Attributes of File Formats for Long-Term Preservation of Scientific and Engineering Data in Digital Libraries

Mike Folk
National Computational Sciences Alliance
University of Illinois at Urbana-Champaign
Champaign, IL 60292 USA
+1 217 244 0072

mfolk@ncsa.uiuc.edu

Bruce R. Barkstrom
Atmospheric Sciences Data Center
NASA Langley Research Center
Hampton, VA 23681-2199
+1 757 864 5676

b.r.barkstrom@larc.nasa.gov

ABSTRACT
In this paper, we describe the need to consider how file formats affect the ease and effectiveness with which scientific and engineering data may be stored and accessed in long term archives. We identify a number of attributes of file formats that can help or hinder them as candidates for long-term digital preservation. We consider how these attributes appear to a number of different audiences for long-term archiving.

Categories and Subject Descriptors
E.2 [**Data Storage Representation**]; E.5 [**Files**]: Organization/Structure.

General Terms
Documentation, Performance, Design, Economics, Reliability, Security, Standardization, Verification.

Keywords
Formats, long-term archive, digital archive, scientific digital libraries.

1. INTRODUCTION
Every day, many terabytes of digital scientific and engineering data are collected and stored in computer files. Much of this data is important not only for current applications, but for uses decades or centuries into the future. For example, the Earth Observing System (EOS) [3] collects data for the purpose of understanding global climate change, data that may be even more valuable in fifty years than it is today. Specifically, the EOS now collects more than two terabytes of information per day from just two missions, known as Terra and Aqua. The system has already amassed two petabytes of data, and will include 15 petabytes before the program is completed.

It is understandable that those who produce, manage, and use scientific and engineering data tend to focus their concern primarily on the short- and intermediate-term uses, and users, of the data. After all, these uses provide the primary reasons for collecting the data, and these users are the primary sources of funding for producing, storing, and disseminating the data. Their needs are pressing and immediate. However, to focus on current uses of scientific data without also planning for its long term preservation and usability is to invite the loss of an enormous and costly resource.

One issue that both current and future users of scientific and engineering data must be concerned with is how the data are formatted. Although it is common to use a single format for both current and future uses, such a format may not be optimized for both of these roles. The format needs of near-term data producers, managers and users may conflict with the needs of those who would preserve the data for the long term, as well as those of users in the distant future.

This is the nub and the cause of the question that this paper is addressing: What are the attributes of file formats that make them more of less suitable for the long-term preservation and use of data?

This paper is divided into two major sections. Section 2 provides background, including a framework for thinking about file formats and a discussion of the community and institutional forces that can influence decisions about formats. Section 3 describes the attributes of formats that we have identified.

JCDL '03, May 27-31, 2003, Houston, TX.

2. BACKGROUND
2.1 What is a File Format?
We define a file format as follows: a *file format* specifies the organization of information, at some level of abstraction, contained in one or more byte streams that can be exchanged between systems [4]. In this document we focus on formats designed for scientific and engineering data exchange, access, and archiving. We exclude discussion of other types of containers, such as relational databases, although data types that are described here often apply equally in other contexts.

Formats are often classified in two different ways: (1) by describing an interface for accessing and/or transferring data in the format, and (2) by describing the objects and structures that constitute the format.

Formats emphasizing interfaces may be described by an *application programming interface* (API), which enables applications to interact with the data, or by a *graphical user interface* (GUI), which enables humans to interact with the data.

Formats emphasizing data structure definitions may be described using formal data description languages (DDLs), or less formally in a specification document. They may, or may not, include an API as part of their specification. Formats emphasizing interfaces may, or may not, include data structure definitions as part of their specification.

2.2 The OAIS Reference Model

As a framework for discussion, we use the Open Archival Information System (OAIS) reference model [8], developed by NASA's Consultative Committee on Space Data Systems. The OAIS reference model identifies archival information preservation functions, including "ingest, archival storage, data management, access, and dissemination,… the migration of digital information to new media and forms, the data models used to represent information, the role of software in information preservation, and the exchange of digital information among archives."

The OAIS reference model describes three entities involved in archival information systems: producer, consumer, and archive (OAIS). Figure 1 shows a high-level view of the flow of information among these three entities. The "information packages" refer to the containers that hold both the content (e.g. scientific data in a particular format) and preservation information (information needed to preserve the content, to ensure it is clearly identified, and to understand the environment in which it was created).

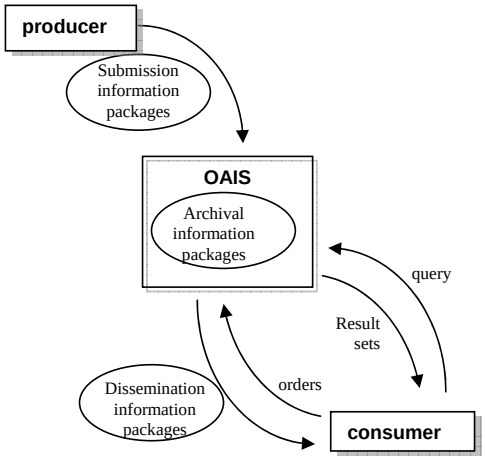

Figure 1. OAIS Reference Model Basic Structure.

2.3 Community Perspectives

The three entities in figure 1 represent communities that can have very different responsibilities and perspectives.

Data producers sometimes have no responsibility or interest in long-term preservation. Instead, producers may be more concerned with making the data available as quickly as possible and in a form that is most usable by its immediate users. This may mean that producers need file formats that handle I/O efficiently, often at rates that are as close as possible to raw I/O throughput.

Archives include "active archives" and "long-term archives." *Active archives* serve the community of users who have immediate or intermediate-term use for the data. *Long-term archives* are primarily responsible for preserving data for future generations.

Data consumers include both active archive users and long-term archive users. *Active archive users* often have a good understanding of the instruments or other methods used to collect the data, as well the context in which the data has been collected. They are likely to have access to, or knowledge of, the hardware and software used to collect, manage, store, and access the data.

Active archive users' needs typically include efficient search and subsetting, requiring complex file structures that may be difficult to understand and maintain. They need files to be formatted in ways that are compatible with existing software for data analysis and visualization. Much of this software may be transient, such as commercial, off-the-shelf tools for data editing and analysis.

The needs of *long-term archive users* to some extent are unknown at the time when the data is produced. They are not likely to have first-hand information about the instruments or methods used to produce the data, and the context in which the data was produced may be long forgotten, or perhaps squirreled away in a notebook or other archived document. These users also have to be concerned about the integrity of the data, which may have been in storage for decades, may have been processed and re-processed many times, and may have been migrated through several different storage media. The hardware and software that were

available at the time when the data were produced may be long gone, and even the programming language used to produce the software may be a thing of the past.

Whereas the active archive consumer may be interested in data produced today, last month, or last year, the long-term archive consumer may need to access data that were produced decades ago, perhaps by an evolving series of instruments and algorithms, with the hardware, software systems, and storage media evolving throughout the production process.

2.4 Environmental Forces

These parties to the discussion of data formats do not converse in isolation. Rather, they all experience different kinds of environmental forces that act on different time scales:

- *Price/Performance Curves and Technological Innovation*, which lead to system changes on time scales of three to five years.
- *Sociological Factors*, which lead to differences in terminology across science and engineering domains and subdomains, and to compartmentalization of data producers and users from data centers and archivists, likely on time scales of five to ten years.
- *Organizational Factors*, which lead to competing approaches to issues of centralization and standardization. On the one hand, standardization and centralization are viewed as reducing duplication of services (or data). On the other, centralization and standardization may introduce bureaucratic inertia and overhead that stifles innovation and makes it difficult to incorporate revolutionary approaches to cost-effective operation. These two organizational factors may also make it difficult to adapt data operations to the needs of niche communities. Organizational factors seem likely to operate on time scales of five to ten years.
- *Institutional Changes* in emphasis and budget, which lead to pressures to delete "obsolete" or unused data, to automate as much of the system migration as possible, and which may lead to reductions in user demand for data. Institutional changes seem likely to be among the longest time scale factors that influence long-term preservation, operating over time scales of perhaps a decade or longer.

2.5 Data Access Patterns

Detailed studies of data access patterns [2] suggest that data distribution can be characterized as falling into three categories:

- *High-Volume Data Users*, who need very large collections of homogeneous data and who typically account for more than half of the data distributed from an archive;
- *Moderate-Volume Data Users*, who need large collections of data distributed in a clumpy fashion through the archive and who typically account for one-third to one quarter of the data distributed from an archive;
- *Low-Volume Data Users*, who typically need only one or two files at a time, ordering data relatively infrequently, and who typically account for one-tenth or less of the data distributed from an archive.

These patterns also suggest that different users would place different emphasis on the format attributes we identify below. High-Volume data users are likely to desire high I/O efficiency, whereas Low-Volume users are likely to desire ease of subsetting.

2.6 Market Friction Forces

The diversity of user communities also creates pockets of users who are familiar with particular data formats, such as those used in Geographic Information Systems. The users in these pockets may well have large investments in formats that they are familiar with, investments in software that has been developed for managing and analyzing data in that format, in commercial licenses for the formats that they are familiar with, and in time to become familiar with the tools for working with data stored in the format. This investment serves as a barrier or source of "market friction" to changing to a new format.

While proponents of a new format may expect that appeals to "reducing the cost of duplication" will serve to rally users to new software, they are likely to underestimate the cost of reeducation and lost productivity on the part of data producers or data users. In many cases, it may be more cost effective to develop specialized staff at data centers that can translate between the archive format and one that is familiar to a niche user community than it is to engage in multi-staff-year negotiations with that community.

Another difficulty in establishing high data access capability for users lies in the rapid evolution of information technology. Scientific and engineering data users often barely have time to keep up with their fields of research and professional activity. They do not have time to keep up with IT developments. For example, even though most computer science students are familiar with Java and databases, most scientists are likely to be familiar with FORTRAN. In the specialized areas of computer system architecture, approaches that worked for scientists in the past may not be well adapted to obtaining high performance on commodity equipment. Data center staff may have to become expert in mediating between the needs of data users and the techniques required for modern computing systems.

3. WHAT MAKES A GOOD ARCHIVE FORMAT?

There are a number of facets to a good archival data format. In the following subsections, we identify six major categories and group file attributes into them. The six categories are

- Ease of archival storage
- Ease of archival access
- Usability
- Data scholarship enablement
- Support for data integrity
- Maintainability and durability

3.1 Ease of Archival Storage

Several attributes of file formats affect the extent to which archives can store data effectively and efficiently. Many archives may be expected to increase in size over time. For such archives, format attributes that can improve scalability are important. These attributes include compactness, and the ability to store large amounts of data in a single file.

1. Compactness

As data sets become larger, it is important to minimize storage and I/O costs. The more compact a file is, the lower its archival

storage cost and the faster it can be written or read. Compactness is affected by the overhead needed to describe the data and data structures in files, by the compactness of the representations used for data values, and by the degree to which the format supports data compression. Good archiving formats are characterized by low overhead, compact number representations, and support for data compression.

2. Size

It is not unusual today for scientific data objects to be many gigabytes in size. It also seems likely that the sizes will increase in the future. If a file format cannot accommodate an entire object, then the object may have to be broken into several pieces, which can increase the difficulty of managing the object.

Access to digital objects, especially objects stored on tape, necessarily incurs some overhead due to latency. One way to overcome this latency is to transfer data in large blocks. Files that can be very large must support this strategy.

3. Ability to aggregate many objects in a single file.

There is usually a limit on the number of objects that can be stored in an archive. The names or identifiers of these objects are typically stored in a database known as the *nameserver*. Name lookup or "name resolution" is required to retrieve particular objects. This name resolution process is widely recognized as a performance bottleneck in a mass storage system. Thus, a file format that supports the aggregation of many digital objects in one file can enable an archive to maintain as small an archive "name space" as possible.

3.2 Ease of Archival Access

In [2] three user categories are identified: High-Volume, Moderate-Volume, and Low-Volume data users. These different types of users place competing demands on archives. The handful of High-Volume Data Users may be well served by continuously streaming, sequential access to the archive, perhaps with data distribution on media (or by storing it on removable hard drives). Alternatively, the data centers may need to develop specialized data access capabilities that can provide high bandwidth access for data investigations involving massive data collections. The few dozen Moderate-Volume Data Users are probably best served by moving the data to large caches and distributing the data by an appropriate combination of media and ftp delivery. The Low-Volume Data Users are probably best served by creating interest-based caches that can deliver data over the network by ftp.

4. Raw I/O efficiency

High-volume data users, such as those who would mine archival data, or those who need to re-process all the data in a large multi-year collection of data, need efficient raw I/O from an archive. For this kind of application, formats that are organized for fast sequential access are probably preferable. Formats that aggregate many large objects in a single file can also be beneficial for this kind of application.

5. Ease of subsetting

Although some users need all of their data in a given container, it seems more frequently to be the case that archive users need to process only parts of data files. For instance, it is not uncommon to search through the metadata among many different objects in order to identify those of particular interest for further processing.

Since archives are frequently stored on sequential media, such as tape, such partial access can be very inefficient if the entire file must be accessed before any information can be retrieved from it. Stream-oriented data files are files that can be interpreted sequentially. Typically, such files have header information at the front of the file that provides the metadata needed to interpret the rest of the file. Searches can often be done by reading this header information.

On the other hand, we have seen that it can be advantageous from a storage perspective to aggregate many large objects into a few files. Thus, for users who need only partial access it is desirable for the file formats to be organized in ways that facilitate extraction of small objects or subsets of objects. For API-based formats, this means that the access library should be able to extract small objects or subsets of files and create new, much smaller files that can be distributed to the user.

A special case involving subsetting involves the retrieval of portions of files containing irregular objects whose spatial extent is much smaller than the full size of the region in the file. For example, in the context of the current NASA data holdings of Earth observations, fires often cover areas that are tens of thousands of square meters, whereas a typical image covers four million square meters. To put this requirement more carefully, the file format needs to support efficient extraction of irregularly-shaped subsets of array elements – and perhaps similarly shaped subsets from several arrays.

3.3 Usability

Data users are, of course, the reason we create data centers and archives. Thus, the formats must serve these communities as well as those of the data producers. We can identify the following attributes that seem likely to be of primary importance to data users.

6. Popularity

A format that is widely used is more likely to have either commercial or Open Source readers available. Wide use also increases the pool of users who understand the format and can write software for accessing and interpreting it.

At the same time, it is important to recognize that user communities often consist of niche markets. Even if a format is in wide use, a particular user who has no one familiar with the tools or details of its use is likely to face a very difficult learning curve.

7. Availability of readers

One way to maintain ease of data access is to ensure that readers are available for accessing archived data files. If the format is not simple (or if the original language in which it was written has become extinct), resources may have to be devoted to maintain and evolve readers over time.

Readers for a file format must be readily available and supported on a wide variety of platforms. Most modern mass storage systems are distributed, in the sense that user's machines connect directly to the mass storage system and download the file in the archived format. Users then expect the file format access software to reside on their native platform so that they can

directly begin to use the downloaded file. This implies that client-side file format access tools must be ubiquitous and widely supported.

A related issue has to do with general-purpose formats that package collections of files. In today's environment, many mass storage systems encourage users to store their data using the UNIX tar file format. Support for the UNIX tar command is taken for granted, since this utility is maintained and supported by UNIX operating system manufacturers. At the same time, the widespread use of Wintel machines may make it difficult for some users with commodity computers to access data if only tar format unpacking is available. In the future, archives will probably need to support file format software that can handle both of these packaging formats.

8. Ability to embed data extraction software in the files

An alternative to requiring that users have readers at their sites is that the files come with read software embedded. This kind of capability is widely used in the Wintel market today. Users get a self-extracting file that installs itself after downloading. With this approach to distributing data, data centers and archives might be able to reduce the time and staff they have to devote to helping niche users build their own readers.

An intermediate possibility is to ensure that Open Source groups provide software that can read the files. For example, the GNU Image Processing library (GIMP) [5] might be seeded with functions that can translate data in archive formats into the formats used in the library.

9. Ease of implementing readers – simplicity

If readers are not available for a particular file format, but the file format is simple, it may be easy to write readers from scratch. For example, if a raster image file consists of a header describing the dimensions of the image, followed by a byte stream that contains the image pixels, a reader could easily be written for this simple format.

Factors that mitigate against simplicity include the variety of object types the format can support, how flexible it is in permitting objects to be mixed, whether it is a single-platform or a multi-platform format, the amount of self-description it supports, the variety of storage options it supports (e.g. compression, tiling), and the variety of datatypes it allows.

10. Ability to name file elements

Hard-core programmers often don't mind getting down to the bit level. Such programmers are used to working with files and extracting data by using a byte offset from the beginning of the file.

Data users may not experience this need. For one thing, it is error-prone and wastes their time (or the time of their staff). For another, this approach is very machine and language dependent. For a third, it forces users to learn things that are irrelevant to their desired data use. Finally, it discourages abstraction.

Accordingly, one of the desirable attributes for file formats for digital libraries is the ability to work with data based on manipulating the element names instead of binary offsets, or other references that require knowledge of the machine on which the data were written or of logical models that are equivalent to knowing the details of some "virtual machine".

3.4 Data Scholarship Enablement

While there is generally a community that will use data shortly after it has been produced and validated, there are also communities that need long time series of measurements. The "data scholars" in this community have some special needs that we identify in the next set of attributes.

11. Provenance traceability

Data scholars definitely need the ability to trace the entire configuration of data production - based on information in the files and in the documentation of how the data were produced. Of course, a data scholar's ability to track back in time depends on items external to the file – the format should not bear the entire responsibility for maintaining provenance.

12. Rigorous definition

Since data in an archive may not be accessed until many years after being written, it is very possible that the format expertise that was available when the data were originally stored will be required when the file is accessed again. If there are no readers available when access is desired, or if readers need to be rewritten to accommodate new technology or with new functionality, it is important that the *format* be defined in a sufficiently rigorous way so that readers can be written that correctly interpret the contents of data files.

Even if the format is fully and rigorously defined, it may not be easy to interpret or give meaning to the data and metadata in an archived file. In the fortunate instance that a rigorously defined *API* is available, the chances of success in creating a new reader increase markedly.

In the even more fortunate instance that *library source code* is available for accessing the format, and that source code is rigorously defined, then the capability to maintain readers over time for accessing the format will be greatly enhanced.

It is also important that the API and the library be adaptable to changes in software technology. An API that is tied to C or to FORTRAN, or to a particular operating system, may be hard to adapt to new software paradigms.

13. Self-describing

Many different types of metadata are required to decipher the contents of a file. These include:

- *Structural metadata*: metadata about the structure of the file, the data structures contained in the file, the representation of data in the file, and storage methods use in it, such as chunking and compression.
- *Application-specific metadata*: information about the contents of the file.
- *Additional features to enhance long-term survival*: standard time/date stamps, standard software version identifiers, checksums or error correction items.

In an archived file, there is a danger that this information can be lost over time unless it is included within the file itself. Thus, the extent to which a file is *self-describing* is one measure of its suitability as an archive format.

Comment [MF2]: I think this belongs in the section where we talk about readers, but don't have time to move it there. My reasoning is that this has less to do with information that data scholars need than it does with just accessing the data.

Comment [MF1]: Here we get into the important issue of API vs. format. In a sense, we're talking about object-level access. In the next version, I would like to expand it and emphasize the potential complexity of the format, not just the problems of dealing with different machine architectures.

14. Citability

In many applications, we need to be able to identify the individual data elements within files. For example, we need the ability to identify the particular pixels within an image that belong to a hurricane, a smoke-plume, or the particular atmospheric columns that are over a selected ecosystem [1]. We do not want these references to depend on the machine that wrote the data or on the particular data center that produced them. In other words, we want files that will give us a machine-independent ability to reference or 'cite' the individual data elements in a stable way.

15. Referential extensibility

If we can cite particular data elements within files, then we also have the ability to build annotations about new interpretations of the data - and to preserve those annotations. This gives us the ability to create indexes of interesting phenomena that are external to the original files.

16. URN embedding capability

In addition to being able to cite individual objects within files, we should ensure that files could reference external documentation or link with other files.

3.5 Support for Data Integrity

With the increasing requirements for information security, we need data formats that explicitly support checks of data integrity – the notion that we can vouch that the data came from a reliable source and that it has not been irretrievably corrupted. If the data also becomes important for legal work, it will be critical to be able to demonstrate an acceptable chain of evidence. This leads to three format attributes that are important in maintaining this chain.

17. Source verification

Because of e-commerce and the legal use of digital data, inclusion of a digital watermark or signature is becoming an important part of security and trust relationships. Some care is needed in this area because digital data files can be quite large, meaning that cryptographic encoding of files or digital watermarks must be created without overburdening the data centers or archives. In many cases, the data cannot be altered even if it is encoded. Some research may be needed to develop appropriate technology in this area.

18. File corruption detection

Being able to detect that a file has been corrupted is a slightly different attribute than having it contain verification that the file is being obtained from an appropriate authority. Corruption detection is useful not only for protecting against malicious actions, but also against unintended changes in the data, such as that caused by faulty equipment.

In academic research that has been done on preventing document loss, considerable effort has been expended on approaches that allow replicated data to be compared to verify that the data has not suffered modification [9].

For scientific data, it is possible that sites might modify replicated data to enhance access for new kinds of data users. One example would be if the original data producer created files in temporal order, but in which the replicant site reordered the data to enhance spatial access. If the original data were corrupted, the mirror site might need to have the ability to reconstruct the original from the reordered data.

19. File corruption correction

With additional work, we may be able to find ways of using error-detection approaches to provide error-correction as well. This is likely to be an important development if the correction can be done at a reasonable computational cost.

Comment [MF3]: This section feels a little funny. There are other ways to get what we want without embedding the information in a format, so we probably ought to argue the pros and cons of having this in the format. Later.

3.6 Maintainability and Durability

In the truly long-run, file formats must allow data archives to be cost-effective. In other words, data formats must help automate archival operations (or at least not hinder automation). It is also true that we need stable institutional arrangements in order to achieve reasonable long-term archival at all. We identify attributes of file formats that have a bearing on maintainability and durability of data access.

20. Long-term institutional support

Institutional support helps ensure the long-term maintenance and support of a data format by placing responsibility for these operations on institutions rather than individuals or projects. If the format is an open standard, then the possibility is opened for an entire user community to support the standard, which could result in a more enduring level of institutional support. On the other hand, an open, free standard runs the risk of not having any support at all if there is not a champion for that format.

21. Suitability for a variety of storage technologies

It is virtually certain that storage technologies will change dramatically in the coming decades. We are already seeing a heavy use of emerging technologies, such as write-once media, systems with 128-bit virtual address spaces, and diskless desk-top systems. What about technologies we don't even know about? It may be that the most important criterion here is that the format not be geared toward any particular technology.

22. Stability

If a format changes often, there are likely to be problems in compatibility between versions. If these changes occur during the time that the files are being archived, then the archive is likely to have many different versions of the format, increasing the cost and complexity of access software. Multiple versions may also introduce confusion regarding the information content of different files in the archive.

Of course, examination of real archives suggests that there will be no utopia in which all of the files in a large archive will be in the same format. The probability of this happening is probably about the same as having Microsoft Word, Version 25 read Word Perfect, Version 1 successfully. Thus, one practical set of issues for archives is if (and if, when) files in an old format should be converted to another format.

Comment [MF4]: This gets again at the issue of object model compatibility.

On a still longer time scale, archives confront issues related to loss of community knowledge and standards. One of us has already faced the situation in which a data set appears to be unreadable – but still received a recommendation from his user working group to keep the data in its electronic form.

23. Formal (BNF- or XML-like) description of format

In the situation where languages die or communities that produced data disappear, archives may need to retain the ability to create new readers solely on the basis of formal descriptions of the file contents. Considerable time can be saved in this situation if the file format is derivable from a formal description.

At the moment, it appears that eXtensible Markup Language (XML) would be the language of choice. However, in the not so distant past, there was Object Description Language (ODL), as well as other forms of formatting language.

24. Multi-language implementation of library software

One defense against language obsolescence is to have multiple implementations of readers for a single format. The different implementations can be done on different kinds of systems and in different computer languages.

Multiple implementations may appear to be duplicative, but it has at least two useful side-effects. First, it provides increased opportunities to verify correctness of the formatting software. Second, it increases the breadth of support for the format itself, making the software available to different communities of users. In general, having a format implemented in a second language is roughly equivalent to having a "second source" for an industrial component.

25. Open Source software or equivalent

Businesses seem to be more mortal than people – or at least the life-expectancy for people is longer than it is for most businesses. In addition, the changing legal environment for information technology suggests that care is required for data centers and archives that have substantial investments in proprietary software for any part of their infrastructure.

Both of these considerations suggest that data centers and archives need to move toward obtaining Open Source arrangements for all parts of the file format and associated libraries. If proprietary formats are used to distribute data, then later business developments may place data access at risk. A prudent strategy would appear to be to move toward an entirely non-proprietary infrastructure

4. CONCLUDING REMARKS

It is perhaps useful to remember that in terms of long-term, persistent storage of digital data, we are at a very early stage of development compared with the experience of keeping text; cuneiform, clay tablets and written manuscripts on papyrus, parchment, and paper have a far longer history. Even with printed text, it has taken almost four hundred years to arrive at such text conventions as chapter headings, numbered sections, and even pagination. To the extent that textual formats provide any indication of what we may expect for data formats for binary data, it would seem reasonable to expect considerable variation within a slowly varying evolution of "generally accepted practice". Donald Knuth's history of mathematical typography [6] provides a useful summary of this evolution. Perhaps in the really long term, aesthetic and formal considerations will govern. Knuth comments that "Mathematics books and journals do not look as beautiful as they used to. It is not that their mathematical content is unsatisfactory, rather that the old and well-developed traditions of typesetting have become too expensive. Fortunately, it now appears that mathematics itself can be used to solve this problem." [6, p. 1] To translate this advice to data formats, perhaps the most useful way to improve data formats for long-term, persistent access is to place their structure within a rigorous mathematical structure, where a prescription akin to XML would allow optimization of the format with respect to the properties we have suggested in this paper. As the saying goes, such a prescription lies "beyond the scope of this article".

Comment [MF5]: Do you mean DTD or schemas, rather than XML? I see XML as a metaformat, somewhat like HDF and netCDF, with a well-defined implementation. Indeed, I think XML might be the basis for a really good long-term archive format.

5. ACKNOWLEDGMENTS

The authors are grateful for support provided by NASA's Earth Science Enterprise, by the National Science Foundation under grant, NARA NSF 02-02 GPG, and by a Cooperative Agreement with NASA under NASA grants NAG 5-2040 and NCC5-599. B. R. B. is also grateful for support provided by NASA Langley Research Center and the Atmospheric Sciences group within which the Atmospheric Sciences Data Center resides.

6. REFERENCES

[1] Barkstrom, B. R., 1998: Digital Archive Issues from the Perspective of an Earth Science Data Producer, paper presented at the *Digital Archive Directions (DADS) Workshop*, June 22-26, 1998, available at http://ssdoo.gsfc.nasa.gov/nost/isoas/dads/dads21b.html.

[2] Barkstrom, B. R. Using Markov models and innovation-diffusion as a tool for predicting digital library access and distribution, in Proceedings of the first ACM/IEEE-CS joint conference on Digital libraries (Roanoke VA, June 2001), ACM Press.

[3] The Earth Observing System Project Science Office homepage is http://eospso.gsfc.nasa.gov.

[4] Folk, M. J., Zoellick, B., and G. Riccardi. File Structures: An Object-Oriented Approach with C++. Addison-Wesley, Reading MA, 1998.

[5] See the GNU Image Manipulation Project (GIMP), available at http://www.gimp.org/

[6] Knuth, D. E. TeX and Metafont: New Directions in Typesetting, American Mathematical Society and Digital Press, Bedford, MA, 1979.

[7] Musick, R. and T. Critchlow, 1999: Practical Lessons in Supporting Large Scale Computational Science, Lawrence Livermore Report UCRL-JC-135606.

[8] "Reference Model for an Open Archival Information System," CCSDS 650.0-R-2 Red Book, July 2001.

[9] Rosenthal, D. S. H. and V. Reich. Permanent Web Publishing. http://lockss.stanford.edu/freenix2000/freenix2000.html