Design criteria for preservation repositories

Frans Dondorp and Kees van der Meer
Delft University of Technology, DIOSE Betake Research Group, Faculty EEMCS
PO Box 5031 2600 GA Delft
{F.P.A.Dondorp, K.vanderMeer}@ewi.tudelft.nl

Abstract

What are the requirements for repositories aimed at long term preservation of digital information objects, containing static objects (documents) and dynamic objects (programs)? It is recognized that preservation efforts should be independent of current technology in order to survive technology obsolescence. This requirement is hard to meet.
In this paper current preservation efforts (projects and techniques) and relevant standards are discussed in relation to this requirement. A view on authenticity of digital objects is presented that leads to the requirement of dependence on the designated community that is to be recognized in the design phase when building repositories.

Keywords: *longevity, preservation, standards, authenticity*

1. Introduction

A landmark in preservation was the publication in 1993 of the book 'Preserving the present' [1]. At that time, regarding the problem of preservation of digital information objects, records of knowledge and memory, neither relevant questions nor possible answers were known. 'Preserving the present', based on research on what organizations were doing then to preserve their electronic documents, was meant as a first guide to what they should do. The publication of that book was extremely useful to draw attention to the problem of digital preservation.
Electronic data on the US census of 1970 were no longer readable and proved to be lost beyond repair. The e-mails on the financial support of the State of the Netherlands to the shipbuilding industry were untraceable and probably deleted. The same for student allowances. Old electronic documents could not be reproduced in their original lay-out. The electronic Domesday books of 1986 was nearly inaccessible - quite a difference from the historic original of 1086.
Before publication of this book, only few people had realized that there was a relation between these phenomena. Preserving the present alerted people to the problems of preservation: the fact that reuse and readability of electronically recorded information is not guaranteed even in the near future. It is a subject in the heart of the science of information.

Ten years after 1993, it is a good moment to look at the state of the art on preservation by presenting design criteria for repositories. The topic of preservation repositories has become very important. The value of digital records has grown enormously. So has the amount of electronic information, as is suggested by Varian and Lyman [2]. Two categories of digital records exist: static and dynamic. Information provided by static objects is stable: it does not change over time. Traditional textual documents in digital form are static objects. Dynamic objects on the other hand contain (possibly machine-specific) instructions to be executed and may provide interactive user interfaces. Programs are dynamic objects, as are documents containing scripts or macros. A growing part of the information that has to be preserved is dynamic.
Moreover, the developments, changes and improvements on functionality in programs for electronic information objects take place at a rapid pace. Compared to the speed of the progress of the industrial revolution the speed of change in the electronic revolution is inconceivable. Astonishingly, the increase in the 'speed of write' (Harnad's term) does not by itself lead to durable thinking on preservation of electronic information objects.
The different aspects of ageing of rendering equipment, program libraries, operating systems, data carriers and hardware needs collaboration of experts from different fields of expertise. This need for collaboration makes the problem not easier to manage.

The problem of digital archiving (or preservation of digital objects in general) can be formulated in design criteria for repositories, as well as functional requirements to the preservation process once such repositories are realized. The repositories contain (static) documents and (dynamic) program derivates, software. The

repository and the preservation process should be independent of computing platform, media technology and format paradigms (stated by Dürr and Lourens in [3]) to the highest possible extent while providing adequate preservation of valuable information objects for as long as possible under heavy economic constraints. Thus, standards need to be developed, used and maintained, and general concepts for information value including selection and authenticity (evidential value) need to be defined. These design criteria are hard to meet.
In this paper examples are given of projects in the area of digital document preservation. The more complicated object class of programs is discussed, relevant standards are listed and a discussion on authenticity is presented. From these pieces of the puzzle a generalization follows and conclusions are drawn as to which (abstract) design criteria have to be met in creating repositories suitable for long term preservation of digital objects.

2. Document preservation

What are organizations doing now to construct a repository for 'until Doomsday or five years - whichever comes first' (Rothenberg)? Several examples exist in which a repository was realized where the design questions were in our opinion thoroughly considered and written down in a detailed way.

E-mail

E-mail messages can be created, received or maintained in the transaction of business or the conduct of affairs and, in that case, may have to be preserved as evidence. The need to preserve e-mails has made itself felt for several years. Fortunately, not all of the about 1000 million e-mails that are produced each year have to be preserved. The well-documented David project [4] reports that the old structure of electronic e-mail archive may appear disorderly due to the sheer quantities of files; this draws attention to the metadata necessary to access the e-mail archive. Apparently, in relation to policy on records management, the construction of a folder structure for an archive to be transferred and the assignment of useful file names is a point of attention. Finally, different governments have given different answers to the question whether paper copies or electronic copies of e-mails should be preserved in the archive. The attachments are a different kind of element; there are also differences to how to deal with the electronic attachments.

For use in Dutch government agencies, the Digital Preservation Testbed has designed and developed a solution to preserve e-mail [5]. This approach aims to provide a practical means to either automatically preserve e-mail when it is sent, or preserve received e-mails at any time. The approach embeds a component in MS Outlook that converts an e-mail message into XML. This XML document is passed to a Web service that formats the XML file into HTML and forwards the XML file to a repository for storage. The HTML is passed back at Outlook and ultimately forwarded to the SMTP server responsible for sending. In this way, outgoing e-mail is automatically stored in XML and centrally formatted using a standard style sheet. Upon sending the user is required to enter metadata that is stored with the object.
The storage of outgoing e-mail is straightforward. All parts of the SMTP message are represented in the XML file that is stored. For received e-mails, the parts are separated into elements. Attachments and possible HTML body content is saved as separate files to which references are included in the XML file. A logfile is also included, as is the original SMTP message (a textual dump of all fields, including header information and encoded binary attachments), in this approach called 'transmission file'. The Testbed approach is a step towards storing messages in a standardized manner, using strict regulations on accompanying metadata, required trace information (logfiles) and redundant inclusion of attachments (both in encoded form (in the transmission file) and in decoded form as saved binaries). Using XML as storage format, the message body part for non-HTML formatted mail is preserved according to the opinion that XML is a future-proof format for textual objects. Preservation of binary attachments is a problem, as these objects can either be static or dynamic. Emulation might be necessary, as will be discussed in the section on program preservation. HTML formatted body content is saved to file, thus making it susceptible to obsolescence. Conversion to XHTML+CSS would make it more durable (as HTML is in danger of becoming extinct and XHTML is an XML application), but requires an extra conversion step that might be done at a later stage.
Basically the approach boils down to a migration technique. Redundancy is used as a safety net: the original message is included in the archive. If the XML packaging technique becomes outdated or gruesome, it can all be done again in some different form.

For web archiving a similar design could be used. The differences would be the transmission file (now a textual dump of a HTTP response) and the composition of metadata, as other contextual information is relevant. Binary attachments can be considered to have the form of embedded content such as Flash movies

that require a viewer to be rendered in the future. On a functional level the approach can be copied from the one proposed by the Testbed for preservation of e-mail. Once again the HTML body content is a problem (and once again conversion to XHTML+CSS might be considered).

Nedlib

Nedlib was the project of libraries, computer science organizations and publishers to design and set up requirements for a deposit system for electronic publications. The Guidelines have been published in the Nedlib report series [6]. This project aimed at preservation of publications for national libraries. What proved to be the major issues in this at this state-of-the-art project? They proved to be the vocabulary (a list of terms was issued!), the applicable standards were the subject of a thorough investigation, the strategy of emulation for maintenance purposes, the use of the OAIS model (see the section on standards), the metadata and its relations to the OAIS model, and of course the realization of a long-term deposit system. Interestingly, the results lead to an operational Deposit system, of which the results have been published, allowing refinement of the original ideas [7].

Cedars

The Cedars project [8] was carried out in 1998-2002 to establish best practices for digital preservation for UK Universities. Like the Nedlib project it was well thought out, had sufficient mass, and was based on research rather than assumptions; it lead to fundamental insight on the practice of preservation. The parties in Cedars (universities) form collections. Collectioning means selection. Selection means that information objects can be excluded for reasons of content (outside scope) or other reasons. This could be stated in a Service Level Agreement (SLA) regarding the types of information objects to be kept in the collection. Selection provides the Cedars organizations with a 'degree of freedom' the Nedlib partners do not have: as deposit libraries, these have the duty to preserve all information objects that form the national intellectual heritage. The emphasis in Cedars was on managerial aspects; technicalities seem to be treated as rather subordinate. The Cedars way of working is based on the OAIS model. The considerations on collection management and costs are valuable. A demonstrator has been built.

E-archive

The e-archive project of Delft, Utrecht and Maastricht [9] can be seen as an extension to the Cedars project. Its aim is to realize a workbench of electronic publications for decades. Again, the OAIS model is adhered to. The publications are put in an XML container, containing a standard identification, the original bitstream, the necessary viewer, zero or more conversions of the original bitstream, and various kinds of metadata. In this project, the business model of the e-archive with two times appraisal, requirements on data management and access, and a cost model are worked out in detail.

Generalization

The list of projects described is meant to be extensive nor complete. This short summary suffices to illustrate the general direction in which these efforts are going: towards a standardized 'archive architecture' based on the OAIS model, incorporating XML applications (such as XHTML) when possible. The aim apparently is technology independence through standardization: a generally applicable architecture using a standardized format for archival content. As these projects are built on a foundation of standards, the choice of standards to incorporate is the crucial cornerstone and therefore the weak spot.

3. Program preservation

The problem of preserving dynamic objects is a subproblem of preserving many object types: for e-mail it is hidden in the attachments and for web pages it is included by scripts and embedded players (such as Flash and Shockwave). Documents containing scripts or macros can also be regarded as dynamic objects: advanced techniques used in wordprocessing can turn a document in a object that is very hard to preserve.

In archiving digital objects, programs are by far the most complicated ones. Preserving such a 'dynamic object' requires the preservation of the runtime environment in which it is to be executed. This environment is crucial to the 'rendering' of a dynamic object.
A problem with this requirement is that it tends to be recursive: to preserve the program, all underlying layers

(operating system and hardware) have to be preserved as well. An executable compiled to run under MS Windows on an Intel platform requires both components to be preserved if the executable is required to run in the future. These components cannot be replaced by others: the executable will contain platform-specific machine code and OS-specific function calls. Preserving one Windows machine to preserve all Windows programs will not work as programs designed for Windows XP will not run on Windows 95 and programs compiled for Windows NT on a DEC Alpha will not run on an Intel machine. The recursion can be drawn further: how about peripheral equipment, network, documentation and required skills? What if a user, other than an experienced computer scientist, is confronted with a thirty years old machine under emulation, which was even then operated by trained personnel?
Two types of programs need to be distinguished. Programs that are enablers to the rendering of data ('viewers') are different types of objects than interactive objects (games for instance). The difference can probably best be illustrated by the degree of dependance on a specific computing platform when 'rendering' the information contained in the object.
A PDF document for example is a static object containing data to be rendered. To do so, a specific computing platform is not required: just a program that can interpret the data correctly. This viewer is a dynamic object of the relatively undemanding viewer kind: creating an emulator to preserve it does not compare to the cost of rewriting the viewer altogether. The virtual machine approach can also be used for this class: as relatively undemanding programs, a simple computing platform can be designed for which emulators can be created at low cost and for which such viewers can be programmed. Once available, access to these viewers (and thus to the data they can render) can be provided by creating the simple emulator. In this way, the cost of emulation can be reduced drastically. The UVC approach discussed further on has a similar design.
To play a level of Quake, more is needed than a graphical image produced on screen: the playing experience needs to be replicated, including sound, video effects (possibly requiring specific video hardware), input devices and speed of game play. Rebuilding such a game is a gruesome operation that might easily compare to the complexity of emulation of the computing platform. To preserve highly interactive objects such as games, emulation is probably the only solution. Virtual machines are no option here: as these programs are very demanding, a virtual machine would have to be so complex that it compares well to an emulator for the original computing platform.

Emulation is an essential strategy in preserving dynamic objects. Even though the costs are high, emulation may be feasible if a large amount of programs running on a specific computing platform need to be preserved. Only a single emulator would be required. This emulator is an extremely complex program. The computing environment in which the game was originally run has to be replicated in such detail that the game can be played in the same way as it could one generation ago. One may question whether Pacman, the well-known old computer game, is fun to play on a modern machine with a 2 GHz CPU. One can state that playing against a figure that moves with the speed of light on your screen is not how the game was intended.
This technique is mostly applied for games. For many platforms no longer in existence (mainly home computers and game consoles) emulators are freely available, quite often created by gaming enthusiasts. The success of these emulators is often referred to as a suggestion that emulation is a feasible approach to preservation. This success is relative: although emulation of a game system that is designed entirely by a single manufacturer might be possible, emulation of current mainstream 'office systems' is quite a different story. The latter category exists of systems that incorporate hardware designed by a multitude of manufacturers in many different configurations.

The first to propose emulation as a preservation strategy was Jeff Rothenberg in 1999 [10]. The widely held discussion on the choice between emulation and migration following his landslide 'Quicksand' article has for a large part set the scene for the problem area of preservation. This discussion, also known as 'Rothenberg vs. Bearman' as David Bearman replied to the 'Quicksand' article with a now equally famous critique [11] seems to have ended in a tie: most researchers seem to feel that neither one approach is feasible to solve all problems. From a certain point of view, the difference between the two boils down to the difference in costs between computer power and storage capacity [9].
Migration and emulation can be seen as two dimensions of one plane. Every solution (a point in the plane) can be regarded as a combination of the two extremes of complete migration and complete emulation. If objects are migrated (converted) at regular intervals to keep up with technology, emulation is not necessary. On the other hand when a 'complete' emulator is build to provide an environment for the original viewer, migration is out of the picture. As migration has high variable costs (it has to be done for each object at regular intervals) and emulation is extremely costly in development and maintenance due to its complexity and has to be repeated for each legacy platform to be 'projected onto' each future platform, optimization by combination seems to be the best way to go.

Such a combination is proposed by Raymond Lorie [12]. His Universal Virtual Computer is for a large part based on emulation, and the entire approach ends in a migration step.
The idea is to design a small and very easy to implement computer. This computer is implemented on each future platform (at relatively low cost, due to its simple design). In this way, a rather inexpensive 'emulator' is provided to run UVC programs. By standardizing the UVC design, it is guaranteed (or expected) that UVC programs do not have to be changed (or recompiled as the case may be) in the future. The second step is to build a UVC program for each format to be supported in the archive. This program 'decodes' a format into a logical representation that can be understood by future users - a migration step. In the future viewers can be built to render this representation.
The UVC is currently being developed and will become operational in the electronic deposit as it is in development at the Royal Library of the Netherlands [13]. It is included in this project as a last resort: once document viewers can no longer provide access to legacy formats, the UVC approach will be used to provide long term access to images of document pages.

Source code

When discussing program preservation, two types of objects can be considered as input of the preservation process: compiled executables and source code. As it is (very) likely that only compiled programs are available to the repository, the most probable option to program preservation is emulation. If the source code is still available, one could argue that the expense of designing a verifiably correct emulator could be saved by re-engineering the program to run on a future computing platform. In simple terms: 'just' re-compile using a more current compiler for a more current platform. Attractive as this may sound, there are still a few complicating issues to deal with.
To start with, code is written in a specific programming language. Even though such languages tend to be standardized (the computer language C is the most obvious example: it is ISO standard 9899:1999), there are few guarantees that a program written for a specific runtime environment can be compiled without problems for another. Programming libraries providing access to platform specific features may differ significantly. Functionality on the level of the operating system may not be available in the same way if available at all. Imagine a program designed to run on a Windows environment that has to be compiled for a future UNIX-like environment. These systems differ significantly. Reconstructing (in software engineering called 'porting') the program is not a trivial task.
To allow for programs to be ported, the source code needs to be well documented and written in a language for which compilers will still be available in the future. If this is not the case, code may still be portable if the programming paradigm does not differ between the language the program was written in and the language to which it is to be ported.
Between language classes of the same paradigm code can be 'translated'. It requires a skilled programmer with expertise in both languages to assert the validity of the translation. The effort of translating code to another language class (for example from a logical language like Prolog to a functional language such as Miranda or to an object oriented language like C++) equals or exceeds that of redesigning the complete program.

These drawbacks illustrate the complexity of reconstructing software, but in some cases this approach may be preferable to emulation. The execution speed and the possible integration of the reconstructed program in existing systems are the most obvious. The end-user will be provided with a program suitable to execute on a current platform and will require no or little additional tools to do so. Problems regarding peripheral devices and user interfaces are dealt with adequately: instead of having to work with ancient text-based interfaces, the user is provided with the modern graphical interface he/she is more used to. Even though the effort required might be comparable to emulation or re-engineering, reconstruction of software might be the preferable preservation technique in situations where a large user community is planning on using the program frequently for years to come.
As this technique requires specific (possibly legacy) programming expertise, this is not a task suitable to be accomplished by repositories. It might even be argued that it is not a preservation technique at all, as the information object (the program) is altered drastically. Yet it is a way to provide access to information structures (such as databases) on abandoned platforms that might otherwise be lost forever. ThereforAn example of a restoration of a program was the restoration of E-plot [3]. The restoration of this program was necessary as its results are used for a widely used reference model. The program was originally written in Fortran and C (to run on an IBM-RT using AIX as operating system) and was dependent on specific source code libraries in use at the time of development. In the article 'programs for ever' the authors describe in detail the complexity of reviving software no longer maintained and stress the importance of preservation of scientific software to allow for preservation of scientific data sets. It proves to be possible to reconstruct old

software to execute on a more modern platform. Again, the use of OAIS AIP's proved to be applicable. The result is in a way medium independent and platform independent.

Generalization

Programs are designed to be executed in a specific runtime environment. Unlike 'static' documents that are nothing more than chunks of data independent of computing platform (as they do not contain machine specific instructions), the functionality of programs is dependent on machine specific parameters. Technology independence is hard to achieve when objects are designed to be technology dependent. Standardization is no longer the remedy of choice. For existing platforms, combinations of hardware and software, these runtime environments cannot be standardized as this would result in 'freezing' technology and disallowing innovation. For abstract platforms standardization is possible. This is the approach used by virtual machines such as the UVC: technology independence by introducing a standardized abstract machine that is to be emulated on existing platforms.
As there are several ways in which digital objects can be used, different preservation strategies are applicable to different types of objects. Even though emulation and migration can be applied to every object type, feasibility and costs are the determining factors in choosing strategies. It is possible to migrate an executable to another platform (by 'translating' the instruction stream), but the costs may be higher than building a general emulator. Emulating a platform to run a viewer for an old format version of software still in use is more costly than allowing for the current software to convert old formats.
Program preservation is a problem that can only be tackled by emulation or reconstruction, due to the nature of programs as instruction streams. The complexity of the emulation solution can be reduced by using virtual machines: this solution is however only feasible for relatively simple programs (of the 'viewer' type) that have to be compiled especially for the virtual machine at hand. To allow access to existing legacy software of a more demanding nature (games), or for which the reconstruction for a modern platform or redesigning/recompiling for a virtual machine is not feasible (i.e. cheaper than building an emulator), 'pure' emulation of legacy platforms is the only possible way to (re)gain access in the future.

4. Standards

Reuse of information objects demands agreement on all aspects of the information objects themselves as well as anticipation on the possible uses of the information objects. These agreements have partly been put down in standards. Partly, because standards have advantages (enhancement of the usage of common tools, enabling the reuse of experts experience) but also disadvantages (they deprive a user of some freedom to optimize a solution to his/her preference, and they take time of qualified staff). In order to discuss design desiderata of a durable repository of information objects, an inventory of standards is presented. Standards have been designed mostly for reuse of information objects independent of distances. Everyone should (under conditions) be able to reuse them.

XML and relations

The information objects are often structured according to the Extensible Markup Language, XML and it relations. Occasionally, domain specific derivatives are found, like MathML, WAP (wireless), XLS (location-based services). Data type specific derivatives include SVG (Vector Graphics) and SMIL for streaming media. Relations are xmlns (namespaces), the Resource Description Framework RDF for content specification. Moreover, XML is the basis for the lay-out structure by the Extensible Stylesheet Language XSL (more precise: XSL Transformations XSLT and the navigation mechanism XPath); other members of this family need not to be mentioned here. The popularity of XML with its derivatives is very impressive.
The great news of XML is that it is self-descriptive, a valuable property for preservation. If in the future a part of an electronic object is found without head or tail, and it contains structures like <Tag>Value</Tag> (to be recognized at byte level), then it is XML or at least HTML. From the name of the tag (when standardized or chosen carefully) the meaning of the tag content can be deduced and the value can be interpreted correctly. This way, a structure and a part of the semantics present themselves. Structures with attributes like <Tag Attribute="AttrValue">Value</Tag> can be interpreted in the same way.
The bad news about XML is that its longevity is not ensured. XML itself is the successor to SGML, ISO standard 8879:1986, its relation XSL is derived from DSSSL, ISO standard 10179:1996, the companion to SGML and XML is a successor to ODA, ISO standard 8613:1986. SGML and XML are not fully compatible. The future of SGML looked bright once, just like that of XML does now. XML is known to have drawbacks. An example: XML files are big and clumsy for location based services. Will there be a successor to XML,

named Enhanced XML - Improved Technology! (EXIT!); and if so, what shall be the future of XML files?

Presentation

For presentation PDF is often used. PDF is not an open standard; it is owned by Adobe. That makes this standard vulnerable for economic incidents. An initiative has been reported by Boudrez et al. in which it is tried to realize a PDF subset for archiving: PDF/A. In PDF/A the targets are as autonomously as possible. External dependencies as encryption, compression methods (that could be proprietary), copyrighted character sets, references to external files, encapsulation of executables etc. are being avoided. The alternative to PDF is the XML partner XSL; occasionally HTML and CSS (Cascading Stylesheets, a companion to (X)HTML) are mentioned. Both XML and PDF are often mentioned as acceptable formats to deliver information objects to the end-user: the output of the preservation process.

OAIS, Open Archives Information System, ISO standard 14721:2003

The OAIS model is a reference model for a system for archiving information, both digital and physical, with an organizational scheme composed of people with the responsibility to preserve information and make it available to a designated community. Firstly, it describes at a high level the processing of information objects. The acceptance procedure, called ingest, describes the processing of Submission Information Packages (SIPs). Also it enables the process of keeping and preserving Archival Information Packages (AIPs), and the delivery to the end-user of Dissemination Information Packages (DIPs). The OAIS model enables to define task structures for the electronic archive in the form of workflow processes. Secondly, the OAIS model contains an anticipation to the future users of the information objects. It is being presented under the term of 'designated communities'. A description of the designated communities enables to state what information objects will have to be kept, and what quality conditions apply.

US DoD 5015-2, MoReq and ReMaNo

They are meant for software specifications for record management applications. The US DoD (Department of Defense) 5015-2 Standard is a set of requirements. It is well known and proves to be in accordance to electronic records management. MoReq, MOdel REQuirements for the management of electronic records, is its up-to-date EC equivalent; ReMaNo (Softwarespecificaties voor Records Management Applicatie voor de Nederlandse Overheid) aims at the same goal but is based upon the Dutch law of Archives. These standards define aspects like control and security, acceptance, folder structure, retrieval, appraisal, selection, retain time, transport, destruction, access and presentation, administrative functions and performance requirements.

Records management, ISO standard 15489:2001

The ISO standard on Records Management is the successor to the Australian AS 4390 standard. In a way, it is a well established standard: many people have expressed ideas about records management, and applied the idea that if the costs to keep records exceed the damage if the records have been disposed of, is a basis principle for records management. The standard addresses policy and responsibilities defined and assigned throughout the organization as well as the records management requirements authenticity, reliability, integrity and usability.

Retrieval languages: OAI-PMH and ANSI Z39.50

In distributed systems, in order to find preserved objects, all kinds of query systems can be used. When several collections are coupled or when multiple copies of objects are stored at different locations (the LOCKSS principle - Lots Of Copies Keep Stuff Save), a mechanism is needed to retrieve information about collection contents in order to search for objects. In the Internet, a well known technique is harvesting: retrieving information by having an automated process retrieve information from data publishers at regular intervals. A result of the Open Archives Initiative (OAI) was the building of the Protocol for Metadata Harvesting (PMH). An archive willing to disseminate their content through the web can open up its electronic archive for harvesters. A harvester of a service provider contacts the archive and retrieves records containing metadata about the objects archived. The service provider offers indexes and retrieval facilities based on these records to end-users. The OAI-PMH does not demand much expertise, less than the older well-known and more powerful ANSI Z39.50 protocol (and the corresponding ISO standard 23950:1998) that has been in use for over a decade.

Data carriers

Information objects have to be saved on 'data carriers' that can be read on all kinds of equipment. Quite a few standards have been established. As an example, the ISO working party of optical disk cartridges gives a list of 32 standards [14]. That, at least, is a witness to the aim for interoperability.

The article 'Overview of technological approaches to digital preservation and challenges in coming years' by Thibodeau [15] is an excellent overview on digital preservation.
However, his article seems to treat ICT standards as fixed entities, as boundary conditions. ICT and its consequences are rather more a variable than a fixed entity. The design of any system means balancing between needs and wants of users, technical possibilities, changes, disadvantages and risks. Also, forecasts on technical possibilities are often inaccurate. The expectations on Information retrieval of the general public and even some experts in the 1980's and 1990's serve as an example. Computers would make it possible to store all documents. It was expected that, once all documents would be stored electronically, full text retrieval would make it possible to find all known information. A complete mistake: the Stairs experiment [20] was the first to shed doubt on the expectation; in 1998 came Schwartz's sigh [17]: improvement on general-domain web search engines may no longer be possible or worth the effort!
IT aspects influence the design desiderata so pervasively that it cannot be 'sorted out' (Thibodeau) and must remain at the heart of the design desiderata.

Generalization

Standards enhance reuse of information objects independent of design environment. But reuse independent of time leads to a different view.
The nice thing of standards is, there are so many ones to choose from (a quote generally ascribed to Tanenbaum). However, from a longevity point of view there is not much choice. The long use of XML is disputable, as it may not live very long. In fact, most standards are blind to the teeth of time. The OAIS model generally adhered to is an exception. It demands to think of future users, although its guidelines are superficial. The standards on software specifications reflect the legal differences between nations on laws on archives. The standard on records management may be the best thing that ever happened to archives, but not all record creators use it well. Still that is essential for a costly repository. Many creators do not know the standard, let alone its consequences. The state of retrieval languages shows one more reinvention of the wheel: although OAI-PMH may be made compatible with Z39.50, it was not created as such. Chances are that enormous investments of libraries and archives and other memory institutions in Z39.50 may eventually be discarded. In the list of standards on data carriers at least relations between types of standards have been inserted, but it looks like the tower of Babel.
For longevity purposes, standards should be built and maintained as long-lived artifacts. One could draw design desiderata for long-lived standards, like: standards should not be too complex, too large and too 'fat'. For standards small is not only beautiful but probably also lasting: the motto 'less is more' certainly applies to standards. This kind of desideratum needs further research.

5. Authenticity

Authenticity of digital objects is probably the most debated preservation requirement. Obviously every object that 'comes out of storage' should preferably be authentic, 'real' and 'trustworthy'. As every computing application imposes different requirements on the objects it requires, authenticity in its broadest sense could be defined for each and every application differently. A digital repository designed to preserve objects of any kind requires a general notion of authenticity or at least an objective means to measure the result of the preservation efforts against the applicability or usability of objects once they are delivered after years of storage.

This research borrows two fundamental concepts from other disciplines. Firstly, the context-dependent interpretation of the 'copy' concept put forward by Paskin in relation to digital rights management [18]. He suggests that two digital objects are only to be considered identical within the same context (i.e. when used for the same purpose). The context of use is the determining factor in establishing the correctness of the copy, the 'sameness'. Properties of the object not of relevance for the purpose to be served are not necessarily copied. This interpretation of the 'copy' concept matches with its use in everyday life: an encoding of digital audio (in MP3 for example) is clearly a 'copy' of a copyrighted work used to serve the purpose of playing music at a reasonable level of audio quality. It is not a copy in the context of CD manufacturing, as in that

context the lost property of binary integrity is relevant. The concepts 'copy' and 'original' only have meaning in a particular context of use: in that context the original is obviously the input of the transformation (copy) process and the copy is the output. This is intuitive: a digital object cannot be a context-independent, 'absolute' original. The original information (the first manifestation of the information) is always lost: whether it is the performance of which the CD is the recording or the document typed in a word processor of which a copy was saved from memory to disk. Only information relevant for the object use is recorded or saved: not the expression on the artists face or the typing rate of the author. Note that a clear definition of the context replaces any physical or logical requirement to be imposed on the copy to assess its quality.
Secondly, from cryptography, it is recognized that messages sent between parties are considered 'secure' if their integrity, authenticity and confidentiality can be established and the procedures used are tamper-free (the requirement of 'non-repudiation'). In this application, authenticity means the requirement that the origin of messages can uniquely be established. This requirement of identification in this context suffices to establish authenticity.

These two building blocks provide all the concepts needed to build a conceptual framework to deal with authenticity.
The terminology used in the literature suggests which properties are relevant: what constitutes authenticity. Dollar for example states "authentic records are records that retain their reliability over time" [19]. The term 'reliability' refers to the authority and trustworthiness of records: they "stand for the facts they are about". Bearman and Trant suggest that authenticity consists of three 'provable' claims: the object is unaltered, it is what it purports to be and its representation is transparent [20].
Using the concepts borrowed from cryptography, relevant requirements are object integrity and identification. The requirement of non-repudiation is implied: Dollar's 'authority' and the 'transparent representation' mentioned by Bearman and Trant indicate the requirement of verifiably tamper-free preservation procedures. The fourth element in cryptography does not seem to be applicable: confidentiality of information conflicts with the purpose of preserving information for the public.
The requirements of 'trustworthiness' and 'authority' can be considered to be combinations of integrity and identification. If any of these two fails, an object is clearly not 'trustworthy'. An additional requirement is needed to assert whether an object can actually replace the original object in the process in which the original was used. This is the intrinsic value of the object: it always serves some purpose and if it no longer can do so it loses its value (and thus the reason to be preserved).
This requirement is taken to be 'authenticity': for a specific (identified) purpose, an authentic object achieves this purpose at least equally well as did the original object. More formally: within a certain context, an authentic object is a verifiably correct implementation of the functional requirements relevant in that context imposed on the original object. This context is the designated community from the OAIS model.

Complex issues regarding authenticity can now be answered. The answers might be surprising at first glance, but are logical expansions of the intuitive notion of authenticity. Two examples are given.
A legacy program, accompanied by a database, is preserved by an repository. The program contains the 'millennium bug' causing it to yield incorrect answers to queries. The repository has preserved the program bit stream flawlessly and is even able to provide a verifiably correct platform emulator (an achievement only possible in theory). Executing the program in 2003 correctly yields the incorrect results. The question arises which object would be the authentic one: the preserved bit stream or a debugged and thus altered copy (with the purpose of execution under emulation)? What purpose does the program bit stream serve if its execution is not without failure? If some researcher wishes to examine the program as it was run decades ago, this bit stream is the authentic one. In the more likely situation that the object is to be executed in order to obtain the correct answers to queries, the altered object is the authentic one.
The Night Watch by Rembrandt, one of the most famous paintings in the Dutch cultural heritage, is in its current form not even close to authentic. During its 360 years of existence, a part has been cut off, it has been 'knifed' by a museum visitor and it has been cleaned. It clearly fails to meet requirements of object integrity and the preservation process does not meet requirements of non-repudiation (as it allows the object to be damaged and altered). Yet thousands of museum visitors from all over the world flock to the Rijksmuseum to see 'the real thing'. For them, there is no question about its authenticity. For the purpose of looking at a painting by Rembrandt, the object stored serves this purpose at least equally well as the original object (in this case the same) did 360 years ago. For this purpose, authenticity is derived from identification: if it is the picture that Rembrandt painted, it is authentic. Any derivative (photo, sketch, drawing) is not. If Rembrandt had painted the picture twice, the second one would have been authentic for the purpose of attracting museum visitors, but not for the purpose of studying the cloth used in the first version.

These examples illustrate that preserving original bit streams and building computing museums do not

provide solutions to all problems regarding object authenticity. Authenticity is not the same as integrity, identification or originality. Terms as 'trustworthiness' and 'reliability' (a term broader than reliability in computing architectures) are too subjective to allow for practical assessments. The reason why in cryptography the terms 'authenticity' and 'identification' are interchangeable is that in those systems the purpose of the messages sent is achieved 'just' by identification of their origin.

Preserving digital objects to keep them 'available', i.e. to allow for future use of the object, imposes functional requirements on the repository. As authenticity is context dependent, the context in which the object is to be used in the future needs to be described in as much detail as possible. This context allows for the identification of what features of the object, which functionalities, need to be preserved. If only the text of newspaper articles need to be preserved (future users will only need to read the information contained and search for strings), it suffices to store text files in Unicode, which is cheaper and less difficult than storing the articles as PDF (for example). If the requirement allowing for textual search is dropped but graphical lay-out is to be provided, optical scans stored in BMP could be stored. To allow for both, both can be stored.
Reducing authenticity to a set of functional requirements seems to be an obvious, somewhat belittling approach as one is tempted to store the object as it is today and engage in all kinds of difficult technical approaches to keep it accessible, convinced that the original object will always be the authentic one. As stated earlier, no object can be authentic for each and every unforeseeable future purpose.

The most important consequence of the concept of authenticity as a context-dependent aspect of objects in storage is that it can (and should) be made explicit as a set of functional requirements that are negotiated upfront, prior to storage. A result of this negotiation would be a service level agreement (SLA) of sorts: a document serving as a contract, exactly describing what preservation efforts are to be expected from the repository and, partly as a result of these, what functionality can be expected of stored objects once they are delivered in the future. Such a 'preservation effort agreement' (PEA) can be the basis of quality assessment after delivery and, in its quality as a contract, a basis to solve disputes once objects do not meet requirements. Such negotiation upfront solves a lot of issues regarding vague and subjective (and therefore unquantifiable) requirements of 'authenticity'. A list of functionalities and quality indicators, the PEA is unambiguous. Furthermore, it connects well to the SLA which has been part of system development and maintenance for years. As digital preservation may itself be part of a larger information system, the PEA could prove to be a valuable quality indicator as part of a larger SLA.

Another problem with storing originals is that no file format lives forever. Preservation techniques might change the object to keep the information it contains available (migration) or provide access in a possibly reduced form by providing a virtual computing environment (emulation). Neither technique can provide warranties that an object stored today can function in the exact same way in the future: probably something, some functionality, will be lost. It seems to be logical to assure oneself that the functionalities crucial to the object's use within a certain context are not among the functionalities in danger of getting lost: hence the formal specification of functional requirements upfront. If these requirements are not made explicit before collections of objects are ingested, design choices in preservation techniques or restrictions on migration possibilities might cause irreparable restrictions for future use. They might even render objects entirely useless for their designated communities.

An example taken from a current preservation project for e-mail proves this point. The strategy adopted was to convert e-mail messages in textual form to XML. In the specific case of 'raw' textual messages this can be done rather easily as the fields used in the SMTP protocol are fixed in amount and the structure of an SMTP message is very suitable to be captured in XML. As it turned out, the conversion process did not allow for so-called HTML-mail: messages with an HTML document as body. Style elements were lost as the body was reduced to its textual content. This is a restriction of functionalities that might be relevant for the future user. Implied by design choices made for preservation strategies, in a worst-case scenario these invisible restrictions would only noticed after years of preservation, when it is too late for repair.

The weak spot in reducing preservation efforts to a set of functional requirements is the necessity to identify the 'designated community' and, more importantly, identify its needs. It is impossible to know upfront what future users will expect from archived objects and how they will use the objects. This is a problem that obviously cannot be solved before the invention of time travelling. As one cannot give more than one has, regarding object quality one can only store objects at the quality they are now. If that quality is reasonable for us, it will (have to) be enough for any future user. Guarantees on authenticity and quality of preservation can only be given by explicitly formulating what constitutes that authenticity and quality for a particular object in a particular context at the time of ingest.

Generalization

Authenticity is clearly a central issue in preservation. On the one hand it defines the quality of the preservation efforts of the repository and on the other hand it defines the objects usability or applicability for the user. As it is the user who will assess both, it is imperative to include the intentions of that user in the authenticity requirement. Practically speaking, the authenticity requirement needs to be regarded in the context of the objects purpose and use. Catched in a catchphrase: 'authenticity is nothing without purpose'.
The design criterium that results from the presented view on authenticity is that of goal dependence. Where preservation, as stated, should be independent of technology, it should be dependent of the designated community, in OAIS terms. This means that the intended future object use should be considered when designing a repository. Illustrated in the previous section, this requirement cannot simply be ignored. If the designated community is not taken into account, stored objects have to be authentic for everyone and every purpose. As shown, this is an unrealistic requirement.

7. Conclusions

Digital information objects in digital repositories should last until Doomsday or until they are no longer useful - whichever comes first. This means that preservation efforts have to be technology independent in order to survive technology obsolescence. This technology independence can partly be realized by standardization: adhering to the OAIS model and choosing XML as intermediate file format are design choices common to most preservation projects in current development.
For dynamic objects such as programs or documents containing 'active content', standardization is only partly applicable. As these objects contain instructions to be executed within a particular runtime environment, this environment needs to be preserved or recreated in order to preserve the object. Technology independence is hard to achieve here, and can only be realized when using virtual machines to provide the runtime environment. This approach is only feasible to preserve dynamic objects that are logically independent of specific hardware (such as viewers). For other dynamic objects (such as games) or legacy software for which reconstruction or recompilation is too costly or impossible, emulation is the only possible solution: temporary technology independence by projecting one computing platform onto another.
Whether emulation or migration will prove to be the most successful remains to be seen. Most likely, every preservation problem for every digital repository will have the choice on the degree to which they will be combined.
Using standardization to achieve technology independence does not result in time independence. Unfortunately, ICT standards do not seem to last and chances are that the standard of choice today will be abandoned tomorrow. When designing preservation repositories using standards as a cornerstone, it is imperative to recognize this weak spot.
Authenticity of digital objects is determined by object purpose. Asserting the authenticity of stored objects requires taking the designated community, the future user, into account. As authenticity determines the value of objects stored and authenticity is dependent of the objects use and purpose, 'purpose dependence' should be taken into account when designing repositories. This dependence could be made explicit by using a Preservation Effort Agreement that serves as a contract containing functional requirements the stored objects have to meet after years of storage.
In order to build repositories that are and will remain useful, technology independence has to be achieved. To allow for 'purpose dependence', clear and well documented functional requirements have to be defined prior to long term storage.

References

All URL's are checked and valid in September 2003.

1. T.K. Bikson and E.J. Frinking: Preserving the present / het heden onthouden. SDU, The Hague, 1993.
2. H. Varian and P. Lyman: how-much-information?
3. http://www.sims.berkeley.edu/research/projects/how-much-info/ (September 2003)
4. E. Dürr and W. Lourens: Programs for ever. In: P. Isaías: Proceedings on NDDL 2002, Ciudad Real, 2002. pp. 63-79.
5. Digitaal Archief Vlaamse Instellingen en Diensten, DAVID.
6. http://www.dma.be/david/
7. ICTU: Bewaren van email. 2003

8. http://www.digitaleduurzaamheid.nl/bibliotheek/docs/bewaren_van_email.pdf
9. J. Steenbakkers: The Nedlib Guidelines. Nedlib report series, 5. Koninklijke Bibliotheek, Nedlib Consortium, 2000.
10. R.J. van Diessen en J.F. Steenbakkers: The long-term preservation study of the DNEP project. IBM/KB Long-term Preservation Study Report Series 1. IBM / Koninklijke Bibliotheek, 2002.
11. Cedars, Curl Exemplars in Digital Archives.
12. http://www.leeds.ac.uk/cedars/
13. R. Dekker, E.H. Dürr, M. Slabbertje and K. van der Meer: An electronic archive for academic communities. In: P. Isaías: Proceedings on NDDL 2002, Ciudad Real, 2002. pp. 1-12.
14. J. Rothenberg: Avoiding technological quicksand. CLIR report 77. 1999.
15. D. Bearman: Reality and chimeras in the preservation of electronic records. D-Lib magazine, April 1999.
16. R.A. Lorie: Long term preservation of digital information. ACM/IEEE Joint Conference on Digital Libraries, 2001.
17. http://www.informatik.uni-trier.de/%7Eley/db/conf/jcdl/jcdl2001.html
18. R. Lorie: The UVC: a method for preserving digital documents. IBM/KB Long-term Preservation Study Report Series 4. IBM / Koninklijke Bibliotheek, 2002.
19. ISO: Standards and guides on JTC 1 / SC 23:
20. http://www.iso.ch/iso/en/stdsdevelopment/tc/tclist/TechnicalCommitteeStandardsListPage.TechnicalCor
21. K. Thibodeau: Overview of technological approaches to digital preservation and challenges in coming years.
22. http://www.clir.org/pubs/reports/pub107/thibodeau.html
23. D.C. Blair en M.E. Maron: An evaluation of retrieval effectiveness for a full-text document-retrieval system. Commun. of the ACM 28 (1985), 289-299; D.C. Blair: Full-text retrieval: evaluation and implication. Int. Class. 13 (1986), 18-23; D.C. Blair en M.E. Maron: Full-text information retrieval: further analysis and clarification. Info. Proc. Mgmt. 26, (1990), 437-447.
24. C. Schwartz: Web search engines. J. Am. Soc. Info. Sci. 49 (11), (1998), 973-982.
25. N. Paskin: On making and identifying a "copy". D-Lib magazine, January 2003.
26. C.M. Dollar: Authentic electronic records. Cohasset Associates, Chicago. 2002.
27. D. Bearman and J. Trant: Authenticity of digital resources. D-Lib magazine, June 1998.