original format, this is not always required. It is not always necessary to preserve every aspect of a digital record, and thus research is underway to define the essential aspects of records and their authenticity requirements. In all cases, however, the record must be preserved so that it retains its integrity and is authentic and usable. This presents interesting challenges.

---

[1] The text of this introduction has been taken nearly unchanged from the earlier published report "Migration: Context and Current Status" / Digital PreservationTestbed, Den Haag, December 2001.

[2] The phrase long term can mean fifty years or more, as indicated by Bennett in *A Framework of Data Types and Formats, and Issues Affecting the Long Term Preservation Of Digital Material* (1997). This appears to us to be a reasonable amount of time for which to prepare a long term preservation strategy. Technological changes after fifty years may well exceed expectations and limit the validity of a well-designed strategy. However, we must bear in mind that Dutch National Archives Regulations, specifically article 11 (1995), speak of a time scale of at least one hundred years.

[3] Dutch regulations (article 8) stipulates that what needs to be preserved depends on the requirements of the working process to which the record belongs.

1.2 The Task of Achieving Digital Preservation

There is a difference between paper and digital records. Any paper record can be perceived through the five human senses; no digital record can be perceived without going through computer hardware and software. For this reason, the speed of technological obsolescence makes digital preservation an important issue for everyone.

Digital records are software dependant. They rely upon the software that was originally intended to interpret (or display) them. When that software becomes obsolete, perhaps within the space of a few years[4], the problem arises of how to read that record without its original software application. It is unlikely that different versions of the application will read the file in the same way, and this may well result in a change in the interpreted record (the visible or available view of the file) that affects its archival integrity. Some data may be lost altogether; in other areas, data may be gained. There may be no way to compare a new version with the original, so changes may go unnoticed. Any changes to the record may affect its authenticity and integrity, which in turn may affect its archival and legal status. Depending on the nature of the record and its use, this can cause problems, not least that of losing or misrepresenting history.

Even a simple office computer system uses several different software applications. For each application, there may be several software manufacturers offering their own products. The rate at which new versions of software are released, with extended and (not necessarily backward compatible) new features, adds to the problem. Take Microsoft's Word® application as an example. The past six years have seen 4 versions released: Word 95; Word 97; Word 2000; and Word 2002. Two other versions of Word have also been produced for non-Windows Operating Systems – Word 98 Special Edition for Apple and IMac, and Word 2001 for the Mac. There are also different releases of Word within these versions. These releases have fewer differences among them, but each has the potential to affect a record's integrity or authenticity.

There are already many examples of how quickly digital records and data can become inaccessible. The specific details concerning who sent the first e-mail communication in the 1960's are no longer available. Some records from the old East German Republic have been lost forever, through technological obsolescence. A recent communiqué on the Joint Information Systems Committee (JISC) listserv revealed articles describing NASA's loss of data from the Viking probes sent to Mars in the mid 1970's[5].

There are several strategies for digital preservation. The following section provides a brief analysis of seven preservation approaches.

1.3 Different Approaches

The main preservation strategies are: *technology preservation; printing to paper; emulation; encapsulation; virtual machine software; XML; storage in standard*

[4] Gail Hodge, in *Best Practices for Digital Archiving* (1999), states that "new releases of databases, spreadsheets, and word processors can be expected at least every two to three years, with patches and minor updates released more often". The Public Record Office at Kew state that it would be unusual if migration occurred more frequently than every three years, in their *Guidelines on the Management, Appraisal and Preservation of Electronic Record* (1999).

[5] JISC listserv, Friday 3rd August 2001, A *nice case study for digital preservation.*

formats; and *migration*. These strategies have different technical requirements and costs. They also have different preservation metadata requirements.

1. *Technology Preservation*. One of the first options to be used was to preserve the technology required to access original records for as long as those records are required. However, this is costly and technologically complex (although in practice, some large corporations continue to employ this approach). Support for the software and hardware eventually ceases and the parts required to maintain the hardware become more and more scarce as manufacturers discontinue obsolete components. The number of machines available that are capable of reading old files continues to decrease, for computers do not last forever. The skills required to operate the hardware and software also become rare and eventually disappear.

2. *Printing to Paper*. This is another of the early approaches which is also still in practice. However, printing all records to paper is not a viable preservation method for the majority of records. Printing to paper loses functional or behavioural traits that the records had in their digital form. Certain information may also be lost. Embedded formulas in a spreadsheet, for example, will not print to paper. Databases were simply not designed to be printed out, and any printed version is only a selective view of the database, and not a preserved format.

   Legal rulings have worked both for and against printing to paper[6]. As the NLA notes, 'flat data', such as text and some still images, can be printed to paper without loss of data but with some possible loss of functionality[7]. Printing to paper is often employed as an interim approach to preservation whilst a digital solution is sought.

3. *Emulation.* The theory behind Emulation is that the only way to ensure the authenticity and integrity of the record over the long term is to continue to provide access to it in its original environment, i.e., its original operating system and software application. This can be done by preserving not only the record, but also an emulator specification, which contains enough details about the original environment for that environment to be recreated on a future computer when necessary.

   Some people believe that emulation is too complicated with too great a potential for error. There is no guarantee that we will be able to recreate the full computing environment of the record on future computers, as we do not know what the future computers will be like. However, Emulation has been explored in other fields with some success, and it may be the only way to maintain complex databases or multi media objects.

4. *Encapsulation.* In contrast to the migration approach, the encapsulation approach retains the record in its original form, but encapsulates it with a set of instructions on how the original should be interpreted. This would need to be a detailed formal description of the file format and what the information means. This encapsulating layer could be expressed using XML, for example.

[6] The saga concerning NARA's GRS20 went on for many years, with the Judge initially ruling against GRS20, stating that an email document is not the same as a paper document, and 'that hard copy printouts of an email may omit important parts of the electronic version'. However, this ruling was later overturned by the Court of Appeal in favour of the Archivist.
[7] National Library of Australia *Draft Research Agenda* 1998, p2.

If the original software used to interpret the data file is complex, then the description must also be complex and care would need to be taken to ensure that it was sufficiently complete. An extension to this idea is to create this description with an executable program: that is the subject of the section "Virtual Machine Approach".

5. *Virtual Machine Software.* A variant of the emulation approach has been proposed by Raymond Lorie of IBM[8]. This addresses the problem of interpreting data files in the future by writing a program to carry out this interpretation in the machine language of a "Universal Virtual Computer" (UVC). This program would be written at the time the record was archived and would be preserved together with the record. This program runs on what Lorie calls a UVC Interpreter, i.e. a virtual machine. In order to interpret the record on a future computer, a UVC Interpreter would be required and this could be produced from the specifications of the UVC. This approach is similar in principle to that used by the Java ™ platform to achieve present day interoperability of Java programs. To make this efficient and achievable, the key features of the proposed UVC language are that it should be simple enough that it is relatively straightforward to produce the future virtual machines, and it should be general enough that it can be widely used for archiving purposes, so that it is cost-effective to produce the future virtual machines. With this approach, the data can be stored in any format and the knowledge required to decode it is encapsulated in the UVC program.

   The approach can be extended to apply to the archiving of a program: this is more like the full emulation approach. It allows the emulator to be written in the UVC language at the time of archiving, without requiring any knowledge of the future target machine.

6. *Migration* (including *Storage in standard formats*). As has become apparent in recent reports, this is the best known and most widely applied preservation strategy[9]. It is also the most criticised method. Within the scope of the Testbed, migration is defined as the transfer of files from one hardware configuration or software application to another configuration or application. A simple example of this is the migration of a file from Microsoft Word 6 to Word 7. A more complex example is the migration of a file from an Apple Macintosh to Microsoft Windows. A frequently heard objection to migration is the fact that the results are often unpredictable, mostly because of a lack of or because the process has not been fully tested. When a new software version comes on to the market, many people carry out a straightforward update of their documents. Not infrequently, this leads to a loss of information, whether this relates to the contents, structure, appearance or context of the file. The new software does not always read the file in the same way as the original software, with the consequence that the contents and functionality can be lost. The results of migration are difficult to predict, unless a substantial amount of work is first done regarding the specifications of the source and target formats. Migration can influence the authenticity of a document. Each

[8] Raymond A. Lorie, *Long Term Preservation of Digital Information (2000)*

[9] The InterPARES report, *Preservation Strategies for Electronic Records, Round 1 (2000-2001) Where We Are Now: Obliquity and Squint?* (2001), features the results of a 2000-2001 survey of recordkeeping institutions, in which 4 out of 13 projects identified migration as their preservation strategy. This was the most prevalent approach. See also Margaret Hedstrom, Digital Preservation: Problems and Prospects (2001); also Jeff Rothenberg and Tora Bikson, *Digital Preservation: Carrying Authentic, Understandable and Usable Records through Time* (1999).

document that is preserved must be preserved 'authentically', because otherwise the meaning and validity of the archival record cannot be guaranteed. This has both legal and archivistic implications. The Testbed project has dedicated another white paper to this subject: *Migration: Context and Current Status* (December 2001).

7. *XML.* This abbreviation stands for Extensible Markup Language, a language for enriching data with information about structure and meaning. It is an open standard, defined by the World Wide Web Consortium and is platform independent. Conversion of files to XML format can viewed as a particular type of migration technique. XML is also regarded as the most promising original data format for archiving and interoperability, and therefore deserves to be thought of as a preservation approach in itself. The rest of this paper addresses itself to XML.

2 *XML in the Regulation for ordered and accessible archival records*

In the Netherlands, the 1995 Archival Law prescribes how government organisations should deal with their archival records. On 3 March 2002 the Regulation for Ordered and Accessible Archival Records came into force. In addition to PDF and other standards, XML and some of its associated standards take a prominent place in this official document. The Regulation can justly be viewed as bold and visionary, because it prescribes on the one hand a young open standard such as XML and, on the other, a proprietary standard such as PDF. In this chapter, we take a look at the contents of the Regulation.

2.1 Definitions

The Regulation first of all defines a number of terms. This white paper attempts as far as possible to follow this terminology. Below, a number of the definitions from Article 1 are cited:

In this regulation the following terms are defined:
(c) file (bestand): a body of data in a single storage format;
(d) operating system (besturingsprogrammatuur): the software which is responsible for control of an information system;
(f) digital archival record (digitale archiefbestanden): archival records which can only be accessed with the help of a software application;
(k) storage format (opslaagformaat): the encoding according to which data are stored on a storage medium;
(l) platform: the combination of hardware and operating system on which a software application runs;
(m) structure (structuur): the logical relationship between the elements of a document or of an archive;
(n) software application (toepassingsprogrammatuur): the software that is responsible for supporting the execution of a business process;
(o) form (vorm): the visible rendition through which structure and formatting are presented;

Article 2 includes the following relevant text:

The caretaker (zorgdrager) ensures that for each archival record the following aspects can be preserved in perpetuity:

a. the original content, structure and form, insofar as the content, structure and form must be made known for the execution of the relevant business process; and
b. within which time period and under which task (“taak”) or *handeling* [broadly equivalent to ‘business process’ but with a particular defined meaning in relation to government record keeping] the record was received by or created by the government organisation; and

the relationship with other archival records received by or created by the government organisation.

In the article above the crucial terms *content*, *structure* and *form* are introduced. There is also an implied reference to *context*, though this term is not explicitly used. *Behaviour*, which the Testbed views as the fifth property of a digital object, is not mentioned.

2.2 Pièce de résistance of the Regulation: thirteen standards

Below is given the text from the heart of the Regulation, article 6, wherein no less than thirteen standards and techniques are prescribed:

> Digital records should be, at least at the time of transfer, as defined in articles 12 and 13 of the Archival Law 1995, stored according to the following standards:
> a. for character sets: ASCII (ISO/IEC 8859-1) or Unicode (ISO/IEC 10646-1);
> b. for text files: Portable Document Format (PDF) or SGML or XML accompanied by a stylesheet (XSL, CSS) or TIFF or PDF with the metadata in an XML-wrapper;
> c. for CAD/CAM files; Portable Document Format (PDF) and STEP (Standard for the exchange of product data) as the metadata standard (ISO 10303);
> d. for images (bitmapped): Portable Document Format (PDF) and if compression is used: ITU T4 or ITU T6;
> e. for databases: the original storage format or ASCII (flat file, with separator tokens), accompanied by documentation, preferably as an XML-DTD, about the structure of the database (at least encompassing the complete logical data model with a description of the entities); queries should be stored in the query language SQL (SQL2).

It is interesting that for text files on the one hand, 'character-based' standards such as SGML and XML (based respectively on ASCII and Unicode) are prescribed, and on the other hand, the binary formats TIFF and PDF. It is not explained in the Regulation why for text documents, TIFF (Tagged Image File Format) is one of the chosen formats; whereas for images themselves PDF is chosen instead of TIFF (or another image format such as JPEG or GIF). This paper will not address images any further, except in Chapter 3 ('The Extended Family of XML') where we mention Scalable Vector Graphics, a standard for storing images, based on XML. Item (e) raises the question: why use a DTD if the content of the database itself does not need to be stored in XML? (see the Arbeidsvoorziening case study in Chapter 4 for a discussion of how to handle databases). The Regulation says nothing about the question of whether a database should be saved as a transactional system rather than as a series of snapshots in time.

2.3 Further explanation

In the explanatory notes to Article 6, XML is discussed further:

> Another option for text files is XML (Extensible Markup Language) in combination with a stylesheet. Originating in the publishing world,[10] XML is a de facto standard, rather than an official one. XML is a subset of the standard SGML (Standardised Generic/alised Markup Language[11]) and is related to the web language HTML. With the help of XML the structure of a (specific type of) document can be saved in a so called Document Type Description[12] (DTD). For the specification of the form of documents a style-sheet can be used. Cascading Style Sheets (CSS), Extensible Stylesheet Language (XSL), or XSL Transformations (XSLT) can be used. Finally, the contents of a document can be stored in ASCII-format[13] with XML "tags".

[10] SGML is widely used in the publshing world, XML actually originated in the internet world.
[11] *Standard Generalised* Markup Language is meant here. In some cases, the term 'generic' is indeed used in place of 'generalised'.
[12] More standard is: Document Type *Definition.*
[13] In fact XML supports Unicode, the successor to ASCII (see the following chapter).

Regarding stylesheets, further explanation is required. A stylesheet consists of instructions to a program on how to generate for example a PDF or HTML file, on the basis of an XML file. By preserving stylesheets, you have just a recipe on the basis of which the appearance can be produced; there is no guarantee that this will reproduced identically in all circumstances (see also the following chapter).

3 *XML and its family of standards*

This chapter describes XML and related standards such as SGML and XSL, that are named in the Regulation.

3.1 Hors-d'oeuvre: form, formatting, structure and content[14]

To give a foretaste of the central idea of XML, let us take this document, which you are reading on a computer screen or on paper. In both cases you can see from the *formatting* of the published version that this sentence is a part of the section with the title "Hors-d'oeuvre: form, formatting, structure and content". This is apparent because of a difference in style: the title is bold and italic, whereas the paragraph text is not. Both use the 10 point Arial font. The global structure of the white paper is evident from, amongst other things, the *formatting* of the chapter titles, which are 18-point bold and italic. This type of formatting information is hidden in the digital file, the storage format of which can only be read with the help of the original software application, in this case MS Word® (at this moment the de facto standard word processor). In the paper *form*, this explicit information is lost.[15]

For this white paper, the formatting was pre-defined in the form of a template. But imagine that the final *form* of the publication medium (for example the World Wide Web, where titles can be given different colours according to the browser settings) was not known. In that case, it would be desirable to introduce as little *formatting* as possible, so that when the content and the target were decided, the document could be cast in its ultimate form. Accordingly, it is preferable not to fix on a specific *software application* with its own *storage format*, but to remain independent through the use of an open standard, in which all information is explicitly accessible. XML has become so popular in such a short time, precisely because it addresses this kind of problem. Let us look at how a part of this paper (greatly simplified) looks in this language or *storage format*:

```
<Whitepaper>
  <PaperTitle>XML and Digital Preservation</PaperTitle>
  <Chapter>
        <ChapterTitle>XML and its family of standards</ChapterTitle>
        <Lead>This chapter describes (...)</Lead>
        <Section>
            <SectionTitle>Hors-d'oeuvre: form, formatting, structure and content</SectionTitle>
            <Paragraph> To give a foretaste of the central idea of XML, let us take this
            document, which you are reading on a computer screen or on paper. In both cases
            you can see from the <RegulationConcept>formatting</RegulationConcept> of the
            published version that this sentence is a part of the section (...)</Paragraph>
        </Section>
  </Chapter>
</Whitepaper>
```

[14] In this section, the key terms from the Regulation are given in italics.
[15] To get an impression how inaccessible this type of format is, try opening the file of this text document with a simple text editor such as Notepad. You could carry out a *migration* to the more independent RTF (Rich Text Format) or a *migration* to the version of this word processor for the Apple Macintosh platform, with its different hardware and operating system.

The first thing that stands out is that the XML *file* consists of recognisable characters, that can be read with a simple editor. More advanced programs[16] recognise the structure from such a file and show - in order to help the human reader - the different building blocks of the text in different colours.[17] In the case of this paper, we present the information in simple black and white. An important difference is that, for example, the title of this section (given in the tag <SectionTitle>) is in the same font as the rest of the XML text (in this case 9-point Arial). An XML file therefore contains no *formatting* and, as with other *flat files*, in principle one can use only a single font. To make it easier for the reader, tabs can be used to make clear the nesting structure of the XML file. The 'root' element, Whitepaper, comprising everything between the opening tag <Whitepaper> and the closing tag </Whitepaper>, is a level lower than 'child' elements such as PaperTitle and Chapter. Chapter, in its turn, consists of elements ChapterTitle, Lead and Section and the XML-tree proceeds in this way.

Naturally, there is much more which could be said about this small piece of XML. Let us leave it for the moment with a remark about the structure of the white paper, which is made explicit in XML format with the use of tags. These tags contain the names of the elements, which describe what the content of the element is, rather than how it should look. This can clearly be seen in the element <RegulationConcept>form</RegulationConcept>, a part of the Paragraph element. Through the use of tags, it is shown that 'form' is a concept from the Regulation. It can be decided later how this should be presented, by specifying in a stylesheet that elements of this type should be displayed as italic or in green. In the same way, the term 'de facto', which is also presented in italics, could be tagged as ForeignExpression or Latin and how it should be presented decided later. It should be clear that, because of this mark-up of the contents in XML, this sort of document (depending on the precision of the tagging) can be very easily searched.

3.2 Grandmother ASCII: from bits to characters

The standard ASCII (American Standard Code for Information Interchange) for characters is a shining example of a technological standard which is used worldwide for the exchange of textual information in digital form. A contrast can be drawn with the platform specific EBCDIC of IBM. The establishment of the original ASCII-standard can be compared with the invention of the alphabet. The latter specifies what kind of tokens represent given sounds and the former specifies which combinations of bits correspond to which characters.

Because the original ASCII specification (ISO/IEC 646) defined only 128 characters (on the basis of 7 bits), there was a problem, particularly in Europe, over how to represent letters with accents, or other alphabets such as Greek. Because of that an extended ASCII was created using 8 bits (thus with twice as many possibilities): ISO 8859; the Regulation opts for ISO/IEC 8859/1 as the character set for the Latin alphabet. However, extended ASCII is still not sufficient for all types of characters and so the Unicode initiative came into being, in order to represent all characters from every written language in the world (ISO/IEC 10646, see www.unicode.org). The Regulation prescribes Unicode alongside ASCII, and this standard forms the basis of XML data. It is expected that ultimately all operating systems and software applications will make use of Unicode. Because in most cases, only a part of the enormous Unicode is actually used, in practice there are various approaches to optimising performance. A full explanation of this material falls outside of the scope of

[16] For example Microsoft Internet Explorer 6.0.
[17] Because of the possibility that the reader has a black and white paper copy, we will proceed without assuming this aid to readability is available.

this paper. For XML it is important to know that at the start of an XML file there must be a specification of which character encoding is used. In the case of the piece of XML given in the previous section, the beginning of the file is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<Whitepaper>
        <PaperTitle>XML and Digital Preservation</PaperTitle>
(…)
```

The 'Processing Instruction' in the first line specifies that the file uses Version 1.0 of XML and that the characters are encoded using UTF-8.[18] UTF stands for UCS Transformation Format and UCS in turn stands for Universal Character Set. For the technically less interested reader it is sufficient to know that in principle SGML uses ASCII and XML uses Unicode.

Perhaps it goes without saying that fonts such as Arial and Times are something distinct from the abstract characters such as those defined by ASCII and Unicode (which computers use to communicate with each other). Fonts give the characters their ultimate form on screen or on paper, so that the human reader can see them.[19]

3.3 The mother of XML: SGML

If ASCII is the mother of SGML, then GML (Generalised Markup Language) is the father. GML brought into practice the idea of generic data storage, in which the content was coded separately from the form. This forerunner was developed in 1969 by a number of pioneers at IBM, who also contributed to the definition of SGML (Standard Generalised Markup Language), which became an independent standard in 1986 (ISO 8879:1986). SGML has been taken up with much success by Boeing for the description of aircraft parts and the production of handbooks. SGML is also widely and successfully used in publishing.

Because, amongst other things, SGML is rather complicated and because there is relatively little software available to work with it, it has never found widespread acceptance.[20] This is not true of the two related languages, XML and HTML, which - because they are applications of SGML - could be called its daughters.[21]

Two aspects of SGML are handled below: markup and DTD, which have been passed on to HTML and XML.

3.3.1 Markup

'Markup' is sometimes defined as 'formatting'. This last term suggests that a language such as SGML defines the formatting, which is not really the case (to describe formatting in SGML, the language DSSSL[22] was created). Markup could better be defined as 'enriching'. The markup consists of tags which always indicate the beginning and end of an element and in principle describe the content of the element. The markup can be thought of as embedded metadata, which gives information about the data itself, i.e. the content. This is a good moment to introduce another feature of

[18] This is also the default encoding, so if no explicit specification is given then UTF-8 is to be assumed.
[19] And subsequently associate the characters with a sound, if they are read out loud.
[20] It is appropriate that the Regulation names this language. If an organisation wants to choose now between SGML and XML, it is probably better to go for the latter.
[21] XML, as a subset of SGML, is stricter and, for example, does not allow inclusion and exclusion (techniques which could be used for example to specify that an image be included in a book but not in a footnote).
[22] DSSSL (Document Style and Semantics Specification Language) is not specified in the Regulation. With the advent of XML, CSS en XSL it has taken something of a back seat.

SGML (and HTML and XML). We will re-use the section of XML from earlier (which is also correct SGML):

```
<Whitepaper>
  <PaperTitle>XML and Digital Preservation</PaperTitle>
  <Chapter id="H3">
        <ChapterTitle>XML and its family of standards</ChapterTitle>
        <Lead>This chapter describes (...)</Lead>
        <Section confidentiality="high">
            <SectionTitle>Hors-d'oeuvre: form, formatting, structure and content</SectionTitle>
            <Paragraph> To give a foretaste of the central idea of XML, let us take this
            document, which you are reading on a computer screen or on paper. In both cases
            you can see from the <RegulationConcept>formatting</RegulationConcept> of the
            published version that this sentence is a part of the section (...)</Paragraph>
        </Section>
  </Chapter>
</Whitepaper>
```

We have now added an 'attribute' to two of the elements.[23] The element Chapter now has an identifier 'H3' and Section has an attribute 'confidentiality' with value 'high'. In the first case, this attribute can be used for example to add pointers to this element from elsewhere in the document. An attribute gives additional information about an element and so could be thought of as 'metametadata'.[24]

3.3.2 DTD: structure for a document type

Via a so-called Document Type Definition (DTD), one can explicitly define the structure of the tags in an XML document. Most importantly, rules can be given about, for example, how often an element can appear. Below is given the DTD for our example section of XML:

```
<!ELEMENT Whitepaper (PaperTitle, Chapter+)>
<!ELEMENT PaperTitle (#PCDATA)>
<!ELEMENT Chapter (ChapterTitle, Lead?, Section*)>
<!ELEMENT ChapterTitle (#PCDATA)>
<!ELEMENT Lead (#PCDATA)>
<!ELEMENT Section (SectionTitle, Paragraph+)>
<!ELEMENT SectionTitle (#PCDATA)>
<!ELEMENT Paragraph (#PCDATA | RegulationConcept?)*>
<!ELEMENT RegulationConcept ANY>
<!ATTLIST Chapter
    id ID #IMPLIED
>
<!ATTLIST Section
    confidentiality (none | low | high) #IMPLIED
>
```

With the help of a formal notation, we can describe what each element consists of. Thus the element Whitepaper (first line), for example, is made up of (not more than and not less than one) PaperTitle and one or more Chapters. The following line shows that PaperTitle is a 'leaf of the tree': it does not consist of other elements, only text. At the bottom, the two attributes are defined using ATTLIST elements.

---

[23] For those reading the paper in black and white, the additions have been made in bold. The XML editor used to produce this snippet, by default displays attributes in red.

[24] A minimalist purist such as Bert Bos of the W3C is of the opinion that XML would have been more appealing if attributes had been left out of the standard. The same information can be added by using a sub-element.

In a DTD the structure of a particular type of document can be specified. It can thus serve as separate documentation of the structure, as a recipe for new documents of this type and as a grammar which can be used by software to determine whether a document is 'valid' (see 3.6).

3.4 The sister of XML: HTML

The different versions of HTML (Hypertext Markup Language, the first version dating from 1990) are defined in SGML using a DTD. Version 4 of HTML[25] is actually presently defined as an XML application: XHTML. Thus as well as being a sister of XML, HTML has also become its daughter (thus software which can process XML can also process XHTML).

Often the most important innovative property of HTML is forgotten: the possibility of using 'hyperlinks'. This concept could be called the father of HTML and forms one of the ingredients of the success of the World Wide Web (WWW). By creating links to elements of the same document, or to other documents on the Internet, a dynamic integrated system has been created, that has brought with it an alternative approach to publishing and reading.[26]

HTML can be described as a formatting language because its tags can refer to formatting concepts such as 'bold' (the tag 'B'). On the other hand, HTML tags such as H1, H2 and H3 that define different levels of headings, do so in quite an abstract way.

HTML is attractive and popular, but also has considerable limitations, which have lead to XML being created: its apparently less interesting but much more intelligent sister. The most important limitations of HTML are:

- It is not extensible: the tags are fixed in the HTML standard.
- Content tagging is very limited: the content and formatting are inseparably tied together.
- The visual presentation is dependent on the settings of the browser. Thus one can never be certain that the reader will be shown precisely what the author of the HTML intended.

3.5 The standard XML

As we have seen, XML builds on the solid foundation of SGML, makes use of the universal Unicode and has learned from the experience of HTML. From the fact that Version 1.0 of the XML specification from 1998[27] has so far had no successors, it seems that this standard hit the bullseye. Given the prestige of the World Wide Web Consortium, it is strange that the explanatory notes to Article 6 of the Regulation describes XML as a "de facto standard". In this section, we present further explanation of XML. We will also describe the software needed to process XML.

3.5.1 Further XML: namespaces, empty elements, etc.

The example from the first section is repeated below and expanded with metadata in the traditional sense: by way of an example, the details of the Testbed Project Manager and the author of the paper are given (with new sections in bold).

```
<?xml version="1.0" encoding="UTF-8"?>
```

[25] See http://www.w3.org/TR/html4; version 4.01 dated 24 December 1999.
[26] And a great challenge for those responsible for archiving this new publishing medium.
[27] Released by the World Word Wide Web Consortium on 10 February 1998 (see http://www.w3.org/TR/2000/REC-xml-20001006).

```
<Whitepaper xmlns="xml.ictu.nl" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://www.ictu.nl/xml/schemas WhitepaperICTU.xsd">
    <Meta>
        <ProjectManager>
            <Name>Jacqueline Slats</Name>
            <Function>Project Manager Digitale Duurzaamheid</Function>
            <Email>Jacqueline.Slats@ictu.nl</Email>
        </ProjectManager>
        <Author>
            <Name>Hette Bakker</Name>
            <Function>Senior Consultant CGE&amp;amp;Y</Function>
            <Email>Hette.Bakker@cgey.nl</Email>
        </Author>
    </Meta>
    <PaperTitle>XML and Digital Preservation</PaperTitle>
    <Chapter id="H1"></Chapter>
    <Chapter id="H2"/>
    <Chapter id="H3">
        <ChapterTitle>XML and its family of standards</ChapterTitle>
        <Lead>This chapter describes (...)</Lead>
        <Section confidentiality="high">
            <SectionTitle>Hors-d'oeuvre: form, formatting, structure and content</SectionTitle>
            <Paragraph> To give a foretaste of the central idea of XML, let us take this
            document, which you are reading on a computer screen or on paper.  In both cases
            you can see from the <RegulationConcept>formatting</RegulationConcept> of the
            published version that this sentence is a part of the section (...)</Paragraph>
        </Section>
  </Chapter>
</Whitepaper>
```

In the above XML file, we have added two more Chapter elements (with id values 'H1'and 'H2'). From this we can see that an element can also be empty. A shortened notation for an 'empty element' can be seen in the second Chapter element: in this case the tag ends in "/>" instead of the usual ">". Using this special exception[28] to the rule that an element must have an opening and closing tag, the element can be fully described with a single tag.

The XML file now includes a trio of attributes in the Whitepaper element. The abbreviation 'xmlns' in the first two points to the concept of 'namespace'. It would be going too far in the context of this white paper to go any deeper into this subject: we just note that namespaces make it possible to use the same element name in different places in the document. Multiple definitions of the element 'Name' for example do not then lead to confusion. From the reference to the HTTP protocol in the last two attributes, it can be seen that XML makes use of the WWW in order to publicise and locate the unique definitions of elements of an XML file.

Finally, for the reader who wants to know the details: in the element Function of the element Author in the element Meta, there is the entry '**CGE**&**Y**' (which for example in HTML must be written 'CGE&Y'). With the assistance of the 'entity' & the ampersand character is 'escaped': processing software knows from this that the ampersand character should be taken literally and not used as a special symbol that indicates the beginning of an entity. Like the angle bracket characters < and > (entities < and >), the & character is a symbol used by the computer to read and analyse XML.

[28] SGML and HTML are less strict in this respect.

3.5.2 XML is readable by humans and machines

This last item brings us to the software used to process XML. Ideally, the user should never be confronted with all of the brackets and symbols, rather the XML coding should remain behind the scenes. A piece of software that can read XML is known as an XML parser. Only if an XML file correctly follows the XML specification (e.g. by ensuring that all elements are correctly closed) can the parser process it successfully. Such so-called 'well-formed' XML satisfies the basic grammar of XML.

The COVAX project, for example, has mapped out the available software for XML processing.[29] The list of software grows day by day. What's more, many existing software applications are rapidly being 'XML-enabled'.

3.6 Description of the structure: XML-Schema

XML has inherited the DTD mechanism from its mother, SGML. In fact, because a DTD can define only very limited datatypes and is not itself XML, it is being pushed aside by another standard: W3C-schema.[30] Officially this standard is called XML Schema Definition Language (XSDL), but in practice the name W3C-schema or XML-schema is more often used. Because this standard is relatively complicated, there are calls to find a simpler language to specify the structure of an XML document. Only the future will tell whether W3C-schema will be broadly accepted; in this paper only this schema standard is covered. At present this language is becoming more and more widely supported, but long-standing XML users continue to make use of DTDs.

3.6.1 The (simplified) schema for this paper

In order to show how an XML-schema actually looks, the schema is given below for the type of document given in Section 3.5.1. From the first line, it can be seen that the schema is itself XML, but in comparison to a DTD it is much more verbose and complex.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="xml.ictu.nl" xmlns="xml.ictu.nl"
xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="Whitepaper">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Meta">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="ProjectManager" type="personType"/>
                            <xs:element name="Author"  type="personType"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="PaperTitle" type="xs:string"/>
                <xs:element name="Chapter" maxOccurs="unbounded">
                    <xs:complexType>
                        <xs:sequence minOccurs="0">
                            <xs:element name="ChapterTitle" type="xs:string"/>
                            <xs:element name="Lead" type="xs:string" minOccurs="0"/>
                            <xs:element name="Section" maxOccurs="unbounded">
```

[29] See http://www.covax.org/public_docum/p_documets.htm. (Note the 'wrong' spelling of 'documents' here is indeed the correct URL).
[30] Accepted as an official W3C Recommendation on 2 May 2001: see http://www.w3.org/TR/xmlschema-1/ and http://www.w3.org/TR/xmlschema-2/.

```
                                    <xs:complexType>
                                        <xs:sequence>
                                            <xs:element name="SectionTitle" type="xs:string"/>
                                            <xs:element name="Paragraph"
                                                        maxOccurs="unbounded">
                                                <xs:complexType mixed="true">
                                                    <xs:choice maxOccurs="unbounded">
                                                        <xs:element name="RegulationConcept"
                                                            minOccurs="0"/>
                                                    </xs:choice>
                                                </xs:complexType>
                                            </xs:element>
                                        </xs:sequence>
                                        <xs:attribute name="confidentiality" use="optional">
                                            <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                    <xs:enumeration value="none"/>
                                                    <xs:enumeration value="low"/>
                                                    <xs:enumeration value="high"/>
                                                </xs:restriction>
                                            </xs:simpleType>
                                        </xs:attribute>
                                    </xs:complexType>
                                </xs:element>
                            </xs:sequence>
                            <xs:attribute name="id" type="xs:ID"/>
                        </xs:complexType>
                    </xs:element>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:complexType name="personType">
            <xs:sequence>
                <xs:element name="Name"/>
                <xs:element name="Function"/>
                <xs:element name="Email"/>
            </xs:sequence>
        </xs:complexType>
</xs:schema>
```

3.6.2 Granddaughters of XML: XML vocabularies

XML itself is just a so-called meta-language, providing a syntax with which a genuine language can be specified. With the help of an XML schema, such languages can be precisely defined. The result is known as an XML vocabulary: as it were, a granddaughter of XML. One could say that XML-schema, as mother, provides the syntax, and a concrete application area, as father, takes care of the semantics. In the case of the schema presented above, this could make use of a vocabulary for publications of ICTU ("IctuML" say) or another organisation. At this moment, there are many initiatives to develop vocabularies; only a small proportion of these will break through as a generally accepted standard. A user or organisation must weigh up whether it is more convenient to use an existing vocabulary or to use XML-schema to create one of its own. Experience shows that the associated discussion of how a document type is built up (and how it should be built up!) can take a great deal of time.

3.6.3 Control of the structure: validators

An XML document can declare at the start that it satisfies a particular schema (or DTD) - though it is not obligatory to have a schema.[31] If a document states that it is an instance of an abstract document type[32] as specified in a schema, then one can check if this is the case with the help of a validator. This piece of software (which may come built-in to an XML editor for example) takes a (well-formed) XML document and checks it against its schema. The result of this validation process is the answer: valid or not valid; in the latter case many validators give help in finding the grammatical errors.

Validation can be done at different moments, for example at the source, thus before sending an XML document, or on receipt of an XML instance. This is a well-tried means of enforcing the use of a template[33] and of checking the quality of data.

3.7 Appearance: daughter XSL and partner CSS

Because of its structured and consistent construction, an XML document is easy for a computer to read. For the average human user, it is just a half-baked form that still needs to be given a more accessible form (without the angle brackets). With the help of the stylesheet mechanism, such a form can be generated; for example an HTML, PDF, or PostScript file which can be presented with the help of the associated application software on the computer screen, on paper or audibly. In this section the stylesheets named in the Regulation and the associated software are described.

3.7.1 Cascading Style Sheets

As explained above, HTML has become an enormous success, but this language has serious limitations. Suppliers rapidly began to formulate their own extensions to HTML to provide extended formatting options. Thus the Netscape browser introduced the CENTER element, that cannot for example be interpreted by speech generating software.[34] In order to rescue standard HTML, in 1996 the W3C introduced the Cascading Style Sheets (CSS) standard as a means of facilitating formatting. CSS, which forms a separate language (i.e. it is not SGML or XML), presently has two generations: CSS1 and CSS2.[35] To give an impression of how CSS looks and what information it contains, a fragment is presented below, without further explanation.

```
BODY {
    font-family: Arial, Helvetica, sans-serif;
    margin-left : 0px;
    margin-top : 0px;
    margin-bottom : 0px;
    margin-right : 0px;
}
.result{
    background-color : #FFFFFF;
}
.resultheader{
    background-color : #BCBCBC;
```

[31] For example the document in 3.5.1 refers to the schema WhitepaperICTU.xsd.
[32] 'Instance' is a word often used in computing literature to refer to an object which is an example of a class of objects with similar features. In more philosophical terms it could be thought of as describing a concrete chair rather than the idea of a chair.
[33] A standard text editor allows the user to deviate from a template.
[34] The Regulation does not mention rendering of text as audible speech, although it does name loudspeakers as a type of physical medium.
[35] CSS can be used with XML and XSL, as well as with HTML. See http://www.w3.org/TR/NOTE-XSL-and-CSS.

```
}
.inactivelink{
    color : Gray;
}
.big_black
{
   COLOR: black;
   FONT-FAMILY: Arial, Helvetica, sans-serif;
   FONT-SIZE: large
}
```

3.7.2 XSL-FO en XSLT

CSS and DSSSL (see 3.3.1) could both be regarded as father of the Extensible Stylesheet Language[36] (XSL). XSL makes use of the experience gained with these older standards and uses the syntax of XML. A consequence of this is that this language, like the XML-schema, is not very compact. It could be said that XSL consists of two parts: XSL-FO and XSLT. XSL-FO (Formatting Objects) is very rich: for example one can use it to formulate a stylesheet that gives instructions on how to take an XML document and use it to generate a PDF file.[37]

XSL is much more powerful that the name stylesheet suggests. With the help of XSL one can transform an XML document to, amongst other things, another XML document or an HTML file. Filtering is also possible; so an XSL document can specify that as part of the transformation to an HTML or XML document for external use, particular elements should be left out (for example dependent on the attribute 'confidentiality' introduced in 3.3.1). A single XML source document can be transformed by one or more stylesheets, for different purposes. The part of XSL which looks after this kind of transformation has been crystallised out in a separate standard: XSL Transformations (XSLT).

As an illustration, we give below the beginning of an XSL-stylesheet which formed part of a demo developed for the Arbeidsvoorziening (an agency of the Dutch government - see the following chapter). The bold text indicates the HTML tags which show that the source XML file must be transformed to HTML. One can also see a link to a CSS-stylesheet (cipers.css).

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xsl:template match="/">
      <html>
         <head>
            <link rel="stylesheet" href="/cipers.css"/>
         </head>
         <body>
            <table class="main_text" width="80%" align="center" cellspacing="0">
                     <tr>
                     <td>
                        <font class="big_blue">Result</font>
                     </td>
                  </tr>
            </table>
            <xsl:apply-templates />
         </body>
```

[36] Published on 15 October 2001 by the W3C; see http://www.w3.org/TR/xsl.
[37] For generation of PDF see also e.g. http://www.tallcomponents.com/.

```
    </html>
</xsl:template>
<xsl:template match="@expiry-date">
  <xsl:variable name="date">
      <xsl:value-of select="."/>
  </xsl:variable>
  <!-- Day -->
  <xsl:value-of select="substring($date, 7, 2)" />
  <xsl:text>-</xsl:text>
  <!-- Month -->
  <xsl:value-of select="substring($date, 5, 2)" />
  <xsl:text>-</xsl:text>
  <!-- Year -->
  <xsl:value-of select="substring($date, 1, 4)" />
      </xsl:template>
```

3.7.3 Stylesheet-processors

A stylesheet processor is required actually to transform XML according to the instructions in a stylesheet. This software is more and more often included as a part of browsers and other software. Aside from that, XSL and associated programs are not the only possibility for transforming XML documents. Experienced programmers may be able to get quicker results using the programming language Perl, for example.

3.8 Extended family of XML

After our account of XML itself and the other standards in its close family, we introduce briefly below a number of other related standards.[38]

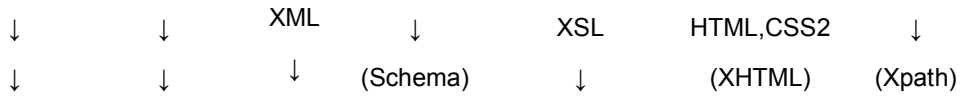
- XPath is a standard for locating information within an XML document. XML-schema and XSLT make use of XPath.
- The query language XQuery can be thought of as the XML equivalent of SQL. This new standard builds on XPath.
- SVG (Scalable Vector Graphics) is an XML-based standard intended for storing images as abstract geometrical forms, instead of as a collection of points (a bitmap).[39]
- The Resource Description Framework[40] offers a complex way of setting up XML documents so that they can be more easily interpreted as metadata. An initiative which has some features in common is Topic Maps (ISO/IEC 13250). Attempts are being made at present to combine these two standards.

3.9 Summing-up

By way of recapitulation, below is given a global collection of the standards we have discussed, grouped according to generations and purpose:[41]

Generation	Characters	Content	Structure	Transformation	Formatting	References
↓	ASCII	SGML	DTD	(DSSSL)		(HyTime)
↓	Unicode	↓	↓	↓	HTML,CSS1	HTML

[38] For additional and up-to-date information, visit the website of the W3C: www.w3.org.
[39] SVG, like XML, aims to provide more intelligent content, whilst a bitmap, like PDF and HTML, stores a frozen form.
[40] See http://www.w3.org/RDF/.
[41] Those standards given in brackets are those which are probably too old or too new to be taken up in the Regulation.



To try to explain the origins and relationships between XML, XSL and (XML-) schema, a number of 'equations' are given below:

XML = Unicode + SGML + XML-grammar
Schema-language = XML + Schema-grammar
XSL = XML + XSL-grammar (+ CSS + DSSSL)

The above comparisons relate to the three named languages. Below we give similar relationships for the concrete 'renditions' which can be made using these languages:

XML-document = content (data) + metadata + structure
Schema = possible structure + datatypes
XSL-stylesheet = transformation - c.q. formatting rules

To give an insight into what application software can do with these languages, we give the following combinations:

XML-document + parser → well-formed?: yes or no
Well-formed XML-document + parser → process XML-document
Well-formed XML-document + schema + validator → valid: yes or no
Well-formed (+ valid) XML-document + XSL-stylesheet + XSL-processor → transformed XML-document or formatted document (e.g. HTML or PDF)

Finally, we present rules for how with the help of (generated) HTML or PDF a publishable form can be presented to the user:

HTML-document (+ CSS-stylesheet) + browser (viewer) → rendition (on the screen)
HTML-document (+ CSS-stylesheet) + browser + printer → rendition (on paper)
HTML-document + speech converter → sound
PDF-document + viewer → rendition (on the screen)
PDF-document + viewer + printer → rendition (on paper)

4 *XML and digital preservation in practice*

In Chapter 2 we saw that the Regulation prescribes a large number of standards for digital preservation. Chapter 3 attempts to map the origin and function of the XML-related standards. In this chapter we first address the question of what the 'unique selling point' of XML is for digital preservation. After that we turn back to the three strategies from Chapter 1 in which XML plays a role: encapsulation, migration and naturally XML itself. For each of these approaches, a case study is given. General aspects such as metadata, security and storage are described for the most relevant strategy. At the end of this chapter and the white paper, we give a summary of the valuable properties of XML and remark on its future for digital preservation.

4.1 XML and digital preservation

In the digital world XML has acquired a solid position and its take-over is in progress. It is steadily becoming the lingua franca for digital data exchange, a communal language comparable with English now, or Latin in the Middle Ages. XML is of the greatest importance for digital preservation, not just because of this widespread uptake, but also because it protects the Achilles' heel of digital documents: the dependence on obsolete operating systems and application software. It does this by being platform- and software-independent. The separation of content, structure and appearance plays an important role here. Because much of the software-dependency is associated with the appearance or form (for example on the software of Adobe in the case of PDF), the chance is much greater that the abstracted *content* in XML can withstand the course of time. A reasonably intelligent person or computer in a few hundred years should be able to decipher a digital object written in XML. Because XML, like Latin,[42] will eventually become a dead language, our distant descendants must be able to lay their hands on the XML specification[43].

Chapter 3 showed that some aspects of XML are already on their way out (for example the DTD) and others must still be proved in practice (such as XML-schema and XSL). Neither a standards organisation nor a ministry can enforce general acceptance. Indeed, one should not forget that XML is just four years old and many of its children are still in the incubator, or just out of it. Thus it cannot reasonably be expected that the XML family will immediately solve all of the problems of digital longevity. On this point, we can also expect little help from the software providers. They are likely to continue to try to establish monopoly positions by introducing proprietary standards;[44] furthermore XML applications such as digital market places and web services are much more commercially attractive than digital longevity. It is thus important, like the Regulation, not just to wait for something to happen but to take an active approach. In the area of digital preservation, XML is so far little tested; the arguments in favour of this language are largely theoretical.

[42] The Vatican attempts to keep Latin alive by defining new Latin words for new concepts. Latin and Greek, like XML, are very 'extensible': one can easily put together new words such as cardiogram or automobile which can be used worldwide.
[43] They will also need the Unicode table, a schema for the document type with documentation and perhaps a Dutch and English dictionary (for the meaning of tag names).
[44] Or in the case of hardware suppliers, to work towards widespread use of emulation, which requires a lot of computer memory and hence hardware. Many versions and short lifetimes of software and hardware and the associated conversions and migrations also have a positive effect on the turnover of the manufacturers.

4.2 XML versus PDF?

Often XML and PDF are put forward as two rivals from which one must choose in order to preserve a document for the long-term. In this ideological struggle the other two standards named in the Regulation, TIFF and SGML, are at a disadvantage. Because PDF and XML are so complementary, it is actually more appropriate to decide to use both XML *and* PDF for preservation of a document than to choose between XML and PDF. Choosing both standards is also a form of risk spreading: if one of the formats can no longer be read after a hundred years, then one still has the other. Ideally, an open standard should develop which can take over the role of PDF, so that the safe storage of the digital legacy is not dependent on a single company.

In Chapter 3, the reader was able to see that PDF can be generated from XML with the help of XSL-FO. Converting a PDF file to XML (or another format) is in contrast a Herculean task. It is easier to bring a mammoth back to life using its preserved DNA (read XML) than on the basis of a photo (read PDF).

4.3 Questions and objections

Will XML last a hundred years? That is difficult to predict. If one considers the speed with which this language has been accepted and integrated, one could expect that, decades from now, our computers will still be able to work with the angle-brackets of XML. A development comparable to the evolution of ASCII into Unicode could take place. Even if XML falls into disuse, it should be reasonably easy to convert the XML files into a new format. In XML, the data are stored in a very structured way and provided with metadata: ideal circumstances for automated migration.

XML lays everything open; can it guarantee authenticity and integrity? That is indeed a point of concern, but it applies not only to XML. After all, binary formats such as PDF can also be manipulated – it even applies to paper documents.

Is XML not too complicated? That depends: the 'hidden-screen' of WP5.1 (where the hidden formatting symbols were made visible) was more complicated. If people understand the concept behind SGML and XML, the rest is technical detail. Furthermore, it is the intention that the everyday user should not come in contact with the XML angle-brackets.

Is there not the danger that suppliers will develop their own proprietary substandards? Yes, there is a danger, because one can use this approach to develop a monopoly position. Microsoft seemed to be going in this direction with its own schema standard, XDR-schema, when they found that the development of the official schema standard by the W3C was going too slowly. In the end the software giant from Redmond conformed with the W3C specification. SQL, which is named in the Regulation as the language for database queries, is an example of a language where suppliers each introduced their own specific features; the question which comes to mind is: which dialect of SQL is meant by the Regulation? In contrast, XML is a strictly controlled language; extensions to XML are proposed to the W3C as separate standards and, after a strong selection process, subsequently accepted as an official Recommendation.

Is it not a lot of work to introduce XML? There is certainly a learning curve, which should not be underestimated. Because XML is being introduced in so many areas, one can expect that over the next few years the knowledge and expertise in this area will become common property. On this foundation, the specific expertise needed for digital archiving can be built up.

4.4 Strategy 4: Encapsulation

This approach is aimed at preservation of the original format. XML is often named as a good language for storing metadata and instructions for the object to be preserved. In this section we review a number of terms used in this context. After a short discussion of metadata, there follows a description of the VERS project.

4.4.1 Wrappers, containers, encapsulation and framework

The Regulation mentions an 'XML-wrapper' as a means of adding metadata to PDF and TIFF files. Although one can imagine what this might entail, there is not (as yet) a fixed meaning for this term. For example, the San Diego Supercomputer Center regard a wrapper as a piece of software that can be used as a 'mediator'.[45] The Roquade project, in contrast, uses the term 'container' for the 'packaging' of digital archival records.[46] A step further than encapsulation is to use XML as a framework to on which a document or parts of a document in e.g. TIFF or PDF format can be hung. a document In this case, XML forms the backbone of a digital archival record.

4.4.2 Metadata

In the previous chapter, we saw that metadata, that is to say data about data, is an integral part of XML (in the form of tags). XML also offers excellent facilities for the storage of metadata in the narrower archivistic sense. For this reason, XML can be used in combination with other preservation strategies, for example with emulation, XML could be the language used to store technical metadata. Adobe, owner of the PDF standard, has recently launched the eXtensible Metadata,[47] that also uses XML for metadata storage.

If a fixed collection of metadata has been agreed (and that is often much more difficult than the technical implementation!), this can be specified in the form of an XML schema, which can be reused by schemas for specific documents. This standardisation is important, because otherwise a digital archive will not know what kind of metadata to expect.

4.4.3 Case study: VERS

In Australia a pioneering project, the Victorian Electronic Records Strategy, has been successfully completed. In the final report (from 1999)[48] it was described with fitting pride that it is possible to preserve electronic archival records for the long-term. To this end a standard format is proposed, with the following features:

- The documents, context and authenticity information must be encapsulated in a single object and not saved separately.
- The data structure must enable metadata to be added in layers (the 'onion model').
- XML must be used for the coding of the encapsulated archival records.
- Each electronic record must have a digital signature.

---

[45] "A wrapper is a piece of software that acts as a translator between the native format of an information source and a commonly agreed protocol (XML for us). The end-user or application interacts with a piece of software called mediator that collects information from multiple wrappers ", page 4 of *Methodologies for the Long-Term Preservation of and Access to Software-Dependent Electronic Records*, http://www.sdsc.edu/NHPRC/Pubs/nhprcfeb2k.doc.

[46] "It was decided to work out the idea of XML containers. So the Archival Information Packages (AIP), to be stored in the electronic archive, will be wrapped in XML." *An electronic Archive for academic communities* (Dekker, R. et al, Nov 2001). The AIP concept originates from the Open Archive Information System (OAIS) model.

[47] See http://partners.adobe.com/asn/developer/xmp/download/docs/MetadataFramework.pdf.

[48] See http://www.prov.vic.gov.au/vers/final.htm.

In the demonstrator which was delivered as a product of the project, the documents themselves were transformed to PDF. Because PDF is a binary format and XML is based on text, before they could be encapsulated in XML, the PDF files were converted to an encoded text format.[49]

4.5 Strategy 6: Migration (to XML)

Structured data such as those found in a database or spreadsheet lend themselves very well to migration to XML. In principle, the contents of database tables can be translated on a one-to-one basis to elements in XML. In addition, a great deal of other information (connected with the technical implementation of the database) should also be transferred to the files to be archived. Migration to XML will have fewer difficulties if in the future the development of a system takes account of the need for a final transformation to XML.

In this section we present a recent project in the area of database migration. There are two important aspects to this, integrity and storage, which are of course also relevant to the other strategies.

4.5.1 Case Study : Databases from the Arbeidsvoorziening

For the Arbeidsvoorziening (Dutch government public employment service) Cap Gemini Ernst & Young (CGE&Y) carried out a proof of concept (POC) project which showed that it is technically realisable to convert the contents of databases to XML. The Arbeidsvoorziening, which at the time of writing is being wound up and its functions transferred to other organisations, must put its digital files in order before they can be transferred to its legal successor(s). This involves in the first instance terabytes of data belonging to more than ten systems. The necessity for data retention is specially acute because of the proceedings to recover overpaid grants by the European Social Fund (ESF).

As was mentioned in Chapter 2, the Regulation contains the direction that databases should be stored in their original format or as a flat file, whilst the structure should preferably be described in a DTD. In the case of the Arbeidsvoorziening, the Dutch National Archivist gave the advice that the content of the databases should be converted to XML. Within a month (March 2002) it was possible to show that:

- The content of a database could be read in a transparent way and converted into raw XML.
- This XML can subsequently be converted stepwise into well-formed XML, enriched with metadata, in the first place regarding the retention period;[50] on the basis of this metadata, data whose 'best-before date' has passed can be destroyed.
- A rudimentary query tool can search the XML and present the results with the help of XSL(T).

The conversion to XML was a sweeping migration (i.e. to a format very different from the original) which answered not so much technical as theoretical questions, for example:

[49] In this case, the well-known standard Base64 was used. This is also widely used in e-mail systems.
[50] On the grounds of the "Basic Selection Document" which specifies the retention period according to each "handeling" (defined in Article 1h of the Regulation as "a complex of activities to fuilfil a task or on the basis of a competency"). A Basic Selection Document is used as an instrument for the selection of archival records.

- What is the domain of the data to be transformed: how can we decide which database tables must be included in the migration (many tables are application specific or not related to the database contents)?
- How far should we denormalise: which tables may be combined without loss of accessibility and authenticity?
- How do we translate the abstract *handelingen* or business processes named in the Basic Selection Document to retention metadata per table or per element?

The Regulation does not specify a standard for preserving the functionality of a database.[51] That there is no generally accepted recipe for this is understandable, as application migration is many times more complicated than data migration. It is however important from an archivistic point of view that future generations have an insight into what was done with these terabytes of data, how they were viewed by the user and in which ways they could be queried. Regarding this last point, the Regulation (in Article 6) specifies SQL as the language for queries. No further instructions are given regarding these queries. Because during the Proof of Concept (POC) the database data were converted to XML, XSL(T) rather than SQL was used for queries. The construction of a full query tool, the choice of storage for the XML files and the setting up of the necessary digital archive fell outside of the scope of the POC and may be addressed later.

4.5.2 Integrity

An aspect that plays a major role in migration is the degree of certainty that the integrity of the migrated data will remain intact. This is certainly the case in a transformation to XML, in which irrelevant data are left out and metadata added: the resulting XML file looks very different to the download from the database. Luckily, in this regard one can make use of the considerable experience in the area of migration. The Arbeidsvoorziening made use of pre-existing migration tools, which included facilities for integrity analysis and making a detailed log of the migration. Through the use of transparent procedures, the Riksarchiefinspectie (inspection service of the Dutch national archives) or a judge must have confidence that the creation and storage of the XML files has not corrupted the data. From this point of view the digital file does not differ essentially from the paper record (which can also have something added to it). The cooperation between man and machine mentioned in Article 1a of the Regulation in the description of the concept of an archive management system is also relevant here.[52]

4.5.3 Storage

The subject of archive management systems brings us to the question of how the products of the migration, the XML files, can be stored. This leads unavoidably to the counter-question: what do you want to do with these files? If one expects to access the files only once in fifty years, then one could consider storing them on a durable physical medium and handing this over to the National Archives. As long as the XML files have been well documented (as required by the Regulation), they can be taken out of mothballs when the time comes.[53] If it is required to have rapid access or to provide a querying facility with access control features, then one must think along the lines of a new database. At this moment, there are two main options: a ('classical')

[51] Explanatory notes to Article 6: " For databases, there does not yet appear to be a specific standard, which fully satisfies the requirements of authencity relating to the appearance of screens and reports".
[52] "The entirety of people, methods, procedures, data collections, storage, processing and communication equipment and other means, intended for the management of archival records".
[53] We don't consider here the problem of preservation of original functionality and queries, mentioned in the case study.

relational database or a native XML database. A further explanation of these options is outside the scope of this paper.

4.6 Strategy 7: XML (from the beginning)

In the previous strategy, the data was converted to XML after creation in some other format. This approach becomes more complicated if we consider unstructured documents, where the structure must later be made explicit in the form of XML tags. This can not always be done fully automatically and will usually require some human intervention. If, for example, we want to distinguish the RegulationConcept element (see 2.1), then it is easier to do this if it has tags associated with it. It is not surprising that there are initiatives to use XML as the underlying format for office productivity software applications. The open source package OpenOffice (see www.openoffice.org) is one example. Given that Microsoft is also making use of XML within Office XP, the trend seems to be towards using XML as the original storage format for office documents.

4.6.1 Will the authentic document please stand up?

If the original format of a document is XML, then this satisfies the requirements of the Regulation. One can also add a stylesheet giving formatting instructions. The file which is published or transmitted will often be one that has been generated with one or more stylesheets. As we saw in Chapter 3, this could be an HTML, PDF or Postscript file. Because this is the form which is ultimately seen by the reader, it is this form, together with information on the time of sending or publication etc., that the archivist will want to preserve. This brings the number of file types associated with archiving of XML up to four:

1. XML-documents (instances)
2. Schemas which specify the structure of a type of XML document
3. Stylesheets containing formatting or transformation rules
4. 'Frozen' forms possibly with timestamps

The Regulation specifies that, as regards text documents, the types 1 to 3 must be preserved.

4.6.2 Case study: outgoing e-mail from the Testbed

In order to investigate the use of XML in practice, a demonstrator has been developed, consisting of two applications: a web service and an extension of the Outlook e-mail software from Microsoft. Outlook is modified so that the e-mail written by the user is created behind the scenes directly into XML format. Also, the user is presented with a metadata form with values to fill in, which are then included with the XML representation of the e-mail message. Finally, the author can preview and then send the e-mail message in HTML format. The validation of the XML, the transformation to HTML and the central storage of the e-mail are handled by a web service. An objective of this pilot is the realisation of the XML transformation and storage without limiting the ease of use of the e-mail software. Two important advantages for the organisation are that (a) official e-mail is validated before sending and has a consistent "house style", and (b) the e-mail messages are stored centrally in XML, accompanied by the required metadata.

4.7 Conclusion: the advantages of XML and a caution

To conclude, we sum up below the most important properties of XML:

- Platform and program independent
- An open standard, widely accepted and applied.

- A practical approach to the concept of separation of content, structure and form
- Extensible and controllable (like a natural language)
- Readable by both humans and machines
- Free

Although XML offers many perspectives in the area of digital longevity, it is important to add a caution that XML is not a miracle solution that must be applied to every digital preservation problem. XML, its related standards and their use form a complex material; much pioneering work still needs to be done.

5 *Bibliography*

Bourret, Ronald	XML and Databases (2002) http://www.rpbourret.com/xml/XMLAndDatabases.htm
COVAX	State of the Art (2000) http://www.covax.org/public_docum/p_documets.htm
Dekker, R et al	An electronic archive for academic communities (Nov 2001) http://www.library.tudelft.nl/e-archive/Documenten/Resultaten/e.archive.acad.comm2.doc
Dürr, E & Lourens, W	Emulation and Conversion: Design and Implementation of an Electronic Archive (Nov 2001) http://www.library.tudelft.nl/e-archive/Documenten/Resultaten/report3.pdf
Dürr, E; Lourens, W; & Meer, K.v.d.	Emulation and Conversion: Organisational and Architectural Overview of an Electronic Archive (2001) http://www.library.tudelft.nl/e-archive/Documenten/Resultaten/reportone13.pdf
Gilheany, Steve	XML for Records Managers (2002) http://www.archivebuilders.com/whitepapers/22033p.pdf
Hedstrom, Margaret	Digital Preservation: Problems and Prospects (2001) http://www.si.umich.edu/CAMILEON/camileon%20Presentations/margaretpresentation.pdf
Hodge, Gail	Best practices for Digital Archiving: An Information Life Cycle Approach (2000) http://www.dlib.org/dlib/january00/01hodge.html
InterPARES	Preservation Strategies for Electronic Records, Round 1 – Where are We now? Obliquity and Squint http://www.interpares.org
Klyne, G	An XML Format for Email Messages (2002) http://www.ietf.org/internet-drafts/draft-klyne-message-rfc822-xml-03.txt
Lorie, Raymond A	A Project on Preservation of Digital Data (2001) http://www.rlg.org/preserv/diginews/diginews5-3.html - feature2
National Library of Australia	A Draft Research Agenda for the Preservation of Physical Format Digital Publications (1998) http://www.nla.gov.au/policy/rsagenda.html
Ploeg, Dr. F. van der	Regeling geordende en toegankelijke staat archiefbescheiden (2002) http://www.nationaalarchief.nl/images/3_2597.doc

Public Record Office (Kew, UK)	Guidelines for the Management, Appraisal and Preservation of Electronic Records (1999) http://www.pro.gov.uk/recordsmanagement/eros/guidelines/default.htm
Public Record Office Victoria (Australia)	Victorian Electronic Records Strategy Final Report http://www.prov.vic.gov.au/vers/published/final.htm
Rothenberg, Jeff & Bikson, Tora	Digital Preservation: Carrying Authentic, Understandable and Usable Records Through Time (1999) http://www.digitaleduurzaamheid.nl/bibliotheek/docs/final-report_4.pdf
SDSC/NHPRC	Methodologies for the Long-Term Preservation of and Access to Software-Dependent Electronic Records http://www.sdsc.edu/NHPRC/Pubs/nhprcfeb2k.doc.
Testbed Digitale Bewaring	Migration: Context and Current Status (Den Haag, December 2001) http://www.digitaleduurzaamheid.nl/bibliotheek/docs/Migration.pdf

6 *Websites*

http://www.digitaleduurzaamheid.nl Website Testbed Digitale Bewaring

http://www.covax.org/ Contemporary Culture Virtual Archives in XML

http://www.nationaalarchief.nl/images/3_2598.doc Website Nationaal Archief; Regeling Geordende en toegankelijke staat archiefbescheiden

http://www.jiscmail.ac.uk/ *JISC* Listserv archives

http://www.w3.org/TR/ W3C Technical Reports and Publications

http://www.interpares.org/ InterPARES Web site

http://www.rlg.org/preserv/diginews/ RLG Diginews website

http://www.prov.vic.gov.au/vers/published/final.htm VERS Final Report site

http://www.pro.gov.uk/recordsmanagement/eros/ PRO (UK) Records Management

http://www.sdsc.edu/NHPRC/Pubs/nhprcfeb2k.doc. San Diego Supercomputer Centre

http://www.antwerpen.be/david/ Project DAVID website

http://www.si.umich.edu/CAMILEON/ Project CAMILEON website