Overview of Technological Approaches to Digital Preservation and Challenges in Coming Years[1]

Kenneth Thibodeau

What Does It Mean to Preserve Digital Objects?

The preservation of digital objects involves a variety of challenges, including policy questions, institutional roles and relationships, legal issues, intellectual property rights, and metadata. But behind or perhaps beneath such issues, there are substantial challenges at the empirical level. What does it mean to preserve digital objects? What purposes are served by preserving them? What are the real possibilities for successful preservation? What are the problems encountered in trying to exploit these possibilities? Can we articulate a framework or an overall architecture for digital preservation that allows us to discriminate and select possibilities?

To address any of these challenges, we must first answer the simple question: What are digital objects? We could try to answer this question by examining the types of digital objects that have been and are being created. Many types of digital information can and do exist in other forms. In fact, many types of digital information are rather straightforward transcriptions of traditional documents, such as books, reports, correspondence, and lists. Other types of digital information are variations of traditional forms. But many forms of digital information cannot be expressed in traditional hard-copy or analog media; for example, interactive Web pages, geographic information systems, and virtual reality models. One benefit of an extensive review of the variety of types of digital information is that it forces one to come to grips with this variety, which is growing both in terms of the number of types of digital objects and in terms of their complexity. In fact, the diversity of digital information exists not only among types but also within types. Consider one application class, documents. There is no single definition or model of a digital document that would be valid in all cases. Information technologists model digital documents in very different ways: a digital document can be a sequence of expressions in natural language characters or a sequence of scanned page images, a directed graph whose nodes are pages, what appears in a Web page, and so on. How documents are managed, and therefore how they are preserved, depend on the model that is applied.

The variety and complexity of digital information objects engender a basic

criterion for evaluating possible digital preservation methods, namely, they must address this variety and complexity. Does that necessarily mean that we must preserve the variety and complexity? It is tempting to respond that the variety and complexity must indeed be preserved because if we change the characteristics of digital objects we are obviously not preserving them. However, that response is simplistic. For example, in support of the argument that emulation is the best method for digital preservation—because it allows us to keep digital objects in their original digital formats—the example of the periodic table of the elements has been offered. The information conveyed by the periodic table depends on the spatial layout of the data contained in it. The layout can be corrupted or obliterated by using the wrong software, or even by changing the font. However, to argue that any software or digital format is necessary to preserve the periodic table is patently absurd. The periodic table was created a century before computers, and it has survived very well in analog form. Thus we cannot say without qualification that the variety and complexity of digital objects must always be preserved. In cases such as that of the periodic table, it is the essential character of the information object, not the way it happens to be encoded digitally, that must be preserved. For objects such as the periodic table, one essential characteristic is the arrangement of the content in a 2-by-2 grid. As long as we preserve that structure, we can use a variety of digital fonts and type sizes, or no fonts at all—as in the case of ASCII or a page-image format.

We can generalize this insight and assert that the preservation of a digital information object does not necessarily entail maintaining all of its digital attributes. In fact, it is common to change digital attributes substantially to ensure that the essential attributes of an information object are preserved when the object is transmitted to different platforms. For example, to ensure that written documents retain their original appearance, authors translate them from the word processing format in which they were created to Adobe's PDF format. Fundamentally, the transmission of information objects across technological boundaries—such as platforms, operating systems, and applications—is the same, whether the boundaries exist in space or time.

Are there basic or generic properties that are true of all digital objects? From a survey of types such as those just described, one could derive an intensive definition of digital objects: a digital object is an information object, of any type of information or any format, that is expressed in digital form. That definition may appear too generic to be of any use in addressing the challenge of digital preservation. But if we examine what it means for information to be expressed in digital form, we quickly come to recognize a basic characteristic of digital objects that has important consequences for their preservation. All digital objects are entities with multiple inheritance; that is, the properties of any digital object are inherited from three classes.

Every digital object is a physical object, a logical object, and a conceptual object, and its properties at each of those levels can be significantly different. A *physical* object is simply an inscription of signs on some physical medium. A *logical* object is an object that is recognized and processed by software. The *conceptual* object is the object as it is recognized and understood by a person, or in some cases recognized and processed by a computer application capable of executing business transactions.

Physical Objects: Signs Inscribed on a Medium

As a physical object, a digital object is simply an inscription of signs on a medium. Conventions define the interface between a system of signs, that is, a way of representing data, and the physical medium suitable for storing binary inscriptions. Those conventions vary with the physical medium: there are obvious physical differences between recording on magnetic disks and on optical disks. The conventions for recording digital data also vary within media types; for example, data can be recorded on magnetic tape with different densities, different block sizes, and a different orientation with respect to the length and width of the tape.

Basically, the physical level deals with physical files that are identified and managed by some storage system. The physical inscription is independent of the meaning of the inscribed bits. At the level of physical storage, the computer system does not know what the bits mean, that is, whether they comprise a natural language document, a photograph, or anything else. Physical inscription does not entail morphology, syntax, or semantics.

Concern for physical preservation often focuses on the fact that digital media are not durable over long periods of time (Task Force 1996). This problem can be addressed through copying digital information to new media, but that "solution" entails another type of problem: media refreshment or migration adds to the cost of digital preservation. However, this additional cost element may in fact reduce total costs. Thanks to the continuing operation of Moore's law, digital storage densities increase while costs decrease. So, repeated copying of digital data to new media over time reduces per-unit costs. Historically, storage densities have doubled and costs decreased by half on a scale of approximately two years. At this rate, media migration can yield a net reduction, not an increase, in operational costs: twice the volume of data can be stored for half the cost (Moore et al. 2000). In this context, the durability of the medium is only one variable in the cost equation: the medium needs to be reliable only for the length of time that it is economically advantageous to keep the data on it. For example, if the medium is reliable for only three years, but storage costs can be reduced by 50 percent at the end of two years, then the medium is sufficiently durable in a preservation strategy that takes

advantage of the decreasing costs by replacing media after two years.

The physical preservation strategy must also include a reliable method for maintaining data integrity in storage and in any change to storage, including any updating of the storage system, moving data from inactive storage to a server or from a server to a client system, or delivering information to a customer over the Internet, as well as in any media migration or media refreshment.

Obviously, we have to preserve digital objects as physical inscriptions, but that is insufficient.

Logical Objects: Processable Units

A digital information object is a logical object according to the logic of some application software. The rules that govern the logical object are independent of how the data are written on a physical medium. Whereas, at the storage level, the bits are insignificant (i.e., their interpretation is not defined), at the logical level the grammar is independent of physical inscription. Once data are read into memory, the type of medium and the way the data were inscribed on the medium are of no consequence. The rules that apply at the logical level determine how information is encoded in bits and how different encodings are translated to other formats; notably, how the input stream is transformed into the system's memory and output for presentation.

A logical object is a unit recognized by some application software. This recognition is typically based on data type. A set of rules for digitally representing information defines a data type. A data type can be primitive, such as ASCII or integer numbers, or it can be composite—that is, a data type composed of other data types that themselves might be composite. The so-called "native formats" produced by desktop application software are composite data types that include ASCII and special codes related to the type of information objects the software produces; for example, font, indentation, and style codes for word processing files. A string of data that all conform to the same data type is a logical object. However, the converse is not necessarily true: logical objects may be composite, i.e., they may contain other logical objects.

The logical string must be stored in a physical object. It may be congruent with a physical object—for example, a word processing document may be stored as a single physical file that contains nothing but that document—but this is not necessarily the case. Composite logical objects are an obvious exception, but there are other exceptions as well. A large word processing document can be divided into subdocuments, with each

subdocument, and another object that defines how the subdocuments should be combined, stored as separate physical files. For storage efficiency, many logical objects may be combined in a large physical file, such as a UNIX TAR file. Furthermore, the mapping of logical to physical objects can be changed with no significance at the logical level. Logical objects that had been stored as units within a composite logical object can be extracted and stored separately as distinct physical files, with only a link to those files remaining in the composite object. The way they are stored is irrelevant at the logical level, as long as the contained objects are in the appropriate places when the information is output. This requires that every logical object have its own persistent identifier, and that the location or locations where each object is stored be specified. More important, to preserve digital information as logical objects, we have to know the requirements for correct processing of each object's data type and what software can perform correct processing.

Conceptual Objects: What We Deal with in the Real World

The conceptual object is the object we deal with in the real world: it is an entity we would recognize as a meaningful unit of information, such as a book, a contract, a map, or a photograph. In the digital realm, a conceptual object may also be one recognized by a business application, that is, a computer application that executes business transactions. For example, when you withdraw money from an ATM machine, you conceive of the transaction as an event that puts money in your hands and simultaneously reduces the balance of your bank account by an equal amount. For this transaction to occur, the bank's system that tracks your account also needs to recognize the withdrawal, because there is no human involved at that end. We could say that in such cases the business application is the surrogate or agent for the persons involved in the business transaction.

The properties of conceptual objects are those that are significant in the real world. A cash withdrawal has an account, an account owner, an amount, a date, and a bank. A report has an author, a title, an intended audience, and a defined subject and scope. A contract has provisions, contracting parties, and an effective date. The content and structure of a conceptual object must be contained somehow in the logical object or objects that represent that object in digital form. However, the same conceptual content can be represented in very different digital encodings, and the conceptual structure may differ substantially from the structure of the logical object. The content of a document, for example, may be encoded digitally as a page image or in a character-oriented word processing document. The conceptual structure of a report—e.g., title, author, date, introduction—may be reflected only in digital codes indicating differences in presentation features such as type size or underscoring, or

they could be matched by markup tags that correspond to each of these elements. The term "unstructured data" is often used to characterize digital objects that do not contain defined structural codes or marks or that have structural indicators that do not correspond to the structure of the conceptual object.

Consider this paper. What you see is the conceptual object. Then consider the two images below. Each displays the hexadecimal values of the bytes that encode the beginning of the document.[2] Neither looks like the conceptual object (the "real" document). Neither is the exact equivalent of the conceptual document. Both contain the title of the article, but otherwise they differ substantially. Thus, they are two different logical representations of the same conceptual object.

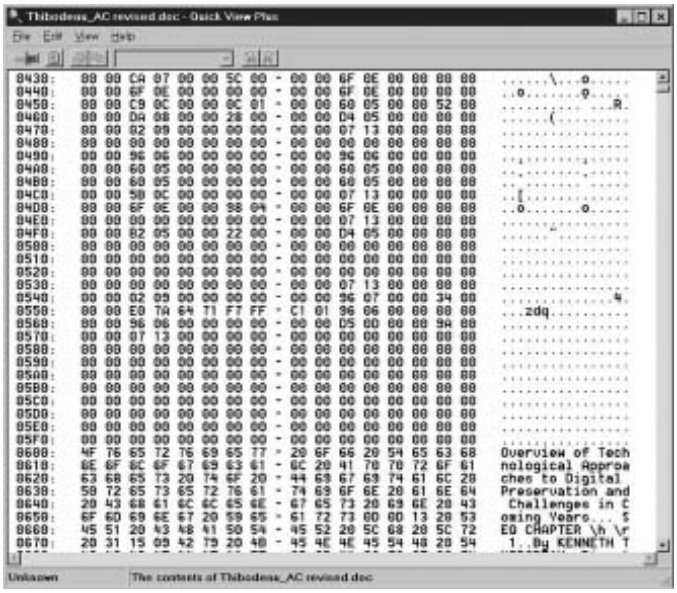


Fig. 1. Hexadecimal Dump of MS Word

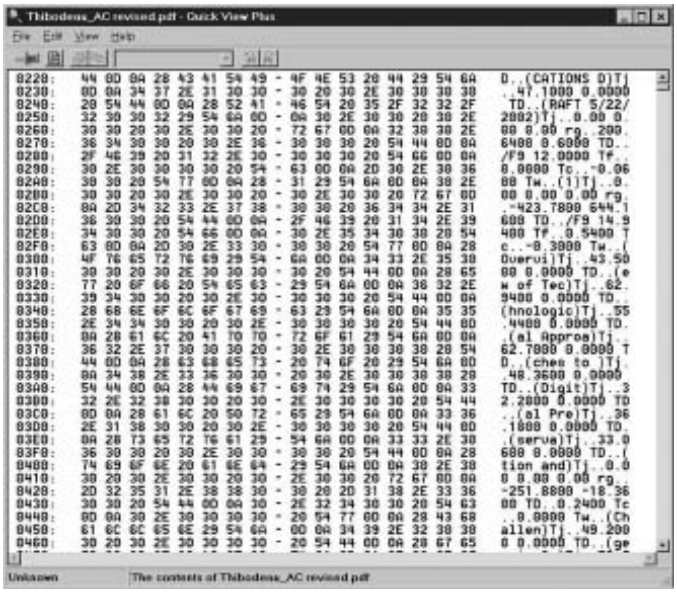


Fig. 2. Hexadecimal Dump of PDF

Is there any sense in which we could say that one of these digital formats is the true or correct logical representation of the document? An objective test would be whether the digital format preserves the document exactly as created. The most basic criterion is whether the document that is produced when the digital file is processed by the right software is identical to the original. In fact, each of these encodings, when processed by software that recognizes its data type, will display or print the document in the format in which it was created. So if the requirement is to maintain the content, structure, and visual appearance of the original document, either digital format is suitable. The two images are of Microsoft Word and Adobe PDF versions of the document. Other variants, such as WordPerfect, HTML, and even a scanned image of the printed document, would also satisfy the test of outputting the correct content in the original format.

This example reveals two important aspects of digital objects, each of which has significant implications for their preservation. The first is that there can be different digital encodings of the same conceptual object and that different encodings can preserve the essential characteristics of the conceptual object. The second relates to the basic concept of digital preservation.

With respect to the first of these implications, the possibility of encoding the same conceptual object in a variety of digital formats that are equally suitable for preserving the conceptual object can be extended to more complex types of objects and even to cases where the conceptual object is not presented to a human but is found only at the interface of two business applications. Consider the example of the cash withdrawal from an ATM. The essential record of that transaction consists of information identifying

the account from which the cash is withdrawn, the amount withdrawn, and the date and time of the transaction. For the transaction to be carried out, there must be an interface between the system that manages the ATM and the system that manages the account. The information about the transaction presented at the interface, in the format specified for that interface, is the conceptual object that corresponds to the withdrawal slip that would have been used to record the transaction between the account holder and a human teller. The two systems must share that interface object and, in any subsequent actions related to that withdrawal, must present the same information; however, there is no need for the two systems to use identical databases to store the information.

Before considering the implications for the nature of digital preservation, we should examine more fully the relationships among physical, logical, and conceptual objects.

Relationships: Where Things Get Interesting

The complex nature of a digital object having distinct physical, logical, and conceptual properties gives rise to some interesting considerations for digital preservation, especially in the relationships among the properties of any object at these three levels. The relationship between any two levels can be simple. It can be one-to-one; for example, a textual document saved as a Windows word processing file is a single object at all three levels. But a long textual report could be broken down into a master and three subdocuments in word processing format, leaving one conceptual object stored as four logical objects: a one-to-many relationship. If the word processing files relied on external font libraries, additional digital objects would be needed to reproduce the document. Initially, the master and subdocuments would probably be stored in as many physical files, but they might also be combined into a zip file or a Java ARchive (JAR) file. In this case, the relationship between conceptual and logical objects is one-to-many, and the relationship between logical and physical could be either one-to-one or many-to-one. To access the report, it would be necessary to recombine the master and subdocuments, but this amalgamation might occur only during processing and not affect the retention of the logical or physical objects.

Relationships may even be many-to-many. This often occurs in databases where the data supporting an application are commonly stored in multiple tables. Any form, report, or stored view defined in the application is a logical object that defines the content, structure, and perhaps the appearance of a class of conceptual objects, such as an order form or a monthly report. Each instance of such a conceptual object consists of a specific subset of data drawn from different tables, rows, and columns in the database, with the tables and columns specified by the form or report

and the rows determined in the first instance by the case, entity, event, or other scope specified at the conceptual level, e.g., order number, "x"; or monthly report for customer, "y"; or product, "z." In any instance, such as a given order, there is a one-to-many relationship between the conceptual and the logical levels, but the same set of logical objects (order form specification, tables) is used in every instance of an order, so the relationship between conceptual and logical objects are in fact many-to-many. In cases such as databases and geographic information systems, such relationships are based on the database model, but many-to-many relationships can also be established on an ad hoc basis, such as through hyperlinks to a set of Web pages or attachments to e-mail messages. Many-to-many relationships can also exist between logical and physical levels; for example, many e-mail messages may be stored in a single file, but attachments to messages might be stored in other files.

To preserve a digital object, the relationships between levels must be known or knowable. To retrieve a report stored as a master and several subdocuments, we must know that it is stored in this fashion and we must know the identities of all the logical components. To retrieve a specific order from a sales application, we do not need to know where all or any of the data for that order are stored in the database; we only need to know how to locate the relevant data, given the logical structure of the database.

We can generalize from these observations to state that, in order to preserve a digital object, we must be able to identify and retrieve all its digital components. The digital components of an object are the logical and physical objects that are necessary to reconstitute the conceptual object. These components are not necessarily limited to the objects that contain the contents of a document. Digital components may contain data necessary for the structure or presentation of the conceptual object. For example, font libraries for character-based documents and style sheets for HTML pages are necessary to preserve the appearance of the document. Report and form specifications in a database application are necessary to structure the content of documents.

In addition to identifying and retrieving the digital components, it is necessary to process them correctly. To access any digital document, stored bit sequences must be interpreted as logical objects and presented as conceptual objects. So digital preservation is not a simple process of preserving physical objects but one of preserving the ability to reproduce the objects. The process of digital preservation, then, is inseparable from accessing the object. You cannot prove that you have preserved the object until you have re-created it in some form that is appropriate for human use or for computer system applications.

To preserve a digital object, is it necessary to preserve its physical and

logical components and their interrelationship, without any alteration? The answer, perhaps surprisingly, is no. It is possible to change the way a conceptual object is encoded in one or more logical objects and stored in one or more physical objects without having any negative impact on its preservation. For example, a textual report may contain a digital photograph. The photograph may have been captured initially as a JPEG file and included in the report only by means of a link inserted in the word processing file, pointing to the image file. However, the JPEG file could be embedded in the word processing file without altering the report as such. We have seen another example of this in the different formats that can be used to store and reproduce this article. In fact, it may be beneficial or even necessary to change logical or physical characteristics to preserve an object. Authors often transform documents that they create as word processing documents into PDF format to increase the likelihood that the documents will retain their original appearance and to prevent users from altering their contents. An even simpler case is that of media migration. Digital media become obsolete. Physical files must be migrated to new media; if not, they will become inaccessible and will eventually suffer from the physical deterioration of the older media. Migration changes the way the data are physically inscribed, and it may improve preservation because, for example, error detection and correction methods for physical inscription on digital media have improved over time.

Normally, we would say that changing something directly conflicts with preserving it. The possibility of preserving a digital object while changing its logical encoding or physical inscription appears paradoxical and is compounded by the fact that it may be beneficial or even necessary to make such changes. How can we determine what changes are permissible and what changes are most beneficial or necessary for preservation? Technology creates the possibilities for change, but it cannot determine what changes are permissible, beneficial, necessary, or harmful. To make such determinations, we have to consider the purpose of preservation.

The Ultimate Outcome: Authentic Preserved Documents

What is the goal of digital preservation? For archives, libraries, data centers, or any other organizations that need to preserve information objects over time, the ultimate outcome of the preservation process should be authentic preserved objects; that is, the outputs of a preservation process ought to be identical, in all essential respects, to what went into that process. The emphasis has to be on the identity, but the qualifier of "all essential respects" is important.

The ideal preservation system would be a neutral communications channel for transmitting information to the future. This channel should not corrupt or change the messages transmitted in any way. You could conceive of a

digital preservation system as a black box into which you can put bit streams and from which you can withdraw them at any time in the future. If the system is trustworthy, any document or other digital object preserved in and retrieved from the system will be authentic. In abstract terms, we would like to be able to assert that, if X_{t0} was an object put into the box at time, t_0, and X_{tn} is the same object retrieved from the box at a later time, tn, then $X_{tn} = X_{t0}$.

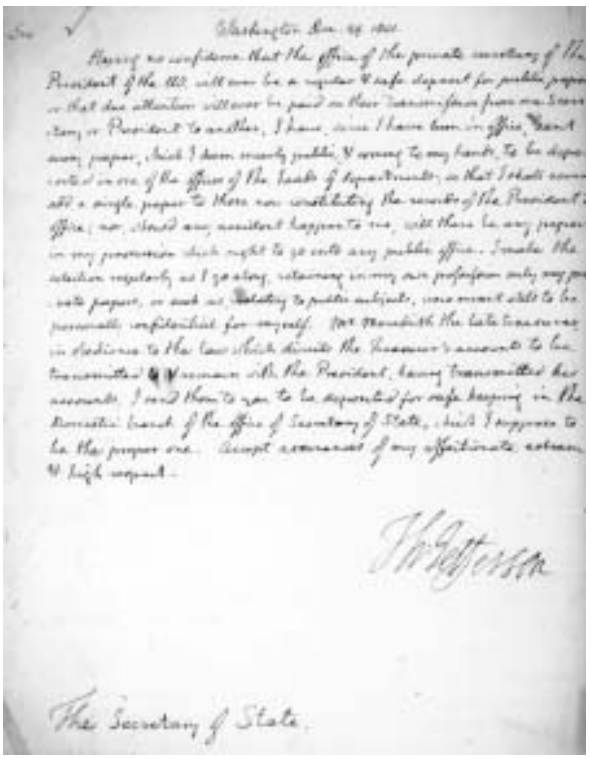
However, the analysis of the previous sections shows that this cannot be the case for digital objects. The process of preserving digital objects is fundamentally different from that of preserving physical objects such as traditional books or documents on paper. To access any digital object, we have to retrieve the stored data, reconstituting, if necessary, the logical components by extracting or combining the bit strings from physical files, reestablishing any relationships among logical components, interpreting any syntactic or presentation marks or codes, and outputting the object in a form appropriate for use by a person or a business application. Thus, it is impossible to preserve a digital document as a physical object. One can only preserve the ability to reproduce the document. Whatever exists in digital storage is not in the form that makes sense to a person or to a business application. The preservation of an information object in digital form is complete only when the object is successfully output. The real object is not so much retrieved as it is reproduced by processing the physical and logical components using software that recognizes and properly handles the files and data types (InterPARES Preservation Task Force 2001). So, the black box for digital preservation is not just a storage container: it includes a process for ingesting objects into storage and a process for retrieving them from storage and delivering them to customers. These processes, for digital objects, inevitably involve transformations; therefore, the equation, then $X_{tn} = X_{t0}$ cannot be true for digital objects.

In fact, it can be argued that practically, this equation is never absolutely true, even in the preservation of physical objects. Paper degrades, ink fades; even the Rosetta Stone is broken. Moreover, in most cases we are not able to assert with complete assurance that no substitution or alteration of the object has occurred over time. As Clifford Lynch has cogently argued, authentication of preserved objects is ultimately a matter of trust. There are ways to reduce the risk entailed by trusting someone, but ultimately, you need to trust some person, some organization, or some system or method that exercises control over the transmission of information over space, time, or technological boundaries. Even in the case of highly durable physical objects such as clay tablets, you have to trust that nobody substituted forgeries over time (Lynch 2000). So the equation for preservation needs to be reformulated as $X_{tn} = X_{t0}$ + delta(X), where

delta(X) is the net effect of changes in X over time.

But can an object change and still remain authentic? Common sense suggests that something either is or is not authentic, but authenticity is not absolute. Jeff Rothenberg has argued that authenticity depends on use (Rothenberg 2000). More precisely, the criteria for authenticity depend on the intended use of the object. You can only say something is authentic with respect to some standard or criterion or model for what X is.

Consider the simple example shown in figure 3. It shows a letter, preserved in the National Archives, concerning the disposition of Thomas Jefferson's papers as President of the United States (Jefferson 1801). Is this an authentic copy of Thomas Jefferson's writing? To answer that question, we would compare it to other known cases of Thomas Jefferson's handwriting. The criteria for authentication would relate to the visual appearance of the text. But what if, by "Jefferson's writing," we do not mean his handwriting but his thoughts? In that case, the handwriting becomes irrelevant: Jefferson's secretary may have written the document, or it could even be a printed version. Conversely, a document known to be in Jefferson's handwriting, but containing text he copied from a book, does not reveal his thoughts. Authenticating Jefferson's writing in this sense relates to the content and style, not to the appearance of the text. So authenticating something as Jefferson's writing depends on how we define that concept.



The Secretary of State.

Fig. 3. Jefferson note

There are contexts in which the intended use of preserved information objects is well-known. For example, many corporations preserve records for very long times for the purpose of protecting their property rights. In such cases, the model or standard that governs the preservation process is that of a record that will withstand attacks on its reliability and authenticity in litigation. Institutions such as libraries and public archives, however, usually cannot prescribe or predict the uses that will be made of their holdings. Such institutions generally maintain their collections for access by anyone, for whatever reason. Where the intentions of users are not known in advance, one must take an "aboriginal" approach to authenticity; that is, one must assume that any valid intended use must be somehow consonant with the original nature and use of the object. Nonetheless, given that a digital information object is not something that is preserved as an inscription on a physical medium, but something that can only be constructed—or reconstructed—by using software to process stored inscriptions, it is necessary to have an explicit model or standard that is independent of the stored object and that provides a criterion, or at least a benchmark, for assessing the authenticity of the reconstructed object.

Ways to Go: Selecting Methods

What are the possibilities for preserving authentic digital information objects? Among these possibilities, how can we select the best option or options? Four criteria apply in all cases: any method chosen for preservation must be feasible, sustainable, practicable, and appropriate. *Feasibility* requires hardware and software capable of implementing the method. *Sustainability* means either that the method can be applied indefinitely into the future or that there are credible grounds for asserting that another path will offer a logical sequel to the method, should it cease being sustainable. The sustainability of any given method has internal and external components: internally, the method must be immune or isolated from the effects of technological obsolescence; externally, it must be capable of interfacing with other methods, such as for discovery and delivery, which will continue to change. *Practicality* requires that implementation be within reasonable limits of difficulty and expense. *Appropriateness* depends on the types of objects to be preserved and on the specific objectives of preservation. With respect to the types of objects to be preserved, we can define a spectrum of possibilities running from preserving technology itself to preserving objects that were produced using information technology (IT). Methods can be aligned across this spectrum because the appropriateness of any preservation method depends on the specific objectives for preservation in any given case. As discussed earlier, the purposes served by preservation can vary widely. Considering where different methods fall across this spectrum will provide a basis for evaluating their appropriateness for any given purpose.

To show the rationale of the spectrum, consider examples at each end. On the "preserve technology" end, one would place preserving artifacts of technology, such as computer games. Games are meant to be played. To play a computer game entails keeping the program that is needed to play the game operational or substituting an equivalent program, for example, through reverse engineering, if the original becomes obsolete. On the "preserve objects" end, one would place preserving digital photographs. What is most important is that a photograph present the same image 50 or 100 years from now as it does today. It does not really matter what happens to the bits in the background if the same image can be retrieved reliably. Conversely, if a digital photograph is stored in a physical file and that file is maintained perfectly intact, but it becomes impossible to output the original image in the future—for example, because a compression algorithm used to create the file was either lossy or lost—we would not say the photograph was preserved satisfactorily.

But these illustrations are not completely valid. Many computer games have no parallels in the analog world. Clearly they must be preserved as artifacts of IT. But there are many games now played on computers that existed long before computers were invented. The card game, solitaire, is one example. Obviously, it could be preserved without any computer. In fact, the most assured method for preserving solitaire probably would be simply to preserve the rules of the game, including the rules that define a deck of cards. So the most appropriate method for preserving a game depends on whether we consider it to be essentially an instance of a particular technology—where "game" is inseparable from "computer"—or a form of play according to specified rules; that is, a member of a class of objects whose essential characteristics are independent of the technology used to produce or implement them. We have to preserve a computer game in digital form only if there is some essential aspect of the digital form than cannot be materialized in any other form or if we wish to be able to display, and perhaps play, a specific version of the computer game.

The same analysis can be applied to digital photographs. With traditional photographs, one would say that altering the image that had been captured on film was contrary to preserving it. But there are several types of digital photographs where the possibilities of displaying different images of the same picture are valuable. For example, a traditional chest X-ray produced three pieces of film, and, therefore, three fixed images. But a computerized axial tomography (CAT) scan of the chest can produce scores of different images, making it a more flexible and incisive tool for diagnosis. How should CAT scans be preserved? It depends on our conception or model of what a CAT scan is. If we wanted to preserve the richest source of data about the state of a particular person's body at a given time, we would have to preserve the CAT scan as an instance of a specific type of technology. But if we needed to preserve a record of the specific image

that was the basis for a diagnosis or treatment decision, we would have to preserve it as a specific image whose visual appearance remains invariant over time. If the first case, we must preserve CAT scanning technology, or at least that portion of it necessary to produce different images from the stored bit file. It is at least worth considering, in the latter case, that the best preservation method, taking feasibility and sustainability into account, would be to output the image on archival quality photographic film.

Here, in the practical context of selecting preservation methods, we see the operational importance of the principle articulated in discussing the authenticity of preserved objects: we can determine what is needed for preservation only on the basis of a specific concept or definition of the essential characteristics of the object to be preserved. The intended use of the preserved objects is enabled by the articulation of the essential characteristics of those objects, and that articulation enables us not only to evaluate the appropriateness of specific preservation methods but also to determine how they should be applied in any case. Applying the criterion of appropriateness, we can align various preservation methods across the spectrum of "preserve technology"—"preserve objects."

More than a Spectrum: A Two-Way Grid

For any institution that intends or needs to preserve digital information objects, selection of preservation methods involves another dimension: the range of applicability of the methods with respect to the quantity and variety of objects to be preserved. Preservation methods vary greatly in terms of their applicability. Some methods apply only to specific hardware or software platforms, others only to individual data types. Still others are very general, applicable to an open-ended variety and quantity of digital objects. The range of applicability is another basis for evaluating preservation methods. Organizations that need to preserve only a limited variety of objects can select methods that are optimal for those objects. In contrast, organizations responsible for preserving a wide variety must select methods with broad applicability. Combining the two discriminants of appropriateness for preservation objectives and range of applicability defines a two-dimensional grid in which we can place different preservation methods and enrich our ability to evaluate them.

Figure 4 shows this grid, with a number of different methods positioned in it. Two general remarks about the methods displayed in this grid are in order. On the one hand, the methods included in it do not include all those that have been proposed or tried for digital preservation. In particular, methods that focus on metadata are not included. Rather, the emphasis is on showing a variety of ways of overcoming technological obsolescence. Even here, the cases included are not exhaustive; they are only illustrative of the range of possibilities. On the other hand, some methods are included

that have not been explicitly or prominently mentioned as preservation methods. There is a triple purpose for this. The first purpose is to show the robustness of the grid as a framework for characterizing and evaluating preservation methods. The second is to emphasize that those of us who are concerned with digital preservation need to be open to the possibilities that IT is constantly creating. The third purpose is to reflect the fact that, in the digital environment, preservation is not limited to transmitting digital information over time. The same factors are in play in transmitting digital information across boundaries in space, technology, and institutions. Therefore, methods developed to enable reliable and authentic transmission across one of these types of boundaries can be applicable across others (Thibodeau 1997).

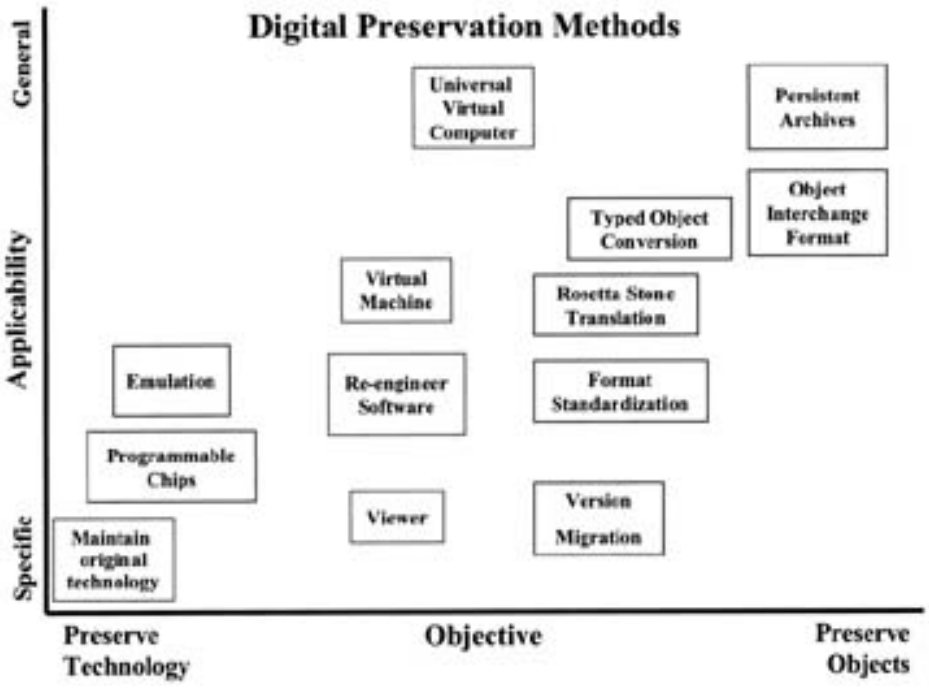


Fig. 4. Digital Preservation Methods

Sorting IT Out

Discussions of digital preservation over the last several years have focused on two techniques: emulation and migration. Emulation strives to maintain the ability to execute the software needed to process data stored in its "original" encodings, whereas migration changes the encodings over time so that we can access the preserved objects using state-of-the-art software in the future. Taking a broader perspective, IT and computer science are offering an increasing variety of methods that might be useful for long-term preservation. These possibilities do not fit nicely into the simple bifurcation of emulation versus migration. We can position candidate methods across the preservation spectrum according to the following

principles:

- On the "preserve technology" end of the spectrum, methods that attempt to keep data in specific logical or physical formats and to use technology originally associated with those formats to access the data and reproduce the objects.
- In the middle of the spectrum, methods that migrate data formats as technology changes, enabling use of state-of-the-art technology for discovery, access, and reproduction.
- On the "preserve objects" end of the spectrum, methods that focus on preserving essential characteristics of objects that are defined explicitly and independently of specific hardware or software.

There are various ways one can go about all these options. For example, if we focus on the "preserve technology" end, we start with maintaining original technology, an approach that will work for some *limited* time. Even for preservation purposes, it can be argued that this approach is often the only one that can be used.

Preserving Technology: The Numbers Add Up, and Then Some

The starting point for all digital preservation is the technology and data formats used to create and store the objects. Digital information objects can be preserved using this "original" technology for 5 to 10 years, but eventually the hardware, software, and formats become obsolete. Trying to preserve specific hardware and software becomes increasingly difficult and expensive over time, with both factors compounded by the variety of artifacts that need to be preserved. Over the long term, keeping original technology is not practicable and may not be feasible.

Enter the Emulator

Various approaches can be used to simplify the problem while still keeping data in their original encodings. The best-known approach is emulation. Emulation uses a special type of software, called an emulator, to translate instructions from original software to execute on new platforms. The old software is said to run "in emulation" on newer platforms. This method attempts to simplify digital preservation by eliminating the need to keep old hardware working. Emulators could work at different levels. They could be designed to translate application software to run on new operating systems, or they could translate old operating system commands to run on new operating systems. The latter approach is simpler in that the former would require a different emulator for every application, and potentially for every version of an application, while the latter should enable all

applications that run on a given version of an operating system to execute using the same emulator.

While proponents of emulation argue that it is better than migration because at every data migration there is a risk of change, emulation entails a form of migration. Emulators themselves become obsolete; therefore, it becomes necessary either to replace the old emulator with a new one or to create a new emulator that allows the old emulator to work on new platforms. In fact, if you get into an emulation strategy, you have bought into a migration strategy. Either strategy adds complexity over time.

Emulation is founded on the principle that all computers are Turing machines and that any command that can run on one Turing machine can run on any other Turing machine. There is, however, evidence that this principle breaks down at an empirical level. For example, basic differences such as different numbers of registers or different interrupt schemes make emulation unreliable, if not impossible (IEEE 2001).

Reincarnation for Old Machines

Another technique that keeps old software running takes the opposite approach from emulation: it relies on a special type of hardware, rather than software emulators. It does this by re-creating an old computer on a configurable chip. An entire computer system could be reincarnated by being programmed on a new, configurable chip. The configurable chip constitutes a single point of failure, but that can readily be offset. If the chip begins to fail or becomes obsolete, the old system could simply be programmed on a newer chip. Intuitively, configurable chips seem like a simpler approach than emulation.

Compound Disinterest

While emulation and configurable chips take opposite directions, they present some common problems. First, current technology is not perfect. There are anomalies and bugs. Any preservation strategy that relies on specific software is carrying all the problems associated with those products into the future. Not all these problems get fixed. For example, it is not always possible to figure out what causes a problem such as a general protection fault, because there are too many variables involved. Furthermore, fixes can increase the complexity of preservation strategies that rely on keeping old software running, because they increase the number of versions of software that are released. Logically, if the authenticity of digital information depends on preserving original data formats and using them with the original software, each format should be processed with the version of the software used to produce it.

Software defects aside, the combinatorics entailed by strategies that involve preserving ever-increasing varieties of data formats, application software, and operating systems are frightening. With new versions being released every 18-24 months, over 25-years or longer, one would need to support thousands of combinations of applications, utilities, operating systems, and formats.

The viability of these strategies gets much more complex when the focus shifts from a single system to combinations of systems, which is the norm today. Emulation and programmable chips might be viable strategies if all we had to cope with were the products of desktop PCs, but not in today's world, where the objects to be preserved often involve a diverse palette of technologies, such as various client-server applications where the servers use different operating systems, distributed applications running on heterogeneous platforms, and virtual machines such as Java. Providing technical support for operations of such a daunting variety of makes, models, and versions may be neither feasible nor affordable, because you would have to get all these applications running in emulation at the same time.

Complexity also increases in the case of collections of documents accumulated over time. Most government records, for example, are accumulated over many years, often many decades. Following the most fundamental principles of archival science—respect for provenance and for original order—we cannot segregate records according to their digital formats. We must preserve and provide access to aggregates of records established by their creators. Under a strategy of preserving technology, doing research in such series would entail using all the different software products used to produce the records.

Even if it were technically and financially possible to keep the technologies operative, staffing a help desk to support end users is inconceivable, especially since most users in the future will never have encountered—not to mention learned how to use—most of the products they will need to access the preserved information objects. Even if it were possible to provide adequate support to a user perusing, for example, a single case file accumulated over 20 years, it is not obvious that this would be deemed an acceptable level of support, because it would cut users off from the possibility of using more advanced technologies for discovery, delivery, and analysis.

Scenarios pegged on preserving specific technology, maintaining the links between specific software and specific data formats, run counter to the major direction of information technology. E-commerce and e-government require that the information objects created and used in these activities be

transportable among the parties involved, independent of the hardware and the software each party uses at any time. Neither e-commerce nor e-government would be possible if the necessary information had to be accessed in original formats using obsolete technologies. Preserve technology strategies will depend on niche technologies and cannot expect widespread support in the IT market.

In this approach, one also encounters some interesting issues of intellectual property rights—not only the usual issues of copyright but also the ownership that the software companies assert over their formats even when they do not own the content.

A View Toward Further Simplification

Various software-engineering methods provide simpler ways of keeping obsolete formats accessible by concentrating on specific requirements.

One such method focuses on documents, a class of objects in which the functionality that has to be preserved is simply the ability to present them visually on a screen or printed page. For such objects, the only specific software needed for preservation is software that reliably renders the content with its original look and feel. This approach is being used in the Victorian Electronic Records System (VERS) developed for the Public Record Office of the State of Victoria, Australia. The system stipulates converting documents created in desktop computing environments to Adobe's PDF format. Instead of attempting to run versions of Acrobat reader for PDF indefinitely in the future, the VERS project conducted an experiment to demonstrate that it is possible to construct a viewer from the published specifications of the PDF format. The VERS approach embodies a combination of format migration, in that the various formats in which records are originally created must be translated to PDF with software reengineering. Similar approaches could be applied to other data types whose essential functionality is presentation in page image.

Finding Virtue in Abstraction

Another application of software engineering involves developing virtual machines that can execute essential functions on a variety of platforms. The Java language is an example of a virtual machine, although it was not developed for purposes of preservation. The virtual machine approach avoids the need for emulator software by providing required functionality in a virtual machine that, in principle, can be implemented on a great variety of computing platforms indefinitely into the future. Raymond Lorie of the IBM-Almaden Research Center has launched an effort to develop a Universal Virtual Computer (UVC) that would provide essential functionality

for an unlimited variety of data types. Following this strategy, objects would be preserved in their original formats, along with the rules for encoding and decoding data in those formats. The rules are written in a machine language that is completely and unambiguously specified. The language is so simple that it can be interpreted to run on any computer in the future. When the UVC program executes, the preserved data are interpreted according to a logical schema for the appropriate data type and output, and each data element bears a semantic tag defined in the logical schema. This approach avoids much of the complexity of emulation and configurable chips, but there are some trade-offs. The UVC only provides a limited set of basic functions. It also sacrifices performance: software that can run on any platform is not optimized for any one of them (Lorie 2000).

Accepting Change: Migration Strategies

In the middle of the spectrum fall data migration approaches that abandon the effort to keep old technology working or to create substitutes that emulate or imitate it. Instead, these approaches rely on changing the digital encoding of the objects to be preserved to make it possible to access those objects using state-of-the-art technology after the original hardware and software become obsolete. There are a variety of migration strategies.

Simple Version Migration

The most direct path for format migration, and one used very commonly, is simple version migration within the same family of products or data types. Successive versions of given formats, such as Corel WordPerfect's WPD or Microsoft Excel's XLS, define linear migration paths for files stored in those formats. Software vendors usually supply conversion routines that enable newer versions of their product to read older versions of the data format and save them in the current version.

Version migration sets up a chain that must be extended over time, because every format will eventually become obsolete. One problem with this approach is that using more recent versions of software, even with the original formats, may present the preserved documents with characteristics they did not, and perhaps could not, have had. For example, any document created with a word processor in the early 1990s, before "WYSIWYG" display was available, would have appeared on screen with a black background and green letters. If one were to open such a document with a word processor today, it would look much like a printed page.

Software vendors control this process of version migration. Their conversion utilities are designed to migrate data types and do not provide for explicit or specific control according to attributes defined at the

conceptual level. Each successive migration will accumulate any alterations introduced previously. Another potential problem is that over time, product lines, and the migration path, may be terminated.

Format Standardization

An alternative to the uncertainties of version migration is format standardization, whereby a variety of data types are transformed to a single, standard type. For example, a textual document, such as a WordPerfect document, could be reduced to plain ASCII. Obviously, there would be some loss if font, type size, and formatting were significant. But this conversion is eminently practicable, and it would be appropriate in cases where the essential characteristics to be preserved are the textual content and the grammatical structure. Where typeface and font attribute are important, richer formats, such as PDF or RTF, could be adopted as standards. The low common denominator provides a high guarantee that the format will be successful, at least for preserving appearance. For types of objects where visual presentation is essential, bit-mapped page images and hard copy might be acceptable: 100 years from now, IT systems will be able to read microfilm. In fact, according to companies such as Kodak and Fuji, it can be done today.

For socioeconomic and other data sets created to enable a variety of analyses, the data structure can often be preserved in a canonical form, such as arrays or relational tables, independently of specific software. Such formats are either simple enough or so unambiguously defined that it is reasonable to assume that information systems in the future will be able to implement the structures and process the data appropriately.

In principle, the standard format should be a superclass of the original data types—one that embodies all essential attributes and methods of the original formats. This is not necessarily the case, so there may be significant changes in standardization, just as with version migration. Moreover, standards themselves evolve and become obsolete. So, except for the simplest formats, there is a likely need for repeated migrations from one standard format to another, with consequent accumulation of changes.

Typed Object Model Conversion

Another approach to migrating data formats into the future is Typed Object Model (TOM) Conversion. The TOM approach starts out with the recognition that all digital data things are objects, that is, they have specified attributes, specified methods or operations, and specific semantics. All digital objects belong to one or another type of digital object, where "type" is defined by given values of attributes, methods, or semantics for that

class of objects. A Microsoft Word 6 document, for example, is a type of digital object defined by its logical encoding. An e-mail is a type of digital object defined, at the conceptual and logical levels, by essential data elements, e.g., "To," "From," "Subject," or "Date."

Any digital object is a byte sequence and has a format, i.e., a specified encoding of that object for its type. Byte sequences can be converted from one format to another, as shown in the earlier example of this document encoded in Microsoft Word and PDF formats. But within that range of possible conversion, the essential properties of a type or class of objects define "respectful conversions," that is, conversions whose result cannot be distinguished when viewed for an interface of that type. The content and appearance of the document in this example remains identical whether it is stored as a Word or PDF file; therefore, conversion between those two formats is respectful for classes of objects whose essential properties are content and appearance (Wing and Ockerbloom 2000). There is a TOM conversion available online that is capable of doing respectful conversions of user submitted files in some 200 formats.

Rosetta Stones Translation

Another migration approach under development is called Rosetta Stones. Arcot Rajasekar of the San Diego Supercomputer Center is developing this approach. Like TOM, this approach starts with data types, but rather than articulating the essential properties of each type, it constructs a representative sample of objects of that type. It adds a parallel sample of the same objects in another, fully specified type, and retains both. For example, if one wanted to preserve textual documents that had been created in WordPerfect 6, one would create a sample of files in version 6 of the WPD format that embodies all the significant features of this format. Then one would duplicate each of the documents in this sample in another format that might be human-readable computer output microfilm (COM) or paper, because we know that we will always be able to read in those human-readable versions. This second sample constitutes a reference set, like the Greek in the original Rosetta Stone. The triad of samples in the original data type, the reference set, and the target type constitutes a digital Rosetta Stone from which rules for translating from the original to the target encoding can be derived.

Given the reference sample—e.g., the printed version of documents—and the rules for encoding in a target format that is current at any time in the future, we can create a third version of the sample in the target format. By comparing the target sample with the original sample, we can deduce the rules for translating from the original to the target format and apply these rules to convert preserved documents from the original to the target format. This approach avoids the need for repeated migrations over time.

Even though the target formats can be expected to become obsolete, migration to subsequent formats will be from the original format, not from the earlier migration. Important to the success of this approach is the ability to construct a parallel sample in a well-characterized and highly durable type. It is not evident that it will be possible to do this for all data types, especially more complex types that do not have analog equivalents, but research on this approach is relatively recent.

Object Interchange Format

Another approach enables migration through an object interchange format defined at the conceptual level. This type of approach is being widely adopted for e-commerce and e-government where participants in a process or activity have their own internal systems, which cannot readily interact with systems in other organizations. Rather than trying to make the systems directly interoperable, developers are focusing on the information objects that need to be exchanged to do business or otherwise interact. These objects are formally specified according to essential characteristics at the conceptual level, and those specifications are articulated in logical models. The logical models or schema define interchange formats. To collaborate or interact, the systems on each side of a transaction need to be able to export information in the interchange format and to import objects in this format from other systems. While it was designed for internal markup of documents, the XML family of standards has emerged as a major vehicle for exchange of digital information between and among different platforms.

A significant example of this approach concerns financial reports. There are several types of financial reports that essentially all corporations produce and share with their business partners and with government agencies around the world. The extensible business reporting language (XBRL) is an initiative to enable exchange of these reports, regardless of the characteristics of systems used either to produce or receive the reports. The initiative comprises major professional organizations of accountants from the United States, Canada, the United Kingdom, Australia, and several non-English-speaking countries, major accounting firms, major corporations, IT companies, and government agencies. XBRL defines a single XML schema that covers all standard financial reports. The schema defines an interchange format. Any system that can export and import data in that format can exchange financial reports and data with any other system with XBRL I/O capability, regardless of the hardware or software used in either case. At the logical level, the XBRL schema is impressively simple. That simplicity is enabled by an extensive ontology of accounting terms at the conceptual level. This approach is obviously driven by short-term business needs, but a method that allows reliable exchange of important financial data across heterogeneous computing platforms around

the world can probably facilitate transmission of information over generations of technology. Given that XML schemas and tags are constructed using plain ASCII and can be interpreted by humans, it is likely that future computer systems will be able process them correctly. Thus, the object interchange method can become a preservation method simply by retaining the objects in the interchange format and, on an as-needed basis, building interpreters to enable target systems in the future to import objects in such formats.

To some extent, object interchange formats have the same purpose as do samples in well-known data types in the Rosetta Stones method: they serve as a bridge between heterogeneous systems and data types. While the Rosetta Stones method is more generic, object interchange specifications have a significant advantage in that the essential properties of the objects are defined by experts who have substantial knowledge of their creation and use. Thus, unlike all the other approaches considered so far, object interchange formats embed domain knowledge in the transmission of information objects across space, time, and technologies. The object interchange model lies close to the "preserve objects" end of the preservation spectrum. It could be said to lie midway between specific and general in its applicability because it provides a single method that potentially could be applied to a great variety of objects and data types, but addresses only the persistence of content and form across technological boundaries.

Preserving Objects: Persistent Archives

A promising approach, persistent archives, has been articulated over the last four years, primarily at the San Diego Supercomputer Center in research sponsored by the Defense Advanced Research Projects Agency, the National Science Foundation, and the National Archives and Records Administration. It has many elements in common with other approaches described in this paper, but it is also markedly different than these other strategies. Like the UVC, it relies on a high level of abstraction to achieve very broad applicability. Like TOM and Rosetta Stones, it addresses the specific characteristics of logical data types. Like object interchange formats and the UVC, it tags objects to ensure the persistence of syntactic, semantic, and presentation elements. Like migration, it transforms the logical encoding of objects, but unlike migration, the transformations are controlled not by target encodings into which objects will be transformed but by the explicitly defined characteristics of the objects themselves. It implements a highly standardized approach, but unlike migration to standard format, it does not standardize on logical data types, but at a higher level of abstraction: on the method used to express important properties, such as context, structure, semantics, and presentation.

The most important difference between persistent archives and the other approaches described is that the former strategy is comprehensive. It is based on an information management architecture that not only addresses the problem of obsolescence but also provides the functionality required for long-term preservation, as stipulated in the OAIS standard. Furthermore, it provides a coherent means of addressing the physical, logical, and conceptual properties of the objects being preserved through the data, information, and knowledge levels of the architecture. Persistence is achieved through two basic routes: one involving the objects themselves, the other the architecture. Objects are preserved in persistent object format, which is relatively immune to the continuing evolution of IT. The architecture enables any component of hardware or software to be replaced with minimum impact on the archival system as a whole. The architecture is notional. It does not prescribe a specific implementation.

The cornerstone of the persistent archives approach is the articulation of the essential characteristics of the objects to be preserved—collections as well as individual objects—in a manner that is independent of any specific hardware or software. This articulation is expressed at the data level by tags that identify every byte sequence that must be controlled to ensure preservation. In effect, tags delimit atomic preservation units in physical storage. The granularity of these data units can vary greatly, depending on requirements articulated at the information and knowledge levels. Every tag is linked to one or more higher-level constructs, such as data models, data element definitions, document type definitions, and style sheets defined at the information level, and ontologies, taxonomies, thesauri, topic maps, rules, and textbooks at the knowledge level. In research tests on a wide variety of data types, conceptual objects, and collections, it has been shown that simple, persistent ASCII tags can be defined to identify, characterize, and control all data units. The research has shown that XML is currently the best method for tagging and articulating requirements at the information level and, to some extent, at the knowledge level; however, it would be wrong to conclude that persistent archives are based or dependent on XML. Rather, persistent archives currently use XML, but there is nothing in the architecture that would preclude using other implementation methods should they prove superior.

The architecture is structured to execute the three basic processes required in the Open Archival Information System (OAIS) standard: *ingest*, for bringing objects into the system; *management*, for retaining them over time; and *access*, for disseminating them to consumers. In ingest, objects in obsolescent formats are transformed into persistent format, through parsing and tagging of data units as described earlier, or, if they are already in persistent format, by verifying that fact at the data, information, and knowledge levels. Over time, data units are maintained in storage, and the metadata and domain knowledge that are necessary to retrieve, use,

and understand the data are maintained in models, dictionaries, and knowledge bases. When access to a preserved object is desired, the data are retrieved from storage and the object is materialized in a target technology current at the time. This materialization requires translating from the persistent form to the native form of the target technology. If the three basic processes are conceived as columns and the three levels (data, information, knowledge) as rows, the persistent archives architecture can be depicted in a 3-by-3 grid (Moore et al. 2000).

The persistent archives architecture is independent of the technology infrastructure in which it is implemented at any time. It achieves this independence through loose coupling of its basic building blocks, using software mediators to link each pair of adjacent blocks. Interactions are between adjacent blocks vertically and horizontally, but not diagonally. Over time, as the components used to implement any block are updated, there is no need to change any of the other blocks, only the mediators.

Conclusion: The Open End

There is an inherent paradox in digital preservation. On the one hand, it aims to deliver the past to the future in an unaltered, authentic state. On the other hand, doing so inevitably requires some alteration. All the methods described in this paper entail altering or replacing hardware, software, or data, and sometimes more than one of these. This paradox is compounded by the fact that in the future, as today, people will want to use the best available technology—or at least technologies they know how to use—for discovery, retrieval, processing, and delivery of preserved information. There is a danger that to the degree that preservation solutions keep things unaltered they will create barriers to satisfying this basic user requirement. Adding to this the recognition that the problem of digital preservation is not static, that it will continue to evolve as information technology and its application in the production of valuable information change, reinforces the paradox to the point that any solution to the challenge of digital preservation must be inherently evolutionary. If the preservation solution cannot grow and adapt to continuing changes in the nature of the problem and continuing escalation of user demands, the "solution" itself will in short order become part of the problem; that is, it will itself become obsolete.

This paradox can be resolved only through the elaboration of a basic conceptual framework for digital preservation—a framework that allows us to identify and analyze all that is involved in the process of digital preservation and to understand how different facets of that process affect each other. Fortunately, such a framework has been articulated over the last few years and has become an international standard. It is the OAIS reference model. While the OAIS model was developed for the space

science community, its articulation was, from the beginning, both international and multidisciplinary. As a result, the model has broad applicability. The OAIS model provides a frame of reference in which we can balance the need for preserving digital objects unaltered and the need to keep pace with changing IT, both to encompass new classes of digital objects and to capitalize on technological advances to improve preservation services (ISO 2002).

However, the OAIS model is too generalized to suffice for implementation. It needs to be refined and extended to be useful in specific domains. One example of such refinement has been articulated for the domain of records. The International research on Permanent Authentic Records in Electronic Records (InterPARES) project is a multinational, multidiscipline research collaboration whose name reflects its basic objective. To fine-tune the OAIS framework for the specific goal of preserving authentic records, the InterPARES Project developed a formal Integrated DEFinition (IDEF) process model for what is required to preserve authentic digital records. This "Preserve Electronic Records" model retains the functions of an OAIS but adds specific archival requirements and archival knowledge. Archival requirements act as specific controls on the preservation process, and archival knowledge was the basis for further refinement of the preservation process. In turn, the process of developing the archival model led to advances in archival knowledge; specifically, to clarification of the characteristics of electronic records at the physical, logical, and conceptual levels, and to improvements in our understanding of what it means to preserve electronic records. The InterPARES Preserve Electronic Records model includes specific paths for accommodating new classes of electronic records over time and for taking advantage of improvements in IT (Preservation Task Force in press).

The InterPARES model illustrates how, starting from the OAIS reference model, one can construct an open-ended approach to digital preservation and effectively address the paradoxical challenge of digital preservation. This case can serve as an example for other domains. There is undeniably a pressing need for technological methods to address the challenge of digital preservation. There is a more basic need for an appropriate method of evaluating alternative methods, such as the two-way grid described in this paper. Finally, there is an overriding need to select and implement preservation methods in an open-ended system capable of evolving in response to changing needs and demands.

Footnotes

[1] An earlier version of this paper appeared as "Digital Preservation Techniques: Evaluating the Options" in *Archivi & Computer: Automatione e Beni Culturali* 10 (2/01): 101-109.

[2] Each image displays the hexadecimal values of (1) in the leftmost column, the position of first byte in that row relative to the start of the file, and (2) the numeric values of 16 bytes starting with the numbered one. It also shows the printable ASCII characters, or a '.' for unprintable bytes in the rightmost column.

References

All URLs were valid as of July 10, 2002.

Institute of Electrical and Electronics Engineers, Inc. (IEEE). June 2001. Transactions on Computers—Special Issue on Dynamic Optimization.

International Standards Organization (ISO). 2002. Open Archival Information System—Reference Model. The draft international standard is available at http://ssdoo.gsfc.nasa.gov/nost/isoas/overview.html.

InterPARES Project Preservation Task Force. 2001. How to Preserve Electronic Records. Available at www.interpares.org.

InterPARES Project Preservation Task Force. In press. Report of the Preservation Task Force. A preliminary version of the report is available at www.interpares.org.

Jefferson, Thomas. 1801. Note from Thomas Jefferson regarding the disposition of his Presidential papers, December 29, 1801. Washington, D.C.: National Archives, General Records of the Department of State. RG 59. Available at http://nara.gov/education/teaching/archives/tjletter.html.

Lorie, Raymond A. 2000. The Long-Term Preservation of Digital Information. Available at http://www.si.umich.edu/CAMILEON/Emulation%20papers%20and%20publications/Lorie.pdf.

Lynch, Clifford. 2000. Authenticity and Integrity in the Digital Environment: An Exploratory Analysis of the Central Role of Trust. In *Authenticity in a Digital Environment*. Washington, D.C.: Council on Library and Information Resources.

Available at http://www.clir.org/pubs/abstract/pub92abst.html.

Moore, Reagan et al. 2000. Collection-Based Persistent Digital Archives—Part 1. *D-Lib Magazine* 6(3). Available at http://www.dlib.org/dlib/march00/moore/03moore-pt1.html.

Rajasekar, Arcot, Reagan Moore, and Michael Wan. Syntactic and Semantic Replicas: Rosetta Stones for Long-Term Digital Preservation. Unpublished manuscript.

Rothenberg, Jeff. 2000. Preserving Authentic Digital Information. In *Authenticity in a Digital Environment*. Washington, D.C.: Council on Library and Information Resources. Available at http://www.clir.org/pubs/abstract/pub92abst.html.

Task Force on Archiving of Digital Information. 1996. *Preserving Digital Information. Report of the Task Force on Archiving of Digital Information.* Washington, D.C.: Commission on Preservation and Access, and Mountain View, Calif.: Research Libraries Group. Available at http://www.rlg.org/ArchTF/.

Thibodeau. Kenneth. 1997. Boundaries and Transformations: An Object-Oriented Strategy for Preservation of Electronic Records. European Commission. In *INSAR Supplement II: the Proceedings of the DLM-Forum on Electronic Records*. Luxembourg: Office for Official Publications of the European Communities.

Wing, Jeannette M., and John Ockerbloom. 2000. Respectful Type Converters. *IEEE Transactions on Software Engineering* 26(7): 579-93.

Web sites noted in this paper:

Extensible Business Reporting Language (XBRL). Available at http://www.xbrl.org/.

Typed Object Model (TOM). Available at http://tom.cs.cmu.edu/intro.html.

Victorian Electronic Records Strategy (VERS). Available at http://www.prov.vic.gov.au/vers/.

Next Previous

Return to CLIR Home Page >>