E-JOURNAL ARCHIVE
DTD FEASIBILITY STUDY

Prepared for the

Harvard University Library
Office for Information Systems
E-Journal Archiving Project

By

Inera™ Incorporated

1 Table of Contents

2 Introduction

Research libraries and journal publishers are concerned with the issue of long-term archival deposit, storage and retrieval of information published in journals. The Harvard University Library Office for Information Systems is conducting a review of issues related to the archive of electronic journals. Harvard's mission statement for the archive project is:

> *The archive's purpose is to preserve the significant intellectual content of journals independent of the form in which that content was originally delivered in order to assure that this content will be available to the scholarly community for the indefinite future. Functionally, the archive is designed to render text and still images and other formats as practical with no significant loss in intellectual content. The archive reserves the right to freely manipulate the internal format of the manifestation over time as long as the plain meaning of the intellectual content is preserved.*

Many journal publishers create their own SGML archives today, however these archives are built with proprietary DTDs that are designed by each publisher for its own purposes. Such archives, while serving the needs of individual publishers, do not meet the long-term requirements of archivists because multiple proprietary DTDs create a Tower of Babel rather than a consistently accessible repository.

Converting publisher-specific archives to an independent form for scholarly archives faces two key questions. First, can a common structure (DTD or Schema) be designed and developed into which publishers' proprietary SGML files can be transformed to meet the requirements of an archiving institution? Second, if such a structure can be developed, what are the issues that will be encountered when transforming publishers' SGML files into the archive structure for deposit into the archive?

To address these questions, the Harvard University Library under a grant from the Mellon Foundation commissioned this report to address the feasibility of developing a common structure that can be used to reasonably represent the intellectual content (text, tables, formulas, still images, and links) of archived journal articles.

Ten publishers were asked to provide their DTD, documentation and sample SGML documents. Inera reviewed these materials to determine if such a structure can be developed and to assess the challenges that will be faced in SGML transformation. We also examined the challenges faced by organizations that have worked with DTDs from multiple publishers, and based our recommendations on Inera's experiences implementing SGML systems for a wide range of journal DTDs.

This report summarizes our findings and provides high-level recommendations for the development of a common structure. It examines challenges that will be faced in content transformation and suggests strategies to address them.

2.1 Terms and Definitions

This report assumes that the reader is familiar with SGML and XML. For this reason, we will not provide a definition of markup language terminology or other commonly used technical terms in the journal-publishing world nor will we provide a detailed explanation as to the importance of markup languages for journal publishers.

However, there are certain terms that we will define in order to be clear and concise, and to avoid ambiguity in meaning.

Standard Generalized Markup Language (SGML): We use the term SGML generically in this report to refer to content tagged in a markup language (either SGML or XML). We use "SGML" rather than "XML" because the majority of publishers surveyed produce SGML today, not XML. In cases where the distinction is important, reference may be made to XML.

Document Type Definition (DTD): A DTD defines the structure of an SGML or XML document. A newer standard, XML Schema, has recently been adopted as a W3C recommendation, allowing XML structures to be defined with DTDs or Schemas. We use term "DTD" rather than "Schema" through this report because all surveyed publishers currently use DTDs rather than Schemas. In cases where the distinction is important, reference may be made to a Schema.

Archive DTD: A DTD into which SGML from a wide range of journal publishers can be transformed such that the resulting SGML preserves and can reasonably render the intellectual content of journal articles. Although we use the term DTD, the final structural definition may be a Schema rather than a DTD. The recommendations made in this report are not altered by the use of a Schema versus a DTD.

Delivery Envelope: SGML files are one part of an electronic journal issue. The complete issue contains SGML files, PDF files, graphic files, and other metadata about the issue. The Delivery Envelope is the set of files required to capture an entire issue electronically and it includes the organization of those files according to the requirements of the publisher or the archive.

Generated Text: Inconsequential, formulaic, or stereotypical text and formatting omitted from an SGML file, which is applied to content by a style sheet when an SGML file is rendered. The style sheet generates text based on the structure information provided by the markup elements and attributes.

Boilerplate Text: Inconsequential, formulaic, or stereotypical text and formatting in an SGML file that could have been omitted and applied as Generated Text by a style sheet. Some publishers use Boilerplate Text rather than Generated Text to avoid the need to archive style sheets.

Metadata: Literally, "data about data". In the context of this report, we use the term to indicate data about the article (e.g. the publisher's article ID) that appears in the front matter of the SGML file and may not appear in print.

Supplementary Content: Files external to articles that are not part of the printed publication of an article. Typically they are multimedia files or files from other PC applications such as Microsoft Office. Supplementary Content excludes still images printed as part of the article.

Related Document: A journal item whose content is directly related to another journal item. For example, an erratum is related to the journal article that it corrects, or a reply is related to a letter. Articles mentioned in "In This Issue" summaries are also considered Related Documents.

When working with translations from one context to another, whether a foreign language translation or an SGML translation, some information is likely to be lost. To describe the degree of loss, we will use the following terms:

Loss-less Conversion: A conversion of SGML from one DTD to another in which the full semantic content of tags and attributes applied with the first DTD is completely preserved in the markup applied by the second DTD. If a conversion is truly loss-less, you can convert from DTD A to DTD B and back to DTD A, and the final document in DTD A is identical to the original document. This can only happen if DTD B contains all of the semantic elements and attributes of DTD A.

Lossy Conversion: A conversion from one DTD to another in which some degree of resolution is lost.

Tolerable Loss: A lossy conversion in which the loss of resolution is not significant relative to the goals of the organization for which the conversion has been done.

Significant Loss: A conversion in which information is lost that is critical to the intended use of the converted document.

Reasonable Representation: A rendering in which the text, tabular, formulaic, still image and link data comprising the intellectual content of a document is represented without significant loss. The manifestation may have tolerable loss of semantic information that does not affect the interpretation of the intellectual content.

Journals contain several different kinds of content. We loosely group these content types into four classifications:

Article: A research article or other primary journal content, such as a review or short item (e.g. short communication) that adheres to the same structural format as an article.

Non-article Content: Significant journal content that is not an article. Examples include correspondence, editorials, book reviews, errata, etc.

Other Content: Journal content that is not of significant research importance. Examples include abstracts of articles published elsewhere, news items, calendars, announcements, etc.

Static Content: Journal content that does not change from month to month such as the masthead or instructions to authors. Static content is one type of Other Content

2.2 Scope

This study examines the structure of SGML in publisher files, and how that SGML can be transformed to a common structure for archival deposit. The study scope includes both Article and Non-Article Content.

Policy recommendations for application of the structure are provided. These policies give guidance on interpretation of the structure, especially regarding content that should (or should not) be placed between the tags. These policies are directed to organizations responsible for transformation of publisher SGML into archive XML, a transformation that may be completed by the publisher or the archive, depending on the specific circumstances of deposit.

In addition, this study reviews the technical and quality issues that may be encountered by either a publisher or an archiving library when converting SGML into archive XML according to an archive DTD. Business issues and imperatives for completing the conversion are beyond the scope of this report.[1]

Each publisher has unique standards for delivery of E-Journals. We refer to these standards as the delivery envelope. This study excludes examination of the delivery envelope. The delivery envelope should be examined in a separate study that focuses on design of the delivery envelope for archive XML deposit by publishers. The content of the Delivery Envelope that should be submitted to the archive by publishers will depend on specific storage and retrieval requirements of the archive and is outside the scope of this study.

This study is only concerned with the archive of final article content. In the traditional print publishing model, libraries have primarily been concerned with the archive of the final printed version of an article. In the rapidly growing world of online peer-review systems and electronic pre-publication, multiple versions of an article may be available. The issues surrounding archive of multiple versions are outside the scope of this study.

Caveat: Every publisher has unique needs they must address when adopting or designing a DTD. The recommendations in this document are designed to meet the specific needs of an archive DTD as outlined in the Harvard mission statement.

[1] We recommend reading the report *Standards for Electronic Publishing*, Mark Bide and Associates, which was written as part of the NEDLIB project. It presents a lucid discussion of many business issues and imperatives regarding the creation of SGML by publishers. It can be accessed at http://www.kb.nl/coop/nedlib/results/e-publishingstandards.pdf.

2.3 Methodology

Harvard and Inera selected ten DTDs for review based on one or more of the following criteria:

1. A significant volume of content has been captured in the DTD.
2. Content captured in the DTD is widely cited.
3. The DTD is used by multiple publishers, possibly for content interchange[2]

The exclusion of publishers from the list does not mean they failed to meet the selection criteria. Rather, the scope of the study was limited to complete this report in a reasonable time period. The goal was inclusion of a sufficient number of DTDs to allow most significant issues to be identified during the course of the study.

All publishers asked to participate in this study accepted. They are:

- American Institute of Physics (AIP)
- BioOne (BioOne)
- Blackwell Science (Blackwell)
- Elsevier Science (Elsevier)
- Highwire Press (HWP)
- Institute of Electrical and Electronics Engineers (IEEE)
- Nature Publishing Group (Nature)
- Pubmed Central (PMC)
- University of Chicago Press (UCP)
- John Wiley & Sons (Wiley)

All participating publishers were asked to submit the current version of their DTD, DTD documentation, and twenty to twenty-five sample document instances from multiple journals and issues. Materials were delivered to Harvard during September and October 2001.

Inera conducted analysis of the materials. The authors also drew on their experience implementing SGML systems for electronic journals to highlight specific issues and problems.[3]

Italic text is used in this report for quotes and Inera's recommendations.

[2] Highwire Press and Pubmed Central are not traditional publishers, however their role in the electronic journal publishing provides some important insights into the issues faced by archivists. Both organizations accept documents in a number of different publisher's DTDs and convert the content to their own DTDs. The use of their DTDs for content interchange illustrates many of the conversion and quality issues faced by an archiving institution. See Section 10 for more details.

[3] Since 1995, Inera has developed systems for the delivery of SGML and XML according to a variety of DTDs used in scientific, technical and medical publishing including the Blackwell, Elsevier, Capital City Press, and Highwire DTDs.

3 Structured Documents

This section briefly reviews the history of electronic journal production with SGML and XML by participating publishers.

3.1 Why SGML?

Traditionally journals were available in one manifestation: print. With the increased use of computers and the explosion of the Internet, however, journal articles are now produced and published in multiple manifestations. In addition to print, articles are available electronically in PDF, SGML and HTML formats.

SGML and PDF each have advantages and disadvantages. While it is beyond the scope of this report to review the differences in detail, some key points are:

- SGML retains the structural information about the content of a document. Because document elements are semantically identified in a precise and granular manner, new information can be discovered from the tagged elements.

  For example, most publishers tag the elements in bibliographic references. With this structure, it is quite easy to create links to Medline, CrossRef, or other databases that may appear in the future. Without this semantic markup, which is not retained in PDF files, such links cannot be created to databases unless the reference is parsed with sophisticated pattern matching.

- PDF retains the full visual presentation of the original article. While it is possible to create PDF files directly from SGML input, they may not match the original PDF without some manual typesetting. If one's only need is to read the article content, PDF is a much handier and easier format to use than SGML.
- SGML is a "human-readable", non-proprietary format. By human readable, we mean that a person can open an SGML file and see understandable content and structure rather than completely unreadable computer code. By non-proprietary, we mean the format is an international standard that is publicly documented. In contrast to SGML, PDF is a "binary" format that is impossible for a person to read without Adobe's software. The PDF format is proprietary, owned and controlled by Adobe.

From an archivist's standpoint, SGML presents two key advantages over PDF:

1. Because SGML is human-readable and non-proprietary, there is a higher probability that files will be usable in the time frame of the proposed archive (75 or more years). There is no guarantee that Adobe Acrobat (or other software that can render PDF files) will be available for the computers in use at a distant time in the future.
2. Because SGML retains content structure, it will be possible to index and search the article content, and to construct links to databases that do not exist today.

For these reasons, the focus of this report is archive of SGML files. However, Inera recommends that if space and logistics allow, PDF files also be included in the archive for the following reasons:

1. PDF files, because they retain the original visual presentation, preserve a different aspect of the print journal, and therefore may be more useful for some research activities.
2. Some Non-Article and Other Content is available only in PDF format. If the archive does not accept PDF, this content will be lost.

3.2 SGML and DTD Design

This section provides a short review of SGML and DTD design and implementation philosophy and how it impacts the SGML of surveyed publishers. Extensive background on SGML and the reasons journal publishers have chosen to use it is beyond the scope of this study.[4]

[4] For an extensive discussion of this subject, see NEDLIB, *Standards for Electronic Publishing*, Mark Bide and Associates, at http://www.kb.nl/coop/nedlib/results/e-publishingstandards.pdf.

SGML enables the separation of format instructions, which are often proprietary, and structural information by tagging content for semantic meaning rather than format. Format is then applied to the structural elements with a style sheet when the content is rendered.

Steve DeRose comments, "Strong separation of formatting from structure is the hallmark of good SGML use."[5] In an ideal SGML application, a complete separation of formatting and structure will be preserved. In journal publishing, the degree of separation varies by publisher. Some publishers, such as Elsevier, follow DeRose's philosophy:

> *In order to separate structure and presentation one applies the concept of generic markup: generic codes (or tags) are placed around most – or all – elements in a document. These elements could be a paragraph, a title, an abstract etc. The tags usually indicate the structure of the document. They do not indicate the style or format of the document, such as fonts, column widths etc. For each different style a style sheet is required to translate the logical structure into a presentation on paper, for example. The set of tags and their mutual relations comprise the 'generic markup language'.*[6]

SGML created according to Elsevier standards excludes almost all boilerplate text and face markup. To render content and apply generated text, a sophisticated style sheet, which is separate from the SGML document instance, is applied to the SGML content. This separation allows the style of presentation to be modified easily, meeting a key goal of Elsevier's electronic workflow requirements. However, because Elsevier does not archive style sheets with SGML files, the style information must be recreated to render SGML in a new environment.

Other publishers, such the University of Chicago Press, take a different view on the issue of generated text:

> *Our overriding concern in our SGML implementation was to accurately preserve the entire text as published.*
>
> *As an example of this design philosophy at work, consider the issue of generated text. Many DTDs, including ISO 12083, either assume or allow for the possibility that the formatting system will generate text such as counters, labels, or the punctuation and connecting text around a list of author names. However, if one uses generated text, then one must also archive the generation rules with the text in order to accurately recreate the original text. We know from experience that journal styles evolve over time; it seemed to us a much better solution to dispense with generated text entirely.*[7]

The approach of the University of Chicago Press shows a keen insight into the problems faced by archivists. By including more boilerplate text and format information in the SGML file and avoiding generated text, the print version of the article can more easily be reproduced from the archive. Blackwell, in the context of explaining their format for structured references, gives further insight into this issue:

> *Many SGML and XML DTDs consider punctuation to be generated text i.e. the punctuation required is generated by stylesheet rules and is not stored in the document. The disadvantages of relying on stylesheet rules to create generated text are:*
>
> 1. *The XML document is no longer a 'standalone' document*
> 2. *The generation rules need to be stored along with the document throughout its life*
> 3. *The document cannot be read without applying a process which applies the punctuation rules to the XML document*
> 4. *It can be quite inefficient if different rules/templates have to be created to reflect differing punctuation styles across a store of documents*
>
> *Storing the generated text in an x element in the XML document means that the XML fragment can easily be converted to, for example, simple text, a typesetting format or*

[5] Steve DeRose. The SGML FAQ Book. 1997. Kluwer Academic Publishers: Norwell, MA. p. 197.

[6] Elsevier DTD 4.1 Reference Manual (refman.pdf), November 12, 1997.

[7] Evan Owens, SGML and The Astrophysical Journal: A Case Study in Scholarly and Scientific Publishing. http://www.journals.uchicago.edu/sgml96.html, accessed on October 11, 2001.

> *HTML without the need for complicated templates or rules. If existing rules for generating punctuation are already in place, or if more 'abstract' XML is required, then the contents of the <x> element can be ignored and the rules applied.*[8]

The range of boilerplate text and formatting publishers include in SGML varies widely. However, inclusion or exclusion is based on publisher policy rather than DTD design because DTDs are not powerful enough to support rules to enforce most of these policies. In many cases, even schemas may not be able to enforce them.

Even if these policies could be enforced in a DTD or schema, a tightly defined structural framework would become unnecessarily restrictive:

> *A narrowly targeted DTD can enforce some of these restrictions, but a broadly targeted one cannot, since it must be adaptable to different house styles if required.*[9]

The problem is multiplied when a DTD must be broadly targeted to accommodate different house styles.

> *The problem is much more complicated than simply choosing names: authors at the two companies are accustomed to thinking about the* structure *of their information in very different ways, and a DTD that is well suited to one will work very poorly for another. If you are a DTD designer, there are four broad approaches that you might chose in this situation:*
>
> 1. *you can create a DTD that uses a new, neutral structure, different from either of the existing ones*
> 2. *you can impose one of the two structures arbitrarily*
> 3. *you can create a DTD that allows either of the two structures as alternatives*
> 4. *you can create a less-restrictive DTD that can be adapted to any appropriate structure*
>
> *Most industry-standard DTDs use the fourth method more often than the others, because the DTDs need to be useful for a wide range of applications within a single industry; but, as a result, the DTDs provide a lower level of guidance for authors, validation for processing, and context-sensitivity for searching.*[10]

A less restrictive archive DTD will allow easy text rendering and maintenance of actionable links while it provides flexibility required for transformation of SGML from a wide range of publisher files.

Inera recommends a less restrictive archive DTD because it must accommodate a wide range of journal styles. The less restrictive design will allow flexibility to retain boilerplate text and formatting in archive XML files, or for publishers to generate text and formatting via a style sheet prior to the submission of archival data. This approach will reduce the need to archive separate style sheets.

3.3 A Brief History of Journal DTDs

3.3.1 DTD Derivation

All of the reviewed DTDs owe their legacy, directly or indirectly, to the ISO 12083 Serial DTD.

The Association of American Publishers (AAP) originally developed the 12083 DTDs in the late 1980's. They were released as an ANSI standard in 1988 and became an ISO standard in 1993. They were last updated in 1995.

[8] Generated text, white space handling and the <x> element. http://www.blackwell-science.com/dtds/4-0/bpg4-0help/generatedtext.htm accessed on October 11, 2001.

[9] David Megginson. Structuring XML Documents. 1998. Prentice Hall: Upper Saddle River, NJ. p. 135.

[10] David Megginson. Structuring XML Documents. 1998. Prentice Hall: Upper Saddle River, NJ. pp. 114-115.

Table 1 shows the legacy of each publisher's DTD. In some cases, DTDs were directly derived from ISO 12083 and only minimal changes were made. In other cases, the DTD author(s) reviewed the structural foundation of 12083, but built a DTD from scratch using a unique set of elements and attributes.

Table 1: DTD Legacy by Publisher.

DTD	Legacy
AIP	Derived from 12083 Serial DTD
BioOne	Derived from 12083 Serial DTD
Blackwell	Developed by Blackwell Science. Significant earlier versions are 2.2 and 3.0
Elsevier	Developed by Elsevier Science. Earlier versions are 1.1.0, 2.1.1, 3.0.0, 4.1.0, and 4.2.0
Highwire	Derived from Elsevier DTD 4.1
IEEE	Derived from 12083 Serial DTD
Nature	First developed by Alden Press
PMC	Derived from the Keton Full Text DTD, which is based on the Cadmus Journal Services (CJS) 2.1 DTD. The CJS DTD is derived from the Elsevier 3.0.0 DTD
UCP	Derived from AAP Article DTD Z39.59 with modifications based on 12083 Serial DTD
Wiley	Based on the Elsevier DTD 3.0.0, as modified from analysis of Wiley journals

Even those publishers who use 12083 with only minimal modification were unable to use it verbatim. In a 1998 survey conducted by the ISO 12083 working group, the comments of one publisher captured the feelings of many: "It is way too complicated, yet it is not flexible enough to represent the things I need to have in the journal I publish on the Internet."[11]

Publishers have not adopted ISO 12083 verbatim because it is too generalized. It was designed to work for everyone, but it does not really work for anyone. Every organization has specific needs that must be met in a DTD. An archive has specific needs that can be addressed in a DTD designed for archive use. We do not believe 12083 meets the archive mission. However, the creation of the archive DTD should be simplified by the common 12083 legacy shared by all journal publishers' DTDs.

3.3.2 SGML vs. XML

The publishers represented in this survey all started to archive their content in a structured form prior to the creation of XML. Table 2 shows that most publishers use SGML DTDs rather than XML.

Table 2: DTD type, version and last revision date.

DTD	Type	Definition	Version	Last Revised
AIP	SGML	DTD	3.0.2	August 14, 2001
BioOne	SGML	DTD	1.0.1	October 16, 2000
Blackwell	XML	DTD	4.0	October 2000
Elsevier	SGML	DTD	4.3.1	April 2001
Highwire	SGML	DTD	4.2.14	July 2001
IEEE	SGML	DTD	2.0	February 2, 2000
Nature	SGML	DTD	3.29	July 27, 2001
PMC	XML	DTD	1.13	Sept 10, 2001
UCP	SGML	DTD	Version 6	Sept 19, 2001
Wiley	SGML	DTD	3.4	July 10, 2000

Over time, we expect most publishers to switch to XML. This change will occur in part because there is a wider array of tools for XML than SGML. The time frame for the SGML to XML transition will vary by publisher.

All of the publishers use a DTD rather than a Schema to define their document structure. Because XML Schema only became a W3C recommendation in February 2001, and tools that utilize the final schema recommendation are just beginning to appear in large numbers, most publishers have not yet adopted it.

Although most publishers use SGML today, Inera recommends the archive use XML, not SGML, because there is a wider range of tools available for XML. While XML is primarily a subset of SGML, we do not see

[11] ISO 12083 Survey. http://www.xmlxperts.com/survey98.htm accessed on October 4, 2001.

the reduced feature set of XML as a limitation because few SGML-specific features have been used by journal publishers. In cases where SGML-only features have been used, alternative solutions are available.

Because of its complexity, the W3C Schema recommendation was not without controversy.[12] *While Inera believes DTDs will be replaced with a more robust structure definition over the long term, we are not certain the current Schema recommendation is sufficiently stable to be used as the foundation for an archive. For this reason, we recommend deferring the decision of using a DTD versus a Schema.*

3.3.3 DTD Versions and Upgrades

Table 2 shows current version number for each DTD. All of these DTDs have been upgraded over time for a variety of reasons. The most common reasons are:

1. A new element or attribute is added to support a semantic object that had not been encountered previously.

   New elements are especially common in DTDs used for a constantly growing number of journals such as the Highwire or PMC DTDs.

2. Publishers decide a new element or attribute is needed to identify material with greater granularity.

   Usually increased granularity is required for better formatting, searching, or linking. For example, the 4.2.0 update of the Elsevier DTD added elements to tag stereochemical equations and the 4.3.1 DTD added support for cras-terre keywords.

3. Occasionally, publishers decide a DTD element must be re-engineered because of limitations discovered in production use of the SGML.

   For example, in Elsevier DTD 3.0 and Blackwell DTD 2.2, empty elements were used for object citations. The original citation text did not appear in the SGML. Using this model allowed adequate rendering of "See Table 1" when tagged as "`See <tblr id="1">`". It was inadequate for more complex citations such as "See Tables 1-4" which would be tagged as "`See <tblr id="1"><tblr id="2"><tblr id="3"><tblr id="4">`". Rendering this SGML as "See Tables 1-4" was a non-trivial process.

   When Blackwell upgraded from DTD 2.2 to 3.0 and Elsevier upgraded from DTD 3.0 to 4.1, both changed this model in two fundamental ways. First, they changed from a unique element for each type of cross-reference (e.g. `tblr` for tables, `figr` for figures, etc) to a generic cross-reference element (`linkr` and `cross-ref` for Blackwell and Elsevier, respectively). Second, they no longer used empty elements allowing preservation of the original text. Under Elsevier DTD 4.x, the SGML for "See Tables 1-4" is "`See <cross-ref refid="tbl1 tbl2 tbl3 tbl4">Tables 1-4</cross-ref>`"

   This change allowed for more accurate rendering of the author's text.

Significant DTD updates require systems updates for publishers and producers of SGML, so publishers with a significant amount of content try to minimize the number of DTD updates and manage them through an orderly process. Elsevier, Blackwell and Wiley are among those who have tried to minimize the number of DTD updates.

Some DTDs may have continual minor updates to accommodate new journals. For example, the Highwire DTD has had several updates during 2001. These updates did not impact journals already in production because the elements were added for new journals. Existing content providers do not have to change their systems to accommodate the new DTD features.

While minor updates are backward compatible, it may be more difficult over time to manage the process of archiving of back content produced with a DTD that has experienced frequent upgrades. In order to correctly parse, transform or render SGML, it is essential to archive each DTD version along with the SGML files.

[12] Roberta Holland. XML schema catches heat. eWeek. April 23, 2001. http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2710691,00.html accessed on October 20, 2001.

Whether the changes are major or minor, multiple DTD versions require more setup work to create transformations to an archive DTD because the transformation must be adjusted to account for the differences of each DTD version.

3.4 DTD Complexity

There is no standard measure to calculate DTD complexity. However, counting the number of elements in each DTD is a simple measure that provides some interesting information. Table 3 shows the number of basic elements (excluding table and math elements) in each DTD, the number of table and math elements, and the total number of elements.

Table 3: Number of unique elements per DTD. Basic elements exclude table and math elements.

DTD	Basic	Table	Math	Total
	Elements	Elements	Elements	Elements
AIP	194	11	55[a]	260
BioOne	157	11	37[a]	205
Blackwell	165	10	131[b]	306
Elsevier	122	12	25	159
Highwire	77	8	17	102
IEEE	217	(na)	7[c]	224
Nature	206	12	36[a]	254
PMC	118	8	3[c]	129
UCP	128	7	48	183
Wiley	250	7	7[c]	264

[a] The number of math elements is large because different elements are used for face markup in math than in the body (e.g. `<bold>` in math vs. `<b>` in the body).
[b] Includes the complete MathML DTD.
[c] Math is encoded as TeX in SGML files. These elements surround TeX equations.

This data reveals several interesting patterns:

1. When a DTD is based on an industry standard DTD (e.g. AIP and IEEE are directly derived from 12083), the number of elements is larger because the industry standard includes elements that are not used by some publishers. Similarly, if a DTD incorporates an industry standard DTD component (e.g. several DTDs include MathML or 12083 math), the number of elements is larger because the DTD component is designed to solve a broader class of problems.
2. Highwire and PMC, organizations primarily concerned with online rendering and linking, have the fewest elements in their DTDs. Most other publishers include additional elements to capture data that may be related to additional publisher-specific needs, such as development of derivative content products.

Because Highwire and PMC use SGML for very specific purposes, many additional elements are not required. We do not believe the smaller number of elements in these DTDs indicates a lack of structural resolution. Rather, we believe they have streamlined their DTDs to meet their specific needs. Similarly, because an archive has a very specific mission, the DTD can probably be streamlined to meet exactly that mission.

Table 4 extends this complexity review by analyzing three randomly selected document instances for each DTD.

Table 4: Element usage per DTD: analysis of three randomly selected files. The number of unique elements is cumulative over the set of files analyzed for each publisher.

DTD	Elements used in front matter		Elements used in documents	
	# of unique elements	% of elements in documents	# of unique elements	% of elements in DTD
AIP	38	33%	114	44%
BioOne	18	31%	58	28%
Blackwell	34	41%	83	27%
Elsevier	20	25%	81	51%
Highwire	17	31%	55	54%
IEEE	24	30%	80	36%
Nature	41	45%	91	36%
PMC	25	39%	64	50%
UCP	27	29%	92	50%
Wiley	64	52%	124	47%

Some interesting patterns emerge from this data:

1. The number of unique elements used is significantly smaller than the number of elements in the DTD. No publisher used more than 54% of the elements in the DTD in the sample articles.
2. With one exception, the DTDs with the largest number of elements utilized the smallest percentage of available elements in the sample document instances (AIP, BioOne, Blackwell, IEEE, and Nature).
3. One quarter to one half of the unique elements were used prior to the document body. The front matter is a small percentage of the document size, but it represents a significant percentage of unique elements. Document metadata accounts for a significant portion of these elements.

Several points from this review of DTD complexity can be applied to development of an archive DTD:

1. In order to meet the goals of an archive, rendering and linking, the number of elements in the DTD need not be large. Additionally, moderating the number of elements will simplify the task of converting SGML from other publisher DTDs and simplify the work necessary to develop rendering software for the archived SGML.
2. Publishers tag a significant volume of metadata in document front matter. Careful review must be done during DTD development to insure all metadata relevant to the needs of an archive is included in the archive DTD.

Inera has found that it is easier to build SGML systems for streamlined DTDs with a more limited element set. For this reason, we recommend an archive DTD be streamlined to meet the specific mission of an archive. Such a streamlined DTD will permit easier implementation of SGML conversion and archive rendering systems.

4 Journal Files

Articles, primarily original research and reviews, constitute the majority of journal content by page count (or character count, from a non-print electronic perspective). All publishers surveyed capture all of this content in SGML and PDF. This section reviews specific issues associated with capture of different article types in SGML and XML.

4.1 Non-Article Content and Other Content

A journal archive must include Non-Article Content and Other Content to be complete. However this content may be more difficult to archive for the following reasons:

1. Most publishers create SGML and PDF for Non-Article Content (e.g. correspondence, editorials, book reviews, and errata). Sometimes these items have a different structure than articles, and they appear in smaller numbers and with lower frequency. Many DTDs also have special elements to tag some of this content (e.g. special elements for letters or errata).

   Unfortunately, standards for tagging Non-Article Content are sometimes less rigorously enforced. For example, we have seen cases where the authors in letters to the editor were tagged as regular paragraph text rather than with author tags because they appeared at the end of the item rather than at the beginning.

2. Many publishers do not create SGML for Other Content and Static Content. In many cases, even PDF versions of this content may not be available. Publishers may choose not to archive this content because:

   a) The content can be generated from other SGML files for electronic products. For example many organizations (Elsevier, Highwire, and IEEE) do not archive electronic versions of tables of contents or print indexes because they are recreated from the SGML.

   b) Static content is not regarded as archival material. This content is rarely available in SGML or PDF.

   c) Advertisements are typically excluded because they may be difficult to capture, and there may be copyright issues.

   Elsevier's content capture requirements present an interesting case study. In 1999, Elsevier required full text SGML for Articles and Non-Article Content. Other Content was not captured in SGML, however PDF was required for every physical page in the journal. This strategy allowed them to reconstruct entire issues from PDF files in the archive.

   After several years of practical experience, Elsevier changed their requirements. Full text SGML and PDF are still required for Articles and Non-Article Content. SGML headers and PDF are required for all other items listed on the table of contents (e.g. calendars, indexes, etc.).

   SGML headers for Other Content are created to provide a means for automatic reconstruction of an electronic table of contents. Full-text SGML is not created for items without research significance. PDF and SGML are not created for items that are not on the table of contents. This system precludes cover-to-cover archiving of journals, an archive capability that was no longer important to Elsevier, but it does capture all significant intellectual content.

3. Most publishers do not capture the masthead in SGML or PDF format. Inera has seen many cases where the only electronic version of the masthead is a Static HTML file on the journal's web site. While this information may be important for future researchers, it is not likely to be part of the archive materials delivered by a publisher to an archive.

In order for an archive to retain the most complete record of a journal, it will be necessary to specify to publishers precisely those files that should be deposited with the archive. The archive must state whether files available in only PDF format should be included. Additionally, if some content is available only in HTML format (e.g. the masthead), the archive must specify if it is to be deposited.

Some publishers may not be able to easily deliver tables of contents and indexes in SGML. For this reason, Inera recommends the archive DTD design pay special attention to article headers and front matter so that the archive can automatically create tables of contents and indexes from the individual archive XML files.

We recommend the archive DTD exclude elements for tables of contents and indexes because it will add an extra level of complexity to the DTD. Providing these files in an archive format will not add significant value to the archive because the archive will have its own capabilities to recreate these items in cases where publishers do not provide them. We believe this approach will provide the easiest route for archive deposit and use, and it will yield the most functional archive because indexing capabilities can be updated over time by the archive based on new software capabilities.

Many DTDs have tags specific to Non-Article Content or Other Content. In most cases, these tags are not necessary to meet the archive mission because the content they are used to tag can be adequately rendered with more generic tags. We recommend dropping most tags specific to these content types.

4.2 Book Reviews

Most Non-Article Content can be appropriately tagged using an Article DTD without special elements. However book reviews present special challenges. These unique challenges include:

1. Some publishers include multiple book reviews in a single SGML file while still listing each book review on the table of contents. These book reviews are often placed in a single file for production reasons (e.g. multiple reviews appear on a single page). Extra processing is often required to reconstruct a table of contents in this situation because it is necessary to extract multiple entries from the SGML file.

2. The standards to structure the book information for the book being reviewed vary from publisher to publisher. Table 5 presents two examples that illustrate the range of tagging by publishers.

Table 5: Examples of book information in Blackwell and Elsevier tagged book reviews.

Publisher	SGML Example
Blackwell	<bookdetails> <title type="book">Hotspots: Earth&apos;s Biologically Richest and Most Endangered Terrestrial Ecoregions<x>. </x></title> <namegroup type="author"><name type="author"><surname>Mittermeier</surname><x>, </x><forenames>R.</forenames></name><x>, </x><name type="author"><forenames>N.</forenames><x> </x><surname>Myers</surname></name><x>, </x><name type="author"><forenames>P.</forenames><x> </x><surname>Robles Gil</surname></name><x> &amp; </x><name type="author"><forenames>C.</forenames><x> </x><surname>Goettsch Mittermeier</surname></name></namegroup>. <date date="1999">1999</date><x>. </x> <publisher>Cemex/Conservation International and the University of Chicago Press</publisher><x>, </x> <address>Chicago, Illinois</address><x>. </x> <page type="total">430 pp.</page><x>. </x> <price>&dollar;65.00.</price><x> </x> <isbn>ISBN 968-639-58-2</isbn><x>.</x> </bookdetails>
Elsevier	<atl>Fifty Years of the International Court of Justice: Essays in honour of Sir Robert Jennings <sbt>Edited by Vaughan Lowe and Malgosia Fitzmaurice. Cambridge, UK: Cambridge University Press, Grotius Publications, 1996. 640p. ISBN 0-521-55093-9 (hardback). $125.00.</sbt></atl>

The Blackwell DTD captures book information with a special tag, `<bookdtl>`. The information about the book is tagged with a high degree of granularity. The Elsevier DTD captures the title of the book being reviewed in the standard article title element, `<atl>`, and the remainder of the book information in the article subtitle element, `<sbt>`.

3. There are no standards to tag the book information when multiple books are reviewed in a single book review. Some solutions we have seen include:
   - Elsevier: Use one `<atl>` element per book
   - Blackwell: Unsupported in DTD 3. DTD 4 loosened the model to allow for multiple books.
   - Highwire: a bullet-separated `<ATL>` listing the titles of the books, followed by a `<MISC>` element which duplicates the titles as needed to establish a correspondence with the citation information.

Further analysis is needed during DTD development to find the best method of tagging book information in book reviews, especially in situations when multiple books are reviewed in a single article. Further analysis not-withstanding, Inera recommends an archive DTD include an element to markup ISBN numbers to allow some degree of linking when services are available for linking to books online. In cases where the publisher has not tagged the ISBN number, it can be automatically identified with a reasonably high degree of accuracy using pattern recognition during conversion to the archive DTD.

4.3 Multi-part SGML Files

As mentioned in Section 4.2, some publishers (Blackwell) may include multiple Non-Article items in a single SGML file. This structure may be used for items such as book reviews, letters to the editor, and errata, especially if they are printed without starting each item on a new page. Other publishers (Elsevier) place each item in a unique file. Publishers use different strategies because they have different production processes and archiving systems.

The archive could require that all items are delivered as individual files, however this requirement may be difficult for publishers who do not assign a unique article identifier to each item within an SGML file that contains multiple items. Alternatively, an archive DTD can support multiple items within a single file, however we recommend that more design work must be done on the anticipated systems for storage and retrieval in the archive before a final recommendation is made.

5 SGML Files

This section presents the main components of journal article content that must be preserved in archival SGML and discusses unique issues associated with the preservation of this content.

5.1 Boilerplate and Generated Text

All surveyed DTDs use a similar high-level structural approach for major Article elements, in part because of the shared 12083 heritage. However, the implementation details of SGML document instances vary widely. The greatest variance is found in items that have the greatest granularity, such as figure numbers in captions and reference citations. Often the differences revolve around the use of boilerplate versus generated text and formatting.

5.1.1 Label Text

Figure numbers, which appear at the start of figure captions, are typical of the range of implementations for generated text. Table 6 shows the differences in tagging figure numbers by surveyed publishers.

Table 6: Examples of figure caption numbers. Note the varying degrees of generated versus boilerplate text and formatting in the SGML.

Publisher	Print Example	Corresponding SGML
AIP	FIG. 1.	<figgrp id="F1">
BioOne	Fig. 1.	<TITLE>Fig. 1. Two recent…</TITLE>
Blackwell	**Figure 1.**	<num id="leg-f1">Figure&thinsp;1.&ensp;&ensp;</num>
Elsevier	**Figure 1.**	<no>Figure 1</no>
Highwire	**Figure 1.**	<no><b>Figure 1.<b> </no>
IEEE	Fig. 1.	<title just="just" autonum="off">Fig. 1.&emsp;The…</title>
Nature	Fig. 1	<fig id="f1" entname="figf1">
PMC	**Figure 1**	<title><p>Figure 1</p></title>
UCP	Figure 1:	<LABEL>Figure </LABEL><NO>1: </NO>
Wiley	Figure 1.	<FIG ID="fig1" LOC="FLOAT"><GRAPHIC NAME="fig001"></GRAPHIC><NUMBER>1</NUMBER>

Several different approaches have been used:

- Some publishers include the label and number as part of the caption (BioOne, IEEE).
- Some publishers include only attribute information (AIP, Nature). The rendering application generates the figure label and number from the attributes. One publisher (Wiley) uses a variation of this model in which only the text of the number is provided.
- One publisher tags the label and number in different elements (UCP).
- Several publishers include the label and number in one element (Blackwell, Elsevier, Highwire and PMC), but there are differences in the degrees of generated text.
  - Highwire includes punctuation after the number and bold face markup.
  - Blackwell and PMC include the punctuation after the number but exclude the bold face markup. The face markup is applied by a style sheet when the SGML is rendered.
  - Elsevier excludes the punctuation after the number and the face markup. A style sheet is used to apply both.

  This overall strategy retains the original text in case anything unusual was done by the author or editor (e.g. "Figures 1a and 1b").

The differences illustrated in this table can be resolved in an archive DTD in conjunction with archive policies for boilerplate and generated text.

Inera recommends the archive DTD be designed to capture the full text of the figure number. We further recommend that archive policy require retention of boilerplate and inclusion of generated text and formatting in archive SGML to minimize the degree of style sheet development required for the rendering system. This strategy, which is similar to that taken by Highwire, can be used to capture and render content from all surveyed publishers.

To standardize the SGML deposited in an archive and facilitate easy archive rendering, Inera recommends that all generated text and face markup be included in archive document instances, avoiding the need for an archive to setup complex style sheets.

Inera recognizes that this recommendation may be controversial. We acknowledge that SGML was created to separate structure from format, and this recommendation violates the spirit of SGML's intent. However because journals with a wide range of styles will be deposited in the archive, we believe that this solution presents the only effective method for an archive to render content.

5.1.2 Citation Text

The issue of boilerplate and generated text becomes more complex with a review of citation links. Table 7 illustrates the variety of tagging used for numbered ("Vancouver" style) citations by surveyed publishers.[13]

Table 7: Examples of Vancouver-style citations in different DTDs. Note the varying degrees of automatic text generation and the handling of IDREF attributes for the middle and last numbers of a range.

Publisher	Citation	SGML Example
AIP	superscript.^{1,6}	`superscript.<citeref rid="r1" style="superior">1</citeref> <citeref rid="r6" style="superior">6</citeref>`
	superscript.$^{3–5}$	`superscript.<citeref rid="r3" style="superior">3</citeref> <citeref rid="r4" style="superior">4</citeref><citeref rid="r5" style="superior">5</citeref>`
BioOne	(1,6)	`<CITEREF RID="i0031-8655-071-01-0001-b1">&lpar;1,6&rpar;</CITEREF>`
	(3–5)	`<CITEREF RID="i0031-8655-071-01-0001-b3">&lpar;3&ndash;5&rpar;</CITEREF>`
Blackwell	[1,6]	`&lsqb;<link rid="b1 b6">1,6</link>&rsqb;`
	[3–5]	`&lsqb;<link rid="b3 b4 b5">3&ndash;5</link>&rsqb`
Elsevier	[1,6]	`<cross-ref refid="bib1 bib6">[1,6]</cross-ref>`
	[3–5]	`<cross-ref refid="bib3 bib4 bib5">[3&ndash;5]</cross-ref>`
Highwire	(1, 6)	`(<cross-ref refid="bib1" type="bib">1</cross-ref>, <cross-ref refid="bib6" type="bib">6</cross-ref>)`
	(3–5)	`(<cross-ref refid="bib3" type="bib">3</cross-ref>&ndash; <cross-ref refid="bib5" type="bib">5</cross-ref>)`
IEEE	[1], [6]	`<citegrp><citeref rid="ref1" type="ref"></citeref></citegrp>, <citegrp><citeref rid="ref6" type="ref"></citeref></citegrp>`
	[3]–[5]	`<citegrp><citeref rid="ref3" type="ref"></citeref><citeref rid="ref4" type="ref"></citeref><citeref rid="ref5" type="ref"></citeref></citegrp>`
Nature	superscript^{1,6}.	`superscript<bibr rid="b1 b6">.`
	superscript$^{3–5}$.	`superscript<bibr rid="b3 b4 b5">.`
PMC [a]	[1,6]	`[<abbr bid="B1">1</abbr>,<abbr bid="B6">6</abbr>]`
	[3-5]	`[<abbr bid="B3">3</abbr>-<abbr bid="B5">5</abbr>]`
UCP	[1,6]	`[<CITEREF RID="rf1">1</CITEREF>,<CITEREF RID="rf6">6</CITEREF>]`
	[3–5]	`[<CITEREF RID="rf3">3</CITEREF><CITEREF RID="rf4"></CITEREF>&ndash;<CITEREF RID="rf5">5</CITEREF>]`
Wiley	[1,6]	`<BIBR HREF="bib1">1</BIBR><BIBR HREF="bib6">6</BIBR>`
	[3–5]	`<BIBR HREF="bib3">3&ndash;5</BIBR>`

[a] PMC allows ranges to be tagged as shown above or as:

```
[<abbr bid="B3">3</abbr>,<abbr bid="B4">4</abbr>,<abbr bid="B5">5</abbr>]
```

The style is determined by the publisher who submits SGML content to PMC

[13] We have chosen only a close review of Vancouver-style citations here. A comprehensive review of name-date ("Harvard" style) citations would show even more variations in style because of the how these citations are formatted for print by different publishers.

Several points should be noted in the tagging of citations:

- For citations to a simple list of references, every publisher includes links to both references 1 and 6.
- For citations to a range of references, publishers may include:
  - a link to only the first reference in the range (BioOne, Wiley)
  - links to only the first and last reference in the range (Highwire, PMC)
  - links to all references in the range (AIP, Blackwell, Elsevier, IEEE, Nature, UCP)
- Publishers may:
  - include all of the original citation text (BioOne, Blackwell, Elsevier, Highwire, Wiley)
  - keep the original text, but create multiple linking elements around the text (PMC, UCP)
  - exclude the original citation text (AIP, IEEE, Nature)
- Formatting (parentheses, brackets, superscript) is handled in a variety of ways:
  - No formatting information is retained (IEEE, Nature). Formatting is applied by a style sheet
  - Formatting information is found in an attribute (AIP)
  - Parentheses/brackets are found inside the linking element (BioOne, Elsevier)
  - Parentheses/brackets are found outside the linking element (Blackwell, Highwire, PMC, UCP)
- Several publishers do not retain the range (AIP, IEEE, Nature). The rendering software must include an additional layer of processing to reconstruct the range if the citations are to appear online as a range rather than as discrete links.

The approach taken by Highwire and PMC is designed to allow easy rendering while retaining the general look of the original text. Both exclude links to the middle references in the range. The resulting user interface in HTML is simplified by creating hyperlinks for the first and last numbers as hyperlink text.

This approach causes no loss of significant functionality when linking to a reference section because references in the middle of the range are easily accessible if the reader clicks on the first or last number. However, if links to middle objects of the range are excluded when this model is used for figure or table citations it may be difficult or impossible to link to some objects.

Some publishers recommend a more sophisticated rendering engine that presents a drop down menu with a list of possible targets.[14] A model where text and links to all objects are preserved, similar to that used by Blackwell and Elsevier, facilitates this rendering style.

Because the primary mission of an archive DTD is to facilitate easy rendering of articles with full linking capabilities, Inera recommends preserving the original text and links to all target objects of citations in a manner similar to the Blackwell and Elsevier models.

5.2 Article Header and Front Matter

5.2.1 Print and SGML Elements

In a printed article, the front matter typically includes some or all of the following structural elements:

Article Title: The article title includes the text of the title. In some DTDs, the article subtitle is captured in a separate subtitle element.

Some DTDs limit the title to only text and face markup. Others use a looser model to allow almost any element that can appear in body text.

[14] Pepping, Simon and Schrauwen, Rob. *Tag by Tag*. 2001 Elsevier Science: Amsterdam. pp. 234-235.

Inera recommends a looser model because graphics and display math may occur in the title. In addition, Inera recommends a language attribute in the title element because some publishers include the article title in multiple languages.

Author(s): All publishers parse author names into a surname element and a given name element to allow searching and indexing of content. Some DTDs further distinguish between author first names versus initials, and author first names versus middle names. In some cases, an index element or attribute is also provided to allow for easier alphabetization when a name contains non-ASCII characters such as accented letters.

Additional tagged elements in most DTDs include the author role or prefix (e.g. "Dr.", "President"), author degrees ("M.D.", "Ph.D."), and the author suffix (e.g. "Jr.", "III"). In some DTDs the author suffix is is included in the surname element rather than in its own element. Most DTDs distinguish between an author who is a person, and an organizational author.

XML DTDs present a unique issue when tagging author names. A common DTD structure in SGML is:

```
<!ELEMENT author - -    ( fname? & surname)>
```

The use of "&" in this structure allows the name to appear as either "first last" or "last first". Flexibility in the order of presentation is important because some countries (e.g. Hungary, China) list surnames before given names. Other countries may, when alphabetizing lists of names, use the first name rather than the last name as the sort key (e.g. Iceland).

The "&" operator was removed from XML, so an alternate structure must be used in an XML DTD. One approach is the addition of a name-style attribute to the author:[15]

> *The "name-style" attribute may be used for choosing an inversion algorithm, sorting, or other processing functions. The three values and approximate meanings are:*

Attribute	Display Order	Sort/Inversion
western	*given family*	*family given*
eastern	*family given*	*family given*
islensk	*given patronymic*	*given, patronymic*

Links to author affiliations and footnotes (e.g. corresponding author, current address, etc.) appear in the author group in most DTDs.

The Elsevier DTD includes a `<ranking>` element to mark the more important authors as is done in some scientific disciplines, especially Chemistry. In print, the ranking indication typically appears as an asterisk.

Inera recommends an archive DTD include elements for the author surname, given name, suffix, degree(s) and role(s). The inclusion of a name-style type attribute will aid in conversion of SGML that has taken advantage of the "&" operator. Additionally, we recommend inclusion of a distinct element to tag organizations as authors. Elements to link to affiliations and footnotes should be kept.

Inera recommends excluding elements for author's middle initials or ranking. The former can be placed in a given name element, and the latter in the unstructured text that is included in the author group. Inera considers this a tolerable loss of resolution.

Affiliation(s): All publishers markup the affiliation of the article author(s) and all surveyed DTDs allow creation of a link between each author and his affiliations(s).

Visually such links can be established in a number of ways:

- A footnote symbol
- The author's initials or full name are located within the affiliation
- The affiliations appear just after the author names

[15] From an email posting by Debbie Lapeyre of Mulberry Technologies to the SGML Forum of New York, August 21, 2001.

SGML links are usually not created when there is only one author or only one affiliation. In some cases when there are multiple authors and multiple affiliations, the SGML links may not be added because there were no links indicated in print (e.g. the journal Kidney International), or because the SGML supplier did not add them. Many of the same issues highlighted in Section 5.1.2 also impact the format of affiliation links.

Three publishers structure information within the affiliation:

- Elsevier tags the city and country
- Nature tags organization, street, city, state, postal code, and country
- Wiley includes the country and organization type as attributes of the <AFF> element

The remaining seven surveyed publishers do not add structure inside the affiliation.

The most complex structural aspect of affiliations is the subdivision of individual organizations and departments within an affiliation. When there are multiple affiliations, most publishers place each affiliation in a unique affiliation element. Consider the following author/affiliation:

> Bruce G. Haffty, M.D.,* Peter L. Perrotta, M.D.,† Barbara Ward, M.D.,‡
> Meena Moran, M.D.,* Malcolm Beinfield, M.D.,‡ Charles McKhann, M.D.,‡
> Diana Fischer, Ph.D.,* and Darryl Carter, M.D.†
>
> *Departments of *Therapeutic Radiology, †Pathology, and ‡Surgery, Yale University School of Medicine, New Haven, CT*

Most of the surveyed DTDs require multiple <AFF> elements in order to create unique author/affiliation links. In this situation, the resulting SGML breaks up the departments in a rather inelegant fashion, as shown in these two examples:

```
<aff id="aff1"><no>*</no>Departments of Therapeutic Radiology, Yale
   University School of Medicine, New Haven, CT</aff>
<aff id="aff2"><no>&dagger;</no>Departments of Pathology, Yale University
   School of Medicine, New Haven, CT </aff>
<aff id="aff3"><no>&Dagger;</no>Departments of Surgery, Yale University
   School of Medicine, New Haven, CT</aff>

<aff id="aff1"><no>*</no>Departments of Therapeutic Radiology</aff>
<aff id="aff2"><no>&dagger;</no>Pathology</aff>
<aff id="aff3"><no>&Dagger;</no>Surgery, Yale University School of
   Medicine, New Haven, CT</aff>
```

In the first case, the text "Departments of" is repeated multiple times and is incorrectly plural. In the second case, the original text is preserved, but the rendering could be awkward if the rendering software placed each affiliation on its own line.

Some publishers have tried to resolve this issue by disallowing this affiliation format in copy editing, however there are cases when it will still appear. Blackwell handles this situation with an address element that can contain multiple author links:

```
<address>
<span id="a1"><number>&ast;</number>Departments of Therapeutic Radiology,
   </span>
<span id="a2"><number>&dagger;</number>Pathology, and </span>
<span id="a3"><number>&Dagger;</number>Surgery, Yale University School of
   Medicine, New Haven, CT</span></address>
```

Alternatively, SGML in this example can be transformed with retention of the symbols, but without the SGML links. While this solution sacrifices the links, we do not believe it is a significant resolution loss because the authors and affiliations are often rendered in close visual proximity.

Inera recommends an affiliation model in which each affiliation is tagged with a unique <AFF> element. The link symbol should be placed in a number element rather than created with generated text. We do not recommend elements to structure address components within the affiliation.

In cases where the original presentation alternated the authors and affiliations, we recommend creation of SGML links between the author and affiliation elements with no number element. All of the authors should be contained in an author group and all of the affiliations in an affiliation group. The DTD is simplified if authors and affiliations cannot appear intermixed in the content model.

The "Departments of…" situation can create messy affiliations, however we do not recommend this issue receive special attention in an archive DTD because a reader can still interpret the connections between authors and affiliations when content is rendered from any of the SGML examples shown above.

Corresponding Author Information: Many articles include a footnote with the name of the corresponding author. Depending on the journal style, the footnote may include the author's name, address, phone or email information. In other cases the corresponding author is marked with an asterisk to cite a simple footnote "Corresponding author", and the address is found in the affiliation. Occasionally articles are published with more than one corresponding author, and in some journals, the corresponding author information is placed near the author rather than as a footnote.

Table 8 shows the wide variation of styles used to indicate the corresponding author.

Table 8: Tagging of corresponding author by publisher. See text for an explanation of the headings.

Publisher	Element Name	Address Parsed	Email Placement	Footnote Symbol	Footnote Placement
AIP	`footnote`	N	Footnote text [a]	N	Back
BioOne	`FOOTNOTE`	N	Footnote text	N	Back
Blackwell	`correspondent`	Y [b]	Footnote text	Y	Front
Elsevier	`cor`	N	Author	Y [c]	Author
Highwire	`cor`	N	Footnote text	Y [c]	Author
IEEE	`affnote`	N	Footnote text [a]	N	Front
Nature	`CAU/CAFF` [d]	Y	Footnote text	N	Front
PMC	`cor`	N	Author	N [e]	Front
UCP	`corraddr` [f]	N	Footnote text	Y	Front
Wiley	`CORR`	N	Author	N	Front

[a] The email address is part of the text but not tagged with a unique element

[b] Fully parsed addresses are supported by the DTD, but Blackwell does not require that suppliers provide this granularity.

[c] Earlier versions of these DTDs did not allow retaining the footnote symbol. The symbol was created with generated text. Newer versions place the symbol in a `<no>` element.

[d] `CAU` and `CAFF` are special forms of a `AU` and `AFF` elements used to indicate the corresponding author and address

[e] The link between `au` and `cor` is established with the attribute `ca="yes"` in `au` rather than with a traditional ID/IDREF pair used in other DTDs.

[f] At UCP, the tagging of corresponding author information depends on the style of the journal. In many cases, the information is in normal `footnote` markup. The placement of the `footnote` element varies, although it's usually in the back with the other floating objects. E-mail addresses can be tagged, wherever they occur, at the discretion of the copy editor.

Element Name gives the name of the element used to tag the corresponding author. *Address Parsed* indicates if the author's address is sub-parsed into street, city, country, etc.

Some DTDs place the corresponding author's email address in the text of the corresponding author footnote. This placement is indicated with "Footnote text" in the *Email Placement* column. Other DTDs that place the email element just after the author's name in the author group are indicated with "Author". This latter model allows SGML to include an email address for each author. It is a more flexible model because email addresses can be retained for all authors rather than just the corresponding author.

Footnote Symbol indicates whether the linking character between the corresponding author and the corresponding author footnote is preserved in the SGML. When it is not preserved, the symbol is usually rendered with generated text. In other cases, the journal style may not include a linking symbol because the name of the author appears in the corresponding author footnote (e.g. "Address correspondence to John

Doe <jdoe@research.org>"). The use of generated text in this case is journal-dependent, not DTD dependent.

Footnote Placement gives the location of the corresponding author footnote element in the SGML file. Depending on the DTD, it may be:

- *Front:* In the front matter at a location determined by the DTD
- *Back:* In the back matter with other floating objects such as figures and tables
- *Author:* After the author's name in the author list. This model allows the content to be linked to an author without using an ID/IDREF pair, however it is inflexible when articles have multiple corresponding authors.

Inera recommends two methods of corresponding author support in an archive DTD:

1. *A corresponding author element should be included in the DTD. It should be located in the front matter, however there should not be a requirement to place it after the author name in the author group. The link between the corresponding author in the author group and the corresponding author element should be established with an ID/IDREF pair.*
2. *Publishers can place the corresponding author in a generic footnote if they have done so in their SGML. The footnote element must have an ID attribute that can be referenced with a linking element from the author group. If a footnote symbol was used in print, it should be included in a number element in the footnote and also in the link in the author group.*

In both of these models, if no linking symbol was used in print the link between the corresponding author in the author group and the corresponding author element should be established with an ID/IDREF pair. A number element will not appear in the footnote or corresponding author element, and the content of the link in the author group will be empty.

We recommend placement of the corresponding author's email address as a tagged element in the footnote or corresponding author element. However, the DTD should also permit an email address to be associated with each author to retain this information for non-corresponding authors. We do not recommend elements to structure address parts within the corresponding author. The recommended model is similar to the corresponding author structure used by Blackwell.

Abstract: The abstract section includes text and occasionally a fixed graphic. In issues that include just abstracts of other articles (such as abstracts of conference papers), abstracts may also include tables. For these reasons, most DTDs use the same structural model for abstracts as they do for body text. We see no reason to change this model in an archive DTD.

Some DTDs allow headings in abstracts (e.g. "Introduction", "Methods", "Conclusions", etc.) to be tagged as section heads, and some just allow face markup (e.g. bold, italic). For the purposes of archive SGML, Inera recommends that these headings be tagged simply with face markup.

Because some publishers include abstracts in multiple languages, Inera recommends the inclusion of a language attribute in the abstract element.

Footnotes: Front matter footnotes include current author address information, dedications, reprint addresses, etc.

As with the example of the figure number given above, we recommend that all generated text, including footnote-linking symbols, be included in the archive SGML. Consideration may be given to an attribute that describes the footnote type, but it is not essential.

Abbreviations and Definition: Abbreviation and definition lists are included to aid the reader in identification of commonly used abbreviations in an article.

Some publishers treat abbreviations as a special class of keywords. Others treat them as a simple footnote. For the purposes of rendering text in an archive DTD, abbreviations can be tagged in either model. However, definition lists present more structure than a typical abbreviation element, including possible headings.

Inera recommends a unified structure for abbreviations and definition lists in an archive DTD. This structure should include an attribute to distinguish if the list contains abbreviations.

By archive policy, publishers that treat abbreviations as footnotes should not be required to re-parse the content into a more granular form. The footnote will be rendered as-is, preserving the intellectual content.

Copyright: All DTDs include elements to tag the article copyright. Some DTDs include a separate element for the copyright year.

Inera recommends an archive DTD include a separate element for copyright year to ease searching archives for material with an expired copyright. Parsing the copyright year will increase the DTD granularity for some publishers, but it should be relatively easy to identify and tag the year using pattern recognition during an automated DTD conversion process.

5.2.2 Other Front Matter Elements

Certain elements may appear in print and SGML, or in some cases they appear only in SGML. These elements are:

Article History: The article history includes the received, revised, and accepted dates for the manuscript. In some cases, these dates appear in SGML but may or may not appear in print. In addition, some publishers now include one or more publication dates (e.g. online, print, etc.) as part of the article history.

Surveyed publishers use several models to tag the article history:

- Both parsed and unparsed article history (Blackwell 4.0):

  ```
  <history><p>Accepted for publication 27 July 1999</p></history>
  <trackinghistory>
  <trackingdate type="accepted" date="1999-07-27"/>
  <trackingdate type="paginated" date="1999-09-12" by="Typesetters Ltd"/>
  </trackinghistory>
  ```

  Note that Blackwell uses `<history>` for print and `<trackinghistory>` to track production internally, e.g. dates at which the XML was created, edited, published online etc.

- Dates parsed into very granular attributes with boilerplate text included (Blackwell 3.0):

  ```
  <hst><re year="1997" month="12" day="30">Received for publication December
  30, 1997 and <rv year="1998" month="07" day="21">in revised form July 21,
  1998 <acc year="1998" month="07" day="27">Accepted for publication July 27,
  1998</hst>
  ```

- Dates tagged separately with boilerplate text included (UCP):

  ```
  <HISTORY>Received <RECEIVED>2001 February 12</RECEIVED>; accepted
  <ACCEPTED>2001 May 22</ACCEPTED></HISTORY>
  ```

- Dates parsed as attributes with no generated text (Elsevier, Nature):

  ```
  <re day="20" mo="7" yr="2000"><acc day="18" mo="12" yr="2000">
  ```

- Dates parsed into date type elements with no generated text (AIP):

  ```
  <history><received><date>26 July 1999</date></received><accepted><date>23
  November 1999</date></accepted></history>
  ```

- Dates parsed into very granular elements with no generated text (PMC):

  ```
  <history><rec><date><day>13</day><month>12</month><year>2000</year></date>
  </rec><acc><date><day>09</day><month>2</month><year>2001</year></date></acc>
  <pub><date><year>2001</year></date></pub><epub><date><day>09</day>
  <month>2</month><year>2001</year></date></epub></history>
  ```

Inera recommends an archive DTD model that preserves the original text while providing attributes for parsed date values to permit archive searches, similar to the model used by the Blackwell 3.0 DTD. The parsed date attributes must be optional, however, because many publishers have not parsed this information, and it would place a burden on them to parse it when transforming files to an archive DTD.

Keywords: Keywords are used to aid in classification of articles and rapid searching of articles by key terms. They often appear in SGML files without appearing in the print edition of an article.

All surveyed DTDs have elements to tag keywords. In the Elsevier DTD, the keyword group includes an attribute to indicate the type of keyword because many disciplines have specific keyword classification schemes:

```
<!ENTITY % kwd-class    "(kwd|abr|jel|msc|pacs|mat|src|idt|psycinfo|
neurosci|inspec-cc|inspec-ct|inspec-chi|stma|astronomy|geo|cras-terre)" >
```

If attributes are used to identify keyword classes rather than unique elements, the model can be extended more easily to accommodate new keyword classifications.

Inera recommends a keyword classification system that uses attributes to identify the keyword type. We recommend the DTD be designed to tag keywords with all generated text retained in the archive SGML file. The generated text should be within the keyword group, but outside of the individual keyword elements.

5.2.3 Metadata Elements

Publishers use a wide variety of metadata elements and attributes in the front matter of journal articles to markup the metadata. There is significant overlap of key metadata in the publisher group, however many publishers include unique elements required for their specific requirements.

Table 9 includes the minimal set of metadata elements that Inera recommends in an archive DTD. Further analysis must be done during DTD development to create a complete list of elements required in an archive DTD.

Table 9: Recommended minimum set of metadata elements for an archive DTD.

Class/Element		Description
Publisher Metadata		Information about the publisher of the article.
	Publisher Name	The publisher may be different from copyright holder.
	Publisher Location	The publisher location should include the city and country, the city and province if Canada, or the city and state if in the United States.
Journal Metadata		Information about the journal in which the article was published.
	Journal Title	The full title of the journal (e.g. "Current Biology").
	Journal Abbreviated Title	The abbreviated title by which the journal is commonly cited (e.g. "Curr. Biol."). It is useful to allow multiple occurrences of this element as journals are sometimes cited in multiple forms (e.g. "JAMA", "J. Am. Med. Assoc.").
	Journal ID	The publisher's ID for the journal (e.g. CURBIO). It is useful to allow for multiple occurrences of this element with a type attribute in each occurrence. This model will allow for expansion, such as DOI numbers that identify an entire journal.
	ISSN	The ISSN number. The archive DTD must allow multiple occurrences of ISSN to account for different print and electronic ISSN numbers.
	Coden	The journal Coden.
Publication Metadata		Information about the issue in which the article appeared.
	Volume	The volume number in which the article appeared.
	Issue	The issue number in which the article appeared.
	Special Numbering	Special issue numbering information such as a supplement number.
	First Page	The first page number of the article.
	Last Page	The last page number of the article.
	Other Pages	An element to hold additional page information if the article appeared on non-contiguous pages.
	Publication Date	The date of publication. This element may repeat to distinguish print and online publication dates.
	Journal Price	The price of the printed journal. [a]
Article Identification Metadata		Information used to uniquely identify the article.
	Public article ID	One or more article ID numbers based on a public standard such as PII, SICI, DOI; This element must have a type attribute and allow for future public article identification systems.
	Publisher article ID	A publisher-specific article ID based on a proprietary standard.

Class/Element		Description
	Related article ID	One or more ID values that allow this article to be linked to other articles from the same publisher. Examples include the `refers-to` attribute in the Elsevier DTD and the `uri-orig` attribute in the UCP DTD. They are used to link errata or other items to the original article.
Article Metadata		Information about the content of the article
	Article Type	Many DTDs include a short code to indicate the type of article as "RA" for Research Article, "BR" for Book Review, etc. Further document analysis across all publishers is required to determine an appropriate list of common values for an archive DTD.
	Figure Count	The number of figures in the article.
	Table Count	The number of tables in the article.
	Equation Count	The number of display equations in the article.
	Reference Count	The number of references in the article.
	Page Count	The number of pages in the article. [b]
	Word Count	The number of words in the article. This number is usually an approximation as the definition of a "word" in scientific research is not always clear.
	Language	The language of the article text.
Sponsorship Information		Information about funding for the research
	Sponsor	The sponsoring organization for the research
	Grant Number	The grant or contract number(s) for sponsorship
Table of Contents Information		Information used to construct the table of contents
	Document Head	The heading, such as "Research Article" or "Book Reviews", that may appears at the top of the first page of the article
	Document Topic	The heading that appears, if the journal TOC is sectionalized by topic, at the start of the TOC section.
	Document Subject	Used to specify one or more subject names or codes for the article. These elements are similar to keywords, except that they are generally drawn from a smaller list of more "coarse-grained" terms.
Source SGML Information		Information about the publisher's SGML [c]
	Publisher DTD	The name of the DTD in which this article was originally tagged by a publisher.
	DTD Version	The version of the original DTD.

[a] One DTD, IEEE, includes the journal price in the article metadata. Although infrequently found, it is quite easy to support and may provide some interesting historical information in the archive

[b] Because page numbers may have letter prefixes or suffixes, or they may be Roman numerals, it's easier to preserve the page count than calculate it.

[c] These elements will aid in tracking problems in archive files.

5.2.4 Article Header

The header of an SGML article instance typically includes article metadata. Table 10 shows the critical metadata that each publisher includes in the document header.

Table 10: Metadata included in SGML file for article.

Publisher	Volume/ Issue	Pages	ISSN/ Coden	SICI/ PII	DOI
AIP	Y/Y	Y	Y/Y	Y/N	Y
BioOne	Y/Y	Y	D/N	Y/N	Y
Blackwell	Y/Y	Y	Y/D	N/N	Y
Elsevier	N/N	N	N/N	N/Y	D
Highwire	Y/Y	Y	N/N	**N/D**	**D**
IEEE	Y/Y	Y	Y/D	N/Y	D
Nature	Y/Y	Y	Y/N	**N/N**	**D** [a]
PMC	Y/Y	Y	Y/N	**D/D**	**D**
UCP	Y/Y	Y	N/N	**N/N**	**D** [b]
Wiley	Y/Y	Y	Y/N	Y/N	Y

[a] Nature encodes the DOI in the `<article>` `id` attribute (e.g. `<article id="ng726">`), however the id does not include the Nature DOI prefix (e.g. `10.XXXX/`) that would be required to create a link.

[b] Although not present in the sample files, the DOI attribute is used in new content at UCP.

Key: Y = Supported in DTD and included in sample files
N = Not supported in DTD
D = Supported in DTD, but not present in sample files.

Bold indicates publishers that do not include a public standard article identifier

SICI, PII, and DOI are the most prominent public standards for article tracking, however some publishers do not include an article identifier based on one of these standard forms. In some sample articles that were reviewed, complete public standard article identifiers were not present (Highwire, Nature, PMC and UCP). It is possible that such information is found in the publisher's delivery envelope.

Most publishers include information about the volume and pages where an article appeared. However in some cases (Elsevier), this information is contained in the delivery envelope rather than the individual SGML files.

Because some SGML files lack critical archive data within the file, a simple conversion from one DTD to another cannot always be conducted. An automated process that converts publisher's SGML to an archive DTD may need to take as input the SGML file *and* data from the delivery envelop. Merging data from multiple sources will make conversion to the archive DTD a more complex process.

Inera recommends requiring ISSN and at least one public standard article-identifier number in all deposited content to allow easier tracking and retrieval of archive material. We believe that DOI is the best public standard article-identifier to use. Additionally, when articles have a print manifestation, we recommend requiring the volume/issue and page numbers.

5.3 Body Elements

All surveyed DTDs tag body text in a structurally similar manner. All have elements for tagging section heads and paragraphs (all of the DTDs use the nearly universal `<P>` element for paragraphs). Within the body, in addition to paragraphs, one typically finds lists, tables, figure captions and equations. Within these primary elements, typical sub-elements include face markup and links.

The majority of article body content consists of regular paragraphs. Some additional elements may appear in the body of articles. The following is a brief survey of these elements:

Section Heads: Section heads appear throughout documents. They include the section title, some indication about the level of the section, and possibly a section number.

Most, but not all, DTDs require a corresponding end tag at the end of the section. Inera recommends the use of an end tag to indicate the end of a section.

Some DTDs indicate the section level strictly through the depth of nested section elements (Elsevier, Highwire). However, this model is difficult to use if an article has a level 1 head followed by a level 3 head without a level 2 head. While this structure may be the result of an editorial error, it may appear in SGML delivered by some publishers.

Because some articles may have improperly nested section levels, Inera recommends the use of a level attribute to indicate the depth of each section head.

Some DTDs, notably 12083, use a different name for each section level (e.g. <`section`>, <`subsect1`>, <`subsect2`>, etc.). Inera recommends the use of one element name (e.g. <`sec`>) for all levels of sections, with the section depth indicated in a level attribute. The Nature DTD illustrates the model recommended by Inera for tagging section heads.

Lists: Lists usually appear in numbered or bulleted form. Lists may be nested, i.e. an item in a list may contain a second level list within the item.

Table 11 indicates that most DTDs use an attribute to indicate the type of list, and publishers use generated text to create the numbers or bullets when rendering the SGML instance. The only exception among the surveyed publishers is UCP where each list item must have a <`no`> element in which the item number or bullet is explicitly given.

Table 11: List attribute types by DTD.

DTD	List Type Attributes
AIP	`1 | 2 | 3 | 4 | 5| 6`[a]
BioOne	`1 | 2 | 3 | 4 | 5| 6`[a]
Blackwell	`1 | a | A | i | I | symbol | text`
Elsevier	`ord|unord|tab`
Highwire	`ord|unord|tab|letter|letterupper|roman|romanupper|disc|square|circle`

IEEE	`1|arab|2|upalph|3|uprom|4|bull|5|dash|6|none|7|loalph|8|lorom|`
	`9|generic|10|steparab|11|casearab`

Nature	`bullet | number | ucletter | lcletter | ucroman | lcroman`
PMC	`1 | 2`
UCP	item number specified in the <`no`> element
Wiley	`SEQUENCED | LABELLED | UNLABELLED`

[a] Follows 12083 list type definitions

A few DTDs, such as the Wiley DTD, allow specification of list numbering by type and also include a provision to create an unlabelled list and then apply an explicit number to each item in a <`NUMBER`> element.

Although most publishers use generated text to number list items, Inera recommends an archive DTD implement a model similar to the UCP DTD for lists. This model requires explicit generation of content for the number element during DTD transformation, however it will ease the burden of style sheet setup when an archive file is rendered.

Text Boxes: Text boxes are sidebars to the main article that appear in a separate box when the journal is printed. They typically have a caption with a number (e.g. "Box 1") and body text.

Textboxes are supported in the Blackwell, Elsevier, Highwire, IEEE, Nature, PMC, UCP and Wiley DTDs.

Elsevier also supports linked textboxes, which can be regarded as small articles included in a document. Occurring in only certain journals, they appear as floating elements in the main article. Some have their own figures and tables, and even bibliographic references. A linked textbox in the Elsevier DTD is declared as an SGML subdocument.

Inera recommends text boxes be fully supported in an archive DTD, however further document analysis is required during DTD development to determine the best way to handle text boxes in an XML DTD where SUBDOC and inclusion and exclusion exceptions are not available.

Figures: The figure caption and linking information are part of an SGML document instance even though the actual graphic is contained in a separate file. Section 6.1 describes this element in greater detail.

Inera recommends a figure element model, similar to that of Highwire, which preserves the figure number along with attribute for information about the image file and its placement. This model can be used to capture significant figure data from all surveyed publishers and allows for easy rendering.

A complete figure under the suggested model would appear as follows:

```
<fig id="fig1" loc="float"><no><b>Figure 1.</b> </no><caption><p>Caption
text… </p></caption><link locator="XYZ1663f1a"><link locator="
XYZ1663f1b"></fig>
```

Figure Copyrights: In some cases figure copyright may differ from that of the entire article. To facilitate better copyright management, it is advisable to have a separate copyright element associated with each figure element (Elsevier, Wiley).

Archivists must consider rights management issues when accepting figures with different rights than articles. In some cases, publishers have only print reproduction rights for figures and may not have rights to deposit them in an electronic archive.

Inera recommends including a copyright element for figures in an archive DTD.

Formatted Text: Several DTDs include elements to explicitly markup formatted text such as poetry (AIP, BioOne, IEEE), transcripts (IEEE), and computer program listings (IEEE). Since these elements differ contextually from standard text, it can be important or even essential to preserve the layout of the text.

The Wiley DTD provides a generalized solution to this problem with the VERBAT element in which line endings and spacing are preserved.

Inera recommends a generic element to indicate preservation of text layout and spacing. This element should have a type attribute to indicate the content within the element. The attribute should be CDATA to allow maximum flexibility, however the accompanying documentation should provide recommended values. We further recommend adding elements that can be used to specify fixed amounts of horizontal and vertical spacing, and line breaks when laying out such text.

Possible markup under the recommended model might be:

```
<p>
<verbat content-type="computer">
// If no error, peek at memory mapped file
<line-break/><vertical-space points="6"/>
if ( dwError == 0 )
<line-break/><horizontal-space points="36"/>
MMFHeader* pHeader = ( MMFHeader* )( pvView );
</verbat>
</p>
```

which would render as:

```
// If no error, peek at memory mapped file

if ( dwError == 0 )
       MMFHeader* pHeader = ( MMFHeader* )( pvView );
```

Questions and Answers: Question and answer sections are designed to allow creation of interactive quizzes such as continuing medical education tests. The Blackwell and Highwire DTDs include special elements used to markup such sections.

Interactive presentation is probably beyond the mission of an archive DTD, so the inclusion of these elements may not be necessary. Inera recommends further analysis during DTD development to determine *if these elements must be kept in an archive DTD, or if question and answer sections can be handled with the more generic recommended markup for formatted text such as poetry.*

5.4 Object Placement

"Fixed" and "float" objects may be distinguished by the fact that fixed objects must appear at a precise point within the full text for the content to be comprehensible, while it is sufficient for floating objects to appear in the vicinity of related text which will be wrapped around them.

Figures and tables cited in the article body float to a position near the first citation of the figure or table. In contrast, a special character displayed as a graphic, because it is unavailable in the entity set of the DTD, is a fixed graphic because it is attached to a precise point in the text.

Table 12: Location of floating tables and figures in SGML files.

DTD	Table Location	Figure Location
AIP	End of article body	End of SGML file
BioOne	End of SGML file	End of SGML file
Blackwell	In objects section of back matter	In objects section of back matter
Elsevier	First citation	First citation
Highwire	At first citation or end of file	At first citation or end of file
IEEE	First citation	First citation
Nature	In objects section of back matter	In objects section of back matter
PMC	End of the paragraph of first citation	End of the paragraph of first citation
UCP	End of SGML file [a]	End of SGML file [a]
Wiley	End of the paragraph of first citation	End of the paragraph of first citation

[a] As a matter of style UCP places floating objects that the end of the file, although the DTD permits placement of floating objects in the body.

Note: If a publisher specifies the object should appear at the point of citation, they usually make an exception if the citation point is inside another object. For example if the first citation of Table 1 is inside the caption of Figure 1, then Table 1 should be placed at the first citation of Table 1 in the article body rather than inside the caption of Figure 1. If the body does not contain a citation (which may be considered poor editorial style), then the object is placed at the end of the article body.

Table 12 shows how publishers specify the location of floating objects in SGML files. Most rendering systems can place a floating object near the point of the first citation no matter where the SGML for the object is located in the file. The link, having been established between the citation and the object, gives the rendering engine enough information to place the object.

This system of object placement breaks down when an object is not cited by the article text. There is one common case where objects often are not directly cited: appendices.

When an object appears in an appendix, it may not be cited directly. For example, Inera has seen many cases where the body of the article cites "Appendix A". At the end of the article, there is a heading, "Appendix A", and then a table that might be labeled "Table A1". In the SGML file, a link appears at the citation (e.g. "`details are presented in Appendix <APPR HREF="app1">A</APPR>`") and the appendix head (e.g. `<APPM><APP ID="app1"><ST>APPENDIX A</ST>`). If there is no citation for Table A1 (and in the appendix case, there rarely is), the table is left hanging.

Most publishers who specify the object should be placed at or near the first citation make an exception for appendix objects. These objects are placed in the appendix section instead. At this location, these objects are quite easy to place when rendering the SGML. When all floating objects appear at the end of the SGML file, more logic is required in the renderer because it must determine which objects are appendix objects and the place them accordingly since there is no citation to aid in placement.

Because of the problems with placement of floating appendix objects, Inera recommends that floating objects be placed at the point where they are first cited in the SGML file. Floating appendix objects should *unconditionally be placed in the appendix section as fixed objects, even in cases where they may be cited directly by the body text.*

5.5 Back Matter

Acknowledgements: Acknowledgements typically consist of one or more paragraphs of text. A few DTDs also allow section heads within acknowledgements.

Inera recommends acknowledgements allow one or more paragraphs. For simplicity, if an acknowledgement contains section heads, they should be treated as text paragraphs with emphasized face markup.

Appendices: Appendices typically contain additional text, or supplementary figures or tables. Most DTDs treat appendices as a separate element in the back matter of the document. A few DTDs exclude appendix elements (Highwire) and treat the appendix as another section of the article body.

Appendices that consist of additional text are relatively easy to handle. The structural model is typically the same as the article body. However, appendices that contain only objects (tables or figures) can be more difficult to render correctly. See Section 5.4 for a description of these problems and recommended solutions.

Inera recommends the appendix element be kept in an archive DTD. We believe this content should be distinctly structured, and the use of an appendix element will aid in placement of appendix figures and tables.

Further Reading: Several DTDs include a section, further reading, as distinct from the reference section. Typically this section includes references that are not cited in the body of the article. In addition, the Elsevier and Wiley DTDs allow paragraphs of text interspersed with the bibliographic entries for additional author commentary on the readings.

Inera recommends the further reading structure be kept in an archive DTD. However if publishers have structured this content in other ways, such as an additional reference list or as an appendix, they should not be required to convert it to the further reading structure.

Glossary: The Elsevier and PMC DTDs include elements to structure glossaries. Both DTDs support cross-references to other locations in the article. The Elsevier DTD also allows for nested glossary entries.

Although not widely used, the structural information presented in glossaries can be quite useful. For this reason, Inera recommends inclusion of glossary elements in an archive DTD.

Note Added in Proof: A few DTDs (PMC, UCP, Wiley) include a specific element for notes added in proof.

Inera recommends that this element be retained in an archive DTD to indicate content that may not have been peer reviewed.

Vita: Most DTDs have an element to present an author's vita or biography. In some DTDs a link can be made between the specific author and the vita element.

The vita can be problematic when rendering because it may include a picture of the author. These images may or may not have a caption and are typically uncited, which can result in placement problems.

Inera recommends the inclusion of this element in an archive DTD. In addition, for those publishers who support a link, we recommend this element be linkable to the appropriate author in the document front matter. Author pictures, when not cited, must have a link within the vita element to allow proper placement.

5.6 References

The devil can cite Scripture for his purpose.[16]

Without doubt, structuring reference sections is one of the most difficult aspects of E-Journal SGML production. Table 13 summarizes the types of references for which different publishers create structured SGML.

Table 13: Reference types for which structured SGML is created. Book includes monographs and edited books. Conf is a conference proceeding. Standard is a publication by a standards organization such as ANSI, ASTM, etc. An E-Ref is a reference to a web page.

DTD	Journal	Book	Conf	Report	Patent	Thesis	Standard	E-ref [d]
AIP	F	F	F	F	N	N	N	F
BioOne	F	F	F	N	N	N	N	F
Blackwell	F	F	N	N	N	F	N	F
Elsevier	F	F	F	N	N	N	N	F
Highwire	P [a]	N	N	N	N	N	N	F
IEEE	F	F	F	F	F	F	F	F
Nature	F	F	N	N	N	N	N	F
PMC	F	F [b]	F [b]	N	N	N	N	F
UCP	P [c]	N	N	N	N	N	N	F
Wiley	F	F	N	N	F	N	F	F

Key: F: Full structure support
N: No structure support. Reference is unstructured text inside a single SGML element.
P: Partial structure support

[a] The Highwire DTD has elements to tag the journal title, volume, date and first page. Other reference components are not tagged because they are not required for reference linking to databases today.

[b] PMC allows fully structured book and edited book references. Some publishers, however, may only partially tag the reference content in their own DTD. The result, when converted to the PMC DTD, may be a partially tagged reference. For example:

```
<bibl id="B12"><title><p>Arenaviruses (Chapter 50).</p></title>
<aug><au ca="no" ce="no"><snm>Peters</snm><fnm>CJ</fnm></au>
<au ca="no" ce="no"><snm>Buchmeier</snm><fnm>MJ</fnm></au>
<au ca="no" ce="no"><snm>Rollin</snm><fnm>PE</fnm></au>
<au ca="no" ce="no"><snm>Ksiazek</snm><fnm>TG</fnm></au></aug>
<source>In Virology; third Edition. Edited by B.N. Fields, et al.
Lippincott-Raven,</source><pubdate>1996</pubdate>
<fpage>1521</fpage><lpage>1551</lpage></bibl>
```

In order to establish a link from this data, it would be necessary to sub-parse the `<source>` data: book name, edition, editors, and publisher.

[c] UCP does not include structural sub-elements in references. Reference linking information to a variety of databases (e.g. ADS, Medline) is stored in reference attributes. When queried, UCP replied they could tag sub-elements based on pattern matching to produce archive SGML.

[d] All surveyed DTDs support tagging of external links in references, allowing the link to be completed if the data is correctly tagged. However in the case of electronic references, many of the other elements may not be fully structured due to the variety of styles used by authors in citing web resources. As a result, E-refs are fully structured in some cases and partially structured in others.

Typically journal reference have a somewhat predictable structure. Most publishers tag these references in order to establish external links. However, the tagging approach varies in several aspects from publisher to publisher.

- Structure:
  - All elements are structured with significant granularity (AIP, BioOne, Blackwell, Elsevier, IEEE, Nature, PMC, Wiley)
  - Only enough content is structured to establish database links (Highwire)

[16] William Shakespeare. *The Merchant of Venice*. Act I. Sc. 3.

  - No elements are structured (UCP - see footnote c of Table 13)
- Element order:
  - All elements appear in their print order (Blackwell, Highwire, Nature, UCP, Wiley)
  - All elements are ordered in a fixed manner prescribed in the DTD (AIP, Elsevier, PMC)
  - Some elements are ordered in a fixed manner prescribed in the DTD, and some elements appear in their print order (BioOne, IEEE)
- Punctuation:
  - Punctuation between elements is retained (AIP, Blackwell, Highwire, Nature, UCP, Wiley)
  - Punctuation between elements is not retained (BioOne, Elsevier, IEEE, PMC)
- Face markup applied to whole elements:
  - Is not retained (AIP, BioOne, Blackwell, Elsevier, IEEE, Nature, PMC, Wiley). A style sheet applies it when the content is rendered
  - Is retained (Highwire, UCP) eliminating the need for journal-specific style sheets. For example: `<it><title>Eur. J. Immunol.</title></it>`
- Database links (e.g. Medline PMID):
  - Are retained (AIP, BioOne, Blackwell, PMC, UCP)
  - Are not retained (Elsevier, Highwire, IEEE, Nature, Wiley). They are created when the content is rendered

Most of these structuring approaches also apply to non-journal references — if they were parsed. Many publishers do not structure non-journal references because they are more difficult to parse than journal references, they appear less frequently, and databases to which these references can be linked are less common. As a result, some publishers do not include support for non-journal references in their DTDs (see Table 13 and Section 6.3.1).

Based on the archive mission, ease of rendering and link creation, Inera recommends the following policies for DTD reference structure and usage:

1. *Unique elements should be included to structure these reference types: journals, books, conference proceedings, patents, and electronic references. Based on further analysis, a determination can be made if unique elements are required for reports, theses, and standards, or if they can be treated as a subset of books.*
2. *Structure should be retained for all elements of significant granularity. Retaining significant structure will allow creation of links to databases that are created in the future. A list of recommended structural elements appears in Table 14.*

Table 14: Reference elements that should be tagged in an archive DTD. The Type column indicates the kind of references that may include the element (All = any type, J = Journal, B = Book, C = Conference Proceeding, P = Patent, E = E-Ref).

Element	Type	Description
Number	All	The number of the item in the reference list. If the article citation style is name-date (i.e. "Harvard" style), the number element should not be included.
Author	All	The author(s) of the publication. The author element should include elements for the given name (including initial or middle name), surname, and suffix. A collaboration element should also be available within the author group to mark up organizational authors.
Editor	All	The editor(s) of the publication. This should contain the same sub-elements as Author.

Element	Type	Description
Translator	All	If a work is cited in translation, a reference may list the translator(s) of the publication in addition to or in place of the author(s). The translator element typically includes elements for the given name (including initial or middle name), surname, and suffix.
Et al	All	An element to tag "et al" if the author list was truncated. Et al is occasionally seen twice within a reference for an edited book, once for the chapter authors, and once for the book editors.
Article Title	All	The title of the serial article or book chapter being cited. The title element should have a language attribute in case it differs from the language of the article.
Translated Title	All	The title of a serial article or book chapter being cited that has been translated to the language of the article in which the reference appears.
Publication Title	J B C	The volume title of a work being cited. This element is used to mark up the name of a journal, the title of a book, the specific volume of a book within a multi-volume set, etc.
Series Title	B C	The series name of a multi-volume book.
Publication Date	All	The date of publication. In some cases, there may be multiple dates if an item has been published in multiple manifestations (e.g. print, web) on different dates. The publication date must allow for references to "in press" content.
Volume	J B C	The volume number of the publication (serial or book).
Issue	J	The serial publication issue number.
Supplement	J	The serial publication supplement number.
Edition	B	The book edition number.
First Page	J B C	The first page of the citation.
Last Page	J B C	The last page of the citation.
Other Pages	J B C	Additional pages beyond the first and last page if the citation is to non-contiguous pages.
Publisher Name	B C	The publisher name of a book or conference proceeding.
Publisher Location	B C	The publisher location of a book or conference proceeding.
Conference	C	Conference information for a conference proceeding.
External Link	All	A link to an external location. Today this element typically links to a URL, URI, or DOI number.
Patent Number	P	A patent number found within a patent reference. This element requires an attribute to distinguish the country of the patent number grant.
Comment	All	Any text within the reference that does not fit into any other elements, e.g. "First published in…", "(in Russian)", etc.

3. *The structure must permit elements to appear in any order, allowing the reference to be reproduced in the style of the print journal without applying a style sheet.*

4. *The structure must permit punctuation between elements to be retained, eliminating the need to apply a style sheet while rendering.*

5. *Although we generally recommend retaining all generated text, we believe the application of face markup to entire elements (e.g. volume numbers, journal titles) should be supported but determined by the individual publishers. In most cases, we believe that the content of references will be completely understandable without this face markup, allowing the archive to render references without the need to apply journal-specific style sheets when this markup has not been included.*

6. *We recommend further research into retention of link IDs to databases (e.g. Medline PMID numbers). We are unsure of the archival viability of link data to proprietary databases and we believe links to public databases (e.g. Medline) can be re-established when content is rendered if the database is still operational. If the database cannot be accessed, the archival usage of the ID is questionable.*

7. *The structure should use elements to tag reference content that is rendered. Information that does not appear in print (e.g. PMID numbers), if retained in the DTD, should be structured with attributes rather than elements so it is easier to exclude during rendering.*
8. *The DTD must include a catch-all "other ref" element used for any reference type that is not explicitly supported. While the expectation is that content in these references may be completely unstructured, all elements listed in Table 14 should be available in the other ref element to allow for partial or full structuring.*
9. *Finally, we recommend that the reference section structure allow multiple references to appear within a single numbered reference (as often occurs in Physics journals), and that a reference annotation be allowed to immediately follow each reference (as occurs in Elsevier Current Opinion journals)*

5.7 Tables

With one exception, all surveyed DTDs use either the CALS or Elsevier table models. In addition, all publishers support scanned images for the table body because some tables are too complex to be captured in SGML. Table 15 summarizes table models used by different publishers.

Table 15: Table model by DTD. Most DTDs use either the CALS or Elsevier table model.

DTD	Model
AIP	CALS
BioOne	CALS
Blackwell	CALS
Elsevier	Elsevier 4.0
Highwire	Elsevier 4.0, although some features have been removed. Highwire also allows submission of CALS tables even though the CALS table model is not part of the Highwire DTD
IEEE	ArborText CALS, although by convention IEEE includes only the table title in SGML, and the body of the table is always scanned.
Nature	CALS
PMC	Elsevier 3.0
UCP	ArborText table model
Wiley	CALS (OASIS version)[a]

[a] The most significant difference between the OASIS CALS model used in the Wiley DTD and the CALS adaptation used by AIP, BioOne, Blackwell and Nature is that table footnotes in the latter DTDs are handled with an element <tfoot> that is part of <tgroup>. Wiley handles table footnotes in a manner similar to regular article footnotes.

The CALS model is based on the MIL-M-38784B 910201 DTD originally developed for the US Department of Defense. Over the years a large number of organizations have adopted it. OASIS adopted a simplified version of the SGML CALS table model in the mid 1990's and later modified it for XML use. The OASIS version was created based in part on polling software vendors about which features they supported and potential users about which features they most needed. Because the CALS table model has been widely adopted, a significant number of SGML applications have built-in support for it.

The Elsevier table model was introduced in DTD 3.0. It was modified in DTD 4.1 to allow for handling of certain complex tables and table embellishments that were unsupported in the earlier DTD.

The Elsevier and CALS table models can structure most tables found in journal articles. The models are not completely parallel, however, so some structures may be tagged in the Elsevier DTD (e.g. multiple alignment points within a single cell) that may be difficult to replicate in the CALS DTD.

In some cases, however, neither DTD can adequately represent a table. The most common case is when a graphic appears within a table, and the alignment of surrounding cells to specific parts of the table must be carefully setup. In such cases, most publishers recommend that the table content be incorporated in the SGML as a scanned image rather than tagged SGML. When tables are scanned, most publishers follow the protocol of tagging the table number and table caption in SGML and scanning the table body (including any heading cells and table footnotes).

An SGML archive must be prepared to receive files in which all tables were scanned rather than tagged. This type of delivery may have been done for one of the following reasons:

1. The DTD did not include table support (many early-version DTDs in the mid 1990's excluded table support because the initial focus was to complete full body-text support)
2. The publisher may have decided to treat all tables as scans even though the DTD supports full table tagging because scans of tables may be easier for a publisher to produce.

The CALS table model does not include a standard structure to tag table footnotes, a requirement of most journal publishers. As a result, publishers have implemented their own structures for tagging footnotes and linking those footnotes to citations in the table body. Therefore, even if a common DTD adopts the CALS model for tables, it will still be necessary to create a standard system for handling footnotes, and it will be necessary to convert content from different publishers into this standard system.

All of the table models support left, right and center alignment of text in table cells. Most DTDs support alignment with a specific character (sometimes called "decimal alignment" because it's most commonly used to align a column of numbers by placing the decimal points in a vertical line). The PMC DTD and early versions of the Highwire DTD do not support character alignment because HTML lacks this capability. As a result, it may be difficult to render some tables with character aligned text.

Inera recommends an archive DTD include both the CALS and Elsevier DTD 4.x table models. Support for two table models will significantly ease the burden of archive deposit for publishers, allowing them to submit SGML with much less work for table transformation. Decimal alignment for tables is essential for good quality rendering.

The archive DTD must support scanned table bodies for both CALS and Elsevier models. Finally, we recommend adoption of the common `<tfoot>` *extension to handle CALS table footnotes.*

5.8 Math

Published math comes in two basic forms: inline math and display math. Inline math describes simple equations that appear in the running text flow. Display math describes more complex equations that appear on their own line or in their own visual block.

Most inline math is quite simple (e.g. "$x^2 + y^2 = z^2$"). It can usually be represented with Unicode character values and face markup (italic, bold, superscript and subscript). Many publishers do not tag such equations as mathematical expressions, although some publishers (Elsevier) request that these expressions be tagged.

In a few cases, inline math may be more complicated. For example stacked elements, $a_j T_j^i = b$, a simple summation, $\sum_{i=1}^{n} b_i = z$, or a radical, $\sqrt[n]{x+y} = z$ can appear inline.[17] In these cases, the equation must be tagged using SGML markup rather than simple face markup.

All surveyed DTDs include a model for encoding display math and complex inline math with a stream of text commands. Four primary encoding models are used by the surveyed publishers: 12083, Elsevier, MathML, and TeX.[18] These models are summarized in Table 16.

[17] Many editors contend that these examples should be display equations, not inline equations, however an archive must be prepared to handle SGML that was not created according to standard editorial practices.

[18] We carefully avoid the word "tagging" here because TeX and LaTeX are not SGML or XML.

Table 16: Math Models by DTD.

DTD	Model
AIP	ISO12083:1993
BioOne	ISO12083:1993
Blackwell	MathML (W3C, 7 April 1998)
Elsevier	Elsevier Math
Highwire	Elsevier Math
IEEE	TeX
Nature	ISO12083:1994
PMC	TeX
UCP	Based on ArborText's implementation AAP Math with further changes by UCP.
Wiley	TeX or LaTeX in external file

Because 12083, AAP and Elsevier math are structural cousins, there are actually three primary math models: 12083, MathML, and TeX. In addition, most publishers (but not all, e.g. UCP) support scanned images for math because some equations may be too complex to be captured in SGML.[19]

AAP Math is the original foundation of SGML math markup for most journal publication. 12083 math, although not directly derived from the AAP DTD, was developed in part based on a review of the AAP DTD. Elsevier's math model is more closely related to AAP although certain semantic constructions, such as explicit integrals and products, have been dropped. Because AAP, 12083, and Elsevier math all share a common structural foundation, Inera expects conversions between these DTDs will not present significant problems for publishers.

MathML is a newer math model, developed for XML rather than SGML. Unlike 12083 math, which is strictly concerned with presentation markup, MathML can be used for presentation or content markup. "The intent of the content markup in the Mathematical Markup Language is to provide an explicit encoding of the *underlying mathematical structure* of an expression, rather than any particular rendering for the expression."[20]

MathML first became a W3C recommendation in February 1998, and version 2.0 became a W3C recommendation on February 21, 2001. Most surveyed publishers do not use MathML because they developed their DTDs prior to the original MathML recommendation, and they have not converted their DTDs to XML. Only one surveyed publisher uses MathML (Blackwell). Several publishers have indicated their long-term intent in conversations with Inera to migrate from 12083 math to MathML.

Neither 12083 math nor MathML can be natively displayed in most current browsers.[21] As a result, when publishers prepare full-text SGML for online presentation, the equations are converted to an image, usually in GIF format.

TeX[22] is a powerful typesetting system created by Donald Knuth of Stanford University in 1981, prior to the creation of SGML. Like SGML, TeX files are ASCII-coded representations. Leslie Lamport developed LaTeX[23], a 'dialect' of TeX in 1985. LaTeX is particularly suited to the production of long articles and books, since it has facilities for the automatic numbering of chapters, sections, theorems, equations etc., and also has facilities for cross-referencing.

Because TeX and LaTeX have been around so long, and because they provide tremendous typesetting facilities for complex expressions, they are widely used in the mathematical community. Many publishers have chosen to retain math in TeX or LaTeX rather than convert it to SGML.

There are many tools that convert TeX to GIF images for web presentation. These tools have encouraged some publishers to stick with TeX and bypass SGML for tagging of math. In fact, some organizations

[19] Inera is familiar with DTDs outside of the scope of this study in which all display equations are represented as scanned images. Publishers may choose this strategy when their content does not contain many equations.

[20] W3C MathML Recommendation, section 4.1.1, http://www.w3c.org/TR/MathML2/chapter4.html, accessed on October 31, 2001.

[21] W3C's Amaya browser can be used to render MathML. See http://www.w3.org/Amaya/.

[22] For more information and links about TeX, see http://www.ams.org/tex/ and http://www.tug.org/.

[23] For more information and links about LaTeX, see http://www.latex-project.org/.

prepare SGML for the web by converting SGML to TeX and then using a TeX to GIF converter to create a graphic file for each equation.

Conversions from 12083 math and MathML to TeX are not especially difficult, however TeX is not SGML and therefore not recommended as a primary archive format for equations. Conversions, however, from TeX to 12083 math or MathML, and conversions between 12083 math and MathML are more difficult to develop.

Most surveyed publishers also permit equations to be captured as scanned images rather than SGML or TeX encoding. While the scanned image route is usually intended for equations that cannot be captured with the available encoding, some SGML suppliers use scanned images because they are easier to create than text-encoded equations. While this approach preserves the exact visual appearance of an equation, it is not the preferred solution because it creates additional external files that are much larger than text-encoded math, and it precludes the possibility of re-formatting equations at a later date.

Inera is reluctant to place the burden on multiple publishers to develop complex math transforms between markup models that may be difficult to complete with a high level of quality. Therefore Inera recommends an archive DTD use a tripartite approach and allow 12083 math, MathML and TeX in document instances. This solution presents a path of least resistance for publishers while also providing the highest quality original math markup to the archive.

During DTD development, we recommend a careful study of the 12083, AAP and Elsevier math DTDs to determine extensions that might be made to the 12083 math model (e.g. limit constructions) for easier transformation of AAP and Elsevier math into 12083 math.

Scanned math must be permitted for those equations that are supplied as scanned images rather than encoded in SGML or TeX.

For the purposes of archive rendering, Inera believes that simple inline equations need not be tagged. The loss of tagging in such cases is tolerable.

5.9 Chemical Formulas

None of the surveyed DTDs include capabilities to markup chemical formulas. Further analysis is required to determine if markup of chemical formulas is required in an archive DTD.

5.10 Special characters

All of the publishers included in this survey, except Elsevier, base their non-ASCII character sets on the standard ISO entity sets. Most publishers use only a subset of the entity files listed in Table 17. The selected subset is usually based on the needs of the content being published.

Table 17: List of ISO character sets. In cases where a set was updated in 1991, publishers use one of either the 1986 or 1991 set. Some publishers still use the 1986 files rather than the 1991 files.

File	Public Identifier
ISOamsa	"ISO 8879-1986//ENTITIES Added Math Symbols: Arrow Relations//EN" "ISO 9573-13:1991//ENTITIES Added Math Symbols: Arrow Relations //EN"
ISOamsb	"ISO 8879-1986//ENTITIES Added Math Symbols: Binary Operators//EN" "ISO 9573-13:1991//ENTITIES Added Math Symbols: Binary Operators //EN"
ISOamsc	"ISO 8879-1986//ENTITIES Added Math Symbols: Delimiters//EN" "ISO 9573-13:1991//ENTITIES Added Math Symbols: Delimiters //EN"
ISOamsn	"ISO 8879-1986//ENTITIES Added Math Symbols: Negated Relations//EN" "ISO 9573-13:1991//ENTITIES Added Math Symbols: Negated Relations //EN"
ISOamso	"ISO 8879-1986//ENTITIES Added Math Symbols: Ordinary//EN" "ISO 9573-13:1991//ENTITIES Added Math Symbols: Ordinary //EN"
ISOamsr	"ISO 8879-1986//ENTITIES Added Math Symbols: Relations//EN" "ISO 9573-13:1991//ENTITIES Added Math Symbols: Relations //EN"
ISObox	"ISO 8879:1986//ENTITIES Box and Line Drawing//EN"
ISOcyr1	"ISO 8879:1986//ENTITIES Russian Cyrillic//EN"
ISOcyr2	"ISO 8879:1986//ENTITIES Non-Russian Cyrillic//EN"
ISOdia	"ISO 8879-1986//ENTITIES Diacritical Marks//EN"
ISOgrk1	"ISO 8879-1986//ENTITIES Greek Letters//EN"
ISOgrk2	"ISO 8879:1986//ENTITIES Monotoniko Greek//EN"
ISOgrk3	"ISO 8879-1986//ENTITIES Greek Symbols//EN" "ISO 9573-13:1991//ENTITIES Greek Symbols //EN"
ISOgrk4	"ISO 8879:1986//ENTITIES Alternative Greek Symbols//EN" "ISO 9573-13:1991//ENTITIES Alternative Greek Symbols//EN"
ISOlat1	"ISO 8879-1986//ENTITIES Added Latin 1//EN"
ISOlat2	"ISO 8879-1986//ENTITIES Added Latin 2//EN"
ISOmfrk	"ISO 9573-13:1991//ENTITIES Math Alphabets: Fraktur//EN"
ISOmopf	"ISO 9573-13:1991//ENTITIES Math Alphabets: Open Face//EN"
ISOmscr	"ISO 9573-13:1991//ENTITIES Math Alphabets: Script//EN"
ISOnum	"ISO 8879-1986//ENTITIES Numeric and Special Graphic//EN"
ISOpub	"ISO 8879-1986//ENTITIES Publishing//EN"
ISOtech	"ISO 8879-1986//ENTITIES General Technical//EN"

None of the surveyed publishers found all of the special characters they require within the ISO character sets. As a result, all of the publishers have their own custom entity files used to meet the specific needs of their publications. These custom entities are typically included in one or more auxiliary entity files.

The Elsevier entity set has a significant overlap with the ISO entity set, however it also has many characters that are unavailable in the ISO set. Additionally, accented letters in the Elsevier DTD are tagged rather than encoded with entities. For example the letter "é" is encoded as `&eacute;` in the ISO entity set and tagged as `<a>e<ac>acute</ac><a>` in Elsevier SGML.

None of these publishers use Unicode as the foundation for their sets. ISO entities became the preferred entity encoding because:

1. ISO 12083 uses ISO entities
2. ISO entities are more readable than Unicode entities when reading and writing SGML directly.
3. Unicode was not a firmly established standard when most journal publishers began their SGML initiatives. Unicode was adopted in conjunction with markup languages when it became the standard encoding of non-ASCII characters in XML.

A large percentage of the ISO, Elsevier and custom publisher entity sets can be mapped directly to Unicode values. The most definitive cross-reference we have found between different representations of special characters is the file Unicode.xml, created as part of the STIX project.[24] Besides mapping between Unicode, ISO and Elsevier, this file also includes mappings for LaTeX and AMS, among others.

Unfortunately different character sets do not have a one-to-one correspondence. In particular, the ISO Greek entities can be a source of confusion when converting characters from one standard to another. The most lucid description of these issues can be found in the Wiley DTD documentation.[25] ISO provides distinct entities for Greek text and Greek math symbols. Wiley maintains this distinction although most other publishers do not.

Inera recommends that an archive DTD use Unicode for the encoding of all non-ASCII characters. Although Unicode is less than perfect, it provides the largest and least-unambiguous character set for the encoding of special characters.

The distinction between the different Greek ISO character sets is significant in some typographic contexts, but we do not believe that this distinction need be maintained in an archive DTD.

Special characters that cannot be represented within Unicode 3.0 should be handled in one of the following ways:

1. *During archive DTD development, publishers (especially those of math-intensive journals) should be polled about their needs for commonly used characters that are outside of the Unicode standard. The results of the STIX project can serve as the foundation for this poll. An additional file of entities should be defined to handle these common non-Unicode special characters.*
2. *Any additional special characters that are outside of Unicode or the additional entity files must be submitted to the archive as graphic images.*

[24] http://www.ams.org/STIX/.

[25] JWS_GUIDE for DTD 3.1, version 1.0, September 30, 1998, pages 12-14.

6 Links

The ability to tag, preserve and navigate hyperlinks is one of the most compelling uses of SGML. This section describes the types of links that must be preserved in the archive and discusses specific markup and usage issues pertinent to different link types. The links are divided into two distinct groups: intra-document links for links within documents or to objects such as graphics that are part of the document, and inter-document links for links to objects outside of the document.

6.1 Intra-document Object Links

Intra-document links are used primarily to link text to related places or objects in a document. Examples of links include:

- Appendix citation to appendix.
- Author to affiliation or vita
- Box citation to text box title
- Corresponding author to corresponding author footnote
- Equation citation to display equation
- Figure citation to figure. Other image objects, e.g. plates, schemes, diagrams, etc., are part of this same class
- Footnote citation to footnote
- List citation to list
- Reference citation to reference in bibliography
- Section number or name to section head
- Table citation to table title
- Table footnote citation to table footnote

For each of these link types, there will be one or more citations, which we will call the "source", and the cited object, which we will call the "target". For example, "see Figure 1" is a source citation and its corresponding target object is the figure whose caption starts "Figure 1".

In SGML and XML, the source and target are linked with attributes in their respective elements. A source element has an attribute with the type IDREF, and a target element has an attribute with the type ID. If the ID and IDREF have identical values, a link is established.

Table 18 shows how the source "see Figure 1" is tagged by each of the surveyed publishers. In each case, there is an IDREF attribute (`rid`, `refid`, `fid`, `href`) that creates the link to the target.

Table 18: Examples of figure citations (also known as callouts). Note the use of a specific figure citation element by some publishers, and a generic citation element by others.

Publisher	Print Example	Corresponding SGML
AIP	see Fig. 1	`see Fig. <figref rid="F1">1</figref>`
BioOne	see fig. 1	`see <FIGREF RID="i0003-0082-294-01-0001-f01">fig. 1</FIGREF>`
Blackwell	see Fig. 1	`see <link rid="f1 f1_legend_span">Fig.&thinsp;1</link>`
Elsevier	see Fig. 1	`see <cross-ref refid="fig1">Fig. 1</cross-ref>`
Highwire	see Fig. 1	`see <cross-ref refid="fig1" type="fig">Fig. 1</cross-ref>`
IEEE	see Fig. 1	`see <xref rid="fig1" type="fig">Fig.&nbsp;1</xref>`
Nature	see Fig. 1	`see <FIGR RID="f1">Fig. 1</FIGR>`
PMC	see Figure 1	`see Figure <figr fid="F1">1</figr>`
UCP	see Figure 1	`see <FIGREF RID="fg1" PLACE="YES">Figure 1</FIGREF>`
Wiley	see Fig. 1	`see Fig. <FIGR HREF="fig1">1</FIGR>`

The amount of text placed inside the source element varies by publisher. Note that unlike the reference citations shown in Table 7, there is no generated text in these examples. The degree of generated text required by publishers for link sources varies depending on the type of object referenced.

A key difference between link source elements is the type of element used. The 12083 DTD has a unique citation element for each object type, i.e. `figref` for figure references, `tableref` for table references, `citeref` for bibliographic references, etc. Most surveyed publishers have followed this model.

Blackwell, Elsevier, Highwire, and IEEE tag the source with a generic element (`link`, `cross-ref`, `xref`) rather than a specific element (`figref`, `figr`). Highwire and IEEE identify the type of target object with an attribute. Elsevier and Blackwell identify the type of target object by publisher-standard letters used in the reference ID.[26]

The target must contain an ID that corresponds to the IDREF in the reference source (e.g. `f1`, `fig1`). Table 19 expands the information shown in Table 6 and illustrates an entire figure object for each surveyed publisher.

Table 19: Examples of tagging a figure object. Target objects include an ID that is cited by an IDREF, and often a number element with the text of the object number. Because figures are external objects, there is an element or attribute that contains the file name for the image.

Publisher	Figure Object
AIP	`<figgrp id="F1"><title>Caption text… </title></figgrp>`
BioOne	`<FIGGRP ID="i0003-0082-294-01-0001-f01"><FIG FILE="i0003-0082-294-01-0001-f01"><TITLE>Fig. 1. Caption Text… </TITLE></FIGGRP>`
Blackwell	`<object position="floating" type="figure" id="f1"><objectsource><file id="f1_source" name="bjd3198.f1.gif" type="gif"/></objectsource><legend id="f1_legend"><p><span id="f1_legend_span"><number>Figure&thinsp;1.&ensp;&ensp;</number>Caption Text... </span></p></legend></object>`
Elsevier	`<fig id="fig1"><no>Fig. 1</no><caption><p>Caption Text...<link locator="gr1"></fig>`
Highwire	`<fig id="fig1"><no>Figure 1. </no><caption><p>Caption Text...</p></caption><link locator="JCB03035f1"></fig>`
IEEE	`<figgrp id="fig1" span="col" type="fltop" mode="portrt"><fig name="bunn1.tif"><title just="just" autonum="off">Fig. 1.&emsp;Caption text...</title></figgrp>`
Nature	`<FIG ID="f1" ENTNAME="figf1"><!--nbt0901-838-F1--><CAPTION><P>Caption text...</P></CAPTION></FIG>`
PMC	`<fig id="F1"><title><p>Figure 1</p></title><caption><p>Caption text...</p></caption><graphic file="1471-2180-1-1-1" hint.layout="single"/></fig>`
UCP	`<FIGGRP ID="fg1" COLS="1" WIDTH="WIDE" PLATE="NO"><FIGURE><GRAPHIC FILENAME="fg1.tiff" TYPE="EPS" MAG="100" ROTATE="0" DPI="300" VERTADJPCT="-25"></FIGURE><LEGEND><LABEL>Fig. </LABEL><NO>1.&mdash;</NO><P>Caption text... </P></LEGEND></FIGGRP>`
Wiley	`<FIG ID="fig1" LOC="FLOAT"><GRAPHIC NAME="fig001" COPYRIGHT="Wiley-Liss, Inc." ERIGHTS="1"></GRAPHIC><NUMBER>1</NUMBER><CAPTION><P>Caption text...</P></CAPTION></FIG>`

The figure object contains:

- An element that identifies the object as a figure. Note that in all DTDs except Blackwell, the element is `fig` or `figgrp`. Blackwell uses `object` with a `type` attribute to indicate it's a figure.
- The ID corresponding to the source figure reference. In all cases, the attribute is called `id`.
- The text with the target object number (only in some DTDs). See Table 6 and Section 5.1.1.

These three items are structurally intrinsic to most target objects. In addition, target objects may contain object-specific information. For example, the figure object typically contains linking information to the figure file, the figure caption, and possibly rendering information.

[26] It is worth noting that both Blackwell and Elsevier used `<figr>`, etc. in their earlier DTD versions, and switched to a generic source in DTD updates.

The figure IDREF/ID pair illustrates one type-case. A broader review of other source/target types (e.g. tables, footnotes, etc.) indicates the links can be retained for all surveyed DTDs.

An archive DTD must be able to preserve all intra-document links that have been tagged in publishers' SGML files. However, an archiving institution should be prepared to receive documents in which these links are not tagged because the publisher (or the publisher's supplier) did not tag them in the original SGML. For more information on this problem, please see Section 10.5.

Inera recommends a generic link element (e.g. `<xref>` *rather than* `<figr>`, `<tabr>`, *etc.) at the source citation point because it is more easily extended for new object types. In addition, we recommend using a type attribute to indicate the target object type rather than inferring the type from the attribute value. The generic link element should include all generated text. This model is similar to that of IEEE.*

We recommend target object elements be specific to the type of object (e.g. `<fig>`, `<table>`, `<bib>`, *etc.) because different objects have different properties. Recommendations for the object number are described in Section 5.1.1. We do not recommend the retention of specific object layout information (e.g. number of columns or layout style for figures) that appears in several DTDs.*

The policy for extent of text included in the source reference element (i.e. "Fig 1" vs. "1") should be considered during DTD development. Setting a standard will result in more consistent rendering of archive documents, however it may place a burden on the publishers during document conversion.

6.2 Intra-document Sub-file Links

Figures, scanned tables, and images of math comprise the bulk of external files referenced by SGML document instances. All surveyed publishers expect these files to be in the same directory as the SGML document instance.

There are two main styles of file reference:

1. The name of the file is placed in an attribute, and the rendering software loads the file. BioOne, Blackwell, Highwire, IEEE, PMC, UCP, and Wiley use this model.
2. The name of an entity is placed in an ENTITY attribute of the appropriate (e.g. figure) element, and an entity declaration appears at the top of the file to define the name, location, and nature of the external file. AIP, Elsevier, and Nature use this model.

Inera prefers the use of an ENTITY attribute because the model is more robust, however it may place a burden on the publishers during document conversion. For this reason, we recommend that a survey of publishers be conducted during analysis for DTD development to determine the most appropriate solution.

Many operating systems (e.g. Windows) do not respect case-sensitivity in file names. For example, "GRAPHIC.TIF" is considered the same file as "graphic.tif" or "Graphic.tif". Other operating systems, such as UNIX, consider these three different files.

Because of different operating system behaviors, Inera recommends that the name of file references in archive XML files must exactly match the case of the file names for graphic files submitted to the archive. It may be prudent to consider requesting that all file names use only lower case letters to aid in assuring matches between file links and the actual files. Finally, we recommend that only ASCII characters be permitted in file names to avoid translation problems when moving files between operating systems.[27]

6.3 Inter-document Links

Inter-document links allow connections to several kinds of resources: bibliographic links, supplementary material, related documents, and external databases. Each of these link targets has unique issues.

[27] It may be necessary to limit the character set even more because some ASCII characters are not valid in file names on certain operating systems.

6.3.1 Bibliographic Links

Links can be established from bibliographic references to databases such as Medline or CrossRef. Establishing these links for journal articles is relatively easy, provided the bibliographic references are correctly structured. In most cases, if the journal title, volume, year, and page are provided, these databases can be queried to establish a link to either an abstract or the full text of an article.

Attempts to establish journal article links may fail for several reasons:

1. The accuracy of the reference is critical. If the journal title is incorrect or is abbreviated incorrectly, or if the volume, year or first page is incorrect, a link cannot be established.[28]
2. The key elements in references may be unstructured in the SGML file, or they may be incorrectly structured due to an error in SGML creation.
3. Some databases, such as Medline, do not have sufficient data to resolve all references unambiguously. Ambiguity problems are most prevalent when journal supplements are cited because Medline queries do not allow the supplement information.

In spite of the problems, journal references are the easiest bibliographic material to link. Links are more difficult to create for monographs, edited books, conference proceedings, and other non-journal content because:

1. Some DTDs do not support structured non-journal references.
2. Some publishers leave non-journal references unstructured, even when there is DTD support. They do this because non-journal references are more complicated to structure than journal references and because databases to link these references are less common than databases to link journal references

For these reasons, an archive must expect a much lower success rate when establishing links to non-journal content.[29]

An archive must use the data in structured references to establish bibliographic links. Because available databases may change over time, Inera recommends keeping references as fully structured as possible in the archive XML files. The structured archived reference data can be used in the future to look up links in new databases.

An archive may be able to use a service such as Medline or CrossRef to establish bibliographic links. If these services are unavailable at a future date, the archive can create a reference resolution system for all journal articles that are available in the archive, although we recommend this route only as a solution of last resort.

6.3.2 Supplementary Content

Supplementary content encountered with SGML files includes a wide variety of formats. The most common formats are Microsoft Office (Excel, Access, PowerPoint), WAV, RealAudio, MP3, QuickTime, and RealVideo.[30] Because these files are binary format, they may be problematic for an archive because utilization software may be unavailable at the time these files are retrieved from an archive. Further discussion of this issue is beyond the scope of this report.

Table 20 summarizes the methods publishers use to create links from SGML files to supplementary content. Publishers encode these links in a wide variety of ways, and many use proprietary data as part of the link information.

[28] Inera's experience has shown that author errors cause linking failures in a significant number of instances. Few journals have procedures to find and correct many of these errors during the editorial and production process.

[29] CrossRef will begin accepting deposits of metadata for books and conference proceedings in 2002 for the purpose of reference linking. We hope this capability will spur publishers to focus more effort on structuring non-journal references.

[30] Bide, Mark. *Standards for Electronic Publishing*. 2000. http://www.kb.nl/coop/nedlib/results/e-publishingstandards.pdf.

Table 20: Methods for linking to supplementary content by DTD.[31]

Publisher	Supplementary Content Link
AIP	`supmatl` element in the article front matter
BioOne	`supmatl` element in the article front matter, or possibly `xref` element with a `file` attribute.
Blackwell	`externallink` element at location of supplementary content citation or within `relatedgroup` in the article front matter. A wide range of possible links, including links to files, URLs, databases, etc. can be made through these links. Setting the `type` attribute of `externallink` specifies how the `id` attribute is to be interpreted.
Elsevier	`upi` (unpublished item) element at location of supplementary content citation. It may contain a number, caption and link to an external file located in the same directory as the SGML document. It also allows for copyright that differs from the article copyright.
Highwire	Links to supplementary content are maintained in a supplemental data manager rather than the SGML file.
IEEE	`linkto` element with `objidref` element that contains a file link
Nature	`suppobj` element in the article front matter. `suppobj` objects maintained within the Nature system are referenced via Nature-specific ID values. For example: `<SUPPOBJ EXTREFID="nn0901-927-S2" FORMAT="MOV" FILESIZE="3547K"><DESCRIP><P><B>Ataxic.mov</B>: The same rat ...</P></DESCRIP></SUPPOBJ>` The path to `suppobj` objects maintained outside the system (e.g., at the author's website) is specified as a URL.
PMC	`suppmat` is the publisher-supplied TOC link for supplemental material to an article
UCP	UCP stores information for linking to supplemental material in URL tags with addresses pointing to files on their site. The `supmatl` element in the front matter stores descriptive information about supplemental materials. For example: `<SUPMATL><LABEL>Accompanying videotape:</LABEL>ApJ, 495, Part 1, Number 1, Segment</SUPMATL>`
Wiley	`ADDLMAT` element at the location of supplementary content citation. It may contain text to describe the file, and a link to the file in the form of a URL, URI, or DOI.

Because many publishers use proprietary data for some or all of the critical linking information, Inera believes an archive may have trouble re-establishing links in some cases, even if the DTD model for linking is flexible.

Inera recommends these characteristics for a supplementary content linking model:

1. *The model must not use proprietary data to maintain links. We recommend that the link be a file name if the file is in the same directory as the SGML instance. If the file is in a different location, we recommend the link be a URI or DOI, however the model must be open to allow for new standard linking models that appear in the future. Publishers who use proprietary systems must convert to a public system during SGML transformation for archive deposit.*
2. *The link element must allow a text description of the file and an optional still preview image of the opening screen of the file.*
3. *The link element must include format attributes to describe the type of data in the file and the application used to render or utilize the file. Further research must be done on appropriate archival coding systems for this information.*

[31] Inera believes this table is substantially correct, however very few examples of links to supplementary files were found in sample files provided by publishers, and in a number of cases the documentation for use of these elements was incomplete.

4. *The document model must allow the links to be added in the front matter of the article or embedded in the body text at the location where the supplementary content is cited.*

During DTD development, we recommend discussion with publishers and further review of sample files to help determine the final model for links to supplementary content.

6.3.3 Related Documents

Linking to related documents shares many of the same issues with linking to supplementary content. Table 21 summarizes the methods publishers use to create links between related SGML files. Publishers encode these links in a wide variety of ways, and many use proprietary data as part of the link information.

Table 21: Methods for linking to related documents in the same journal.[32]

Publisher	Related Document Link
AIP	`link` element with `type` and `target` attributes in the article front matter. The target value is proprietary to AIP
BioOne	`xui` element with `xdb` and `ui` attributes.
Blackwell	`related` element (in front matter) with attributes that point to a file, or if the item is in the same issue, an attribute that points to the page number
Elsevier	`refers-to` and `refers-to-doi` attributes in the `art` element. The `refers-to` attribute contains one or more PII numbers.
Highwire	`addart` element with `type`, `vol` and `pg` attributes. This element can appear in the article front matter, or it can be applied to text that cites the related document, e.g. "`The figure caption in <addart type="err" vol="101" pg="352">volume 101, page 352</addart> was incorrect…`" in an erratum.
IEEE	`linkto` element with `objidref` element that contains the file link
Nature	`nvid`, `xref`, `xnav`, `xrespond`, `xrelate`, and `errorcor` elements in the article front matter. The attribute value with the link is proprietary
PMC	`addart` and `relart` elements in the article front matter. These elements have rich linking attributes because they include the `ui` (unique identifier) which may have types of `(aid | artnum | doi | pmid | pmc)`
UCP	`uri-orig` attribute of the `UCP-Article` element: linking information of the original article (used with erratum and addendum)
Wiley	`HREF` or `XML-LINK` attribute of `ERRATUM` element for errata, addenda, corrections, etc.

Because many publishers use proprietary data to link related documents, Inera believes an archive may have trouble re-establishing links in some cases. In other cases, we expect related document links may not be tagged because there is no DTD support (e.g. some DTDs only support related document links for errata, but not other document types). Finally, the links may not be tagged because these elements are not commonly used and the SGML supplier may have overlooked the tagging.

Inera recommends these characteristics for a related document linking model:

1. *The model must not use proprietary data to maintain links. We recommend the link be a DOI, because it is the most widely accepted standard to identify journal articles. The model must be open to allow for new standard linking models that appear in the future. Publishers who use proprietary systems must convert to a public system during SGML transformation for archive deposit.*
2. *For those articles that have not been identified with DOI numbers, we recommend considering a volume and page number for linking to articles in the same journal. If the article is in a different journal, and no DOI is available, we recommend using a volume, page, and journal name to establish the link. However, this structure requires that the archive establish a mechanism to enable these links.*

[32] Inera believes this table is substantially correct, however very few examples of links to related documents were found in sample files provided by publishers, and in a number of cases the documentation for use of these elements was incomplete.

3. *The link element must allow a text description of the link target (e.g. "The figure caption in volume 101, page 352 was incorrect…") and an attribute to describe the type of content found at the target (e.g. errata, letter, in this issue, etc.).*
4. *The document model must allow the links to be added in the front matter of the article or embedded in the body text at the location where the related document is cited.*

6.3.4 External Databases

Certain disciplines have standard databases of information that authors may cite in their documents. In most cases, these databases are accessed by standard format numbers for which links can be created. For example Genbank (http://www.ncbi.nlm.nih.gov/) is the NIH genetic sequence database and it holds an annotated collection of all publicly available DNA sequences.

Some DTDs may not support links to discipline-specific databases. In such cases, references to these databases will be untagged in SGML files. Other DTDs provide rich tagging capabilities for references to these discipline-specific databases. Even when a DTD supports tagging database citations, the SGML supplier may not have tagged these elements (see Section 10.5 for a case study of such problems).

An archive will receive files, whether in a publisher's DTD or the archive DTD, in which links to external databases have not been tagged by the supplier.

Inera recommends archive DTD support for external databases with a generic element (e.g. `<external-link>`*). The element should include an attribute to describe the database type. This attribute should be CDATA to allow maximum flexibility and expansion, however the accompanying documentation should provide recommended values for commonly cited databases.*

7 Graphic Files

Graphic files must be supplied to the archive along with the SGML content. Table 22 shows the graphic file formats used by publishers. TIFF is the most universally accepted format, followed by EPS (Encapsulated PostScript) and then JPEG. Several other formats are listed for some publishers, however we suspect these formats are found less frequently in print production or SGML archive formats. They are more likely seen in preparation of data for the web, especially GIF.

Table 22: Graphic file formats accepted by publishers.

Publisher	Graphic File Formats
AIP	TIFF, JPEG, BMP, JPG, GIF
BioOne	TIFF, EPS
Blackwell	TIFF, JPEG, BMP, GIF, EPS, WMF, PICT
Elsevier	TIFF, JPEG, EPS
Highwire	TIFF, JPEG, EPS
IEEE	TIFF, EPS
Nature	TIFF, JPEG, BMP, GIF, EPS, PICT
PMC	TIFF, JPEG, BMP, GIF
UCP	TIFF, EPS
Wiley	TIFF, EPS, GIF, JPEG

For archival purposes, TIFF and JPEG are excellent formats. Both TIFF and JPEG are non-proprietary and fully documented, have good compression properties, and they are bitmap (as opposed to object-based) formats for which rendering code can easily be written from the specifications even if there is no rendering engine available for a future computer system.

EPS is a widely used format that also has excellent compression properties, however rather than being a simple bitmap format, it is a complex object-based format driven by the proprietary PostScript language. We have concerns that PostScript interpreters, which are very difficult to develop, may be unavailable on future computers used to render archived articles.

Inera recommends that all graphics are deposited in TIFF or JPEG format. During DTD development, full specifications for graphic delivery must be developed including specifications for graphic resolution and image color.

We recommend further research into the risk of associated with rendering EPS graphics. If there is sufficient risk that software may not be available, the archive may require conversion of these graphics to TIFF or JPEG.

All remaining graphic formats should be converted to TIFF or JPEG for archival deposit.

8 Publisher-Specific Elements

This report has reviewed the intersection of major elements found in publisher DTDs and presented recommendations for how they may be treated in an archive DTD. There are additional, often less frequently used elements that are found in one or more DTDs reviewed in this survey. A random sampling of such elements includes:

- `acidfree` (AIP): Acid-free paper indicator
- `synonymy` (BioOne): Use unknown - no documentation provided.
- `rating` (Blackwell): A rating assigned to a document.
- `stereochem` (Elsevier): A group of elements use to capture the chemical compound in a stereochemistry abstract.
- `response` (Highwire): An element used to capture the reply(s) to a letter to the editor.
- `obitgrp` (IEEE): An element containing the name, title, years and picture of the subject of an obituary.
- `greeting` (Nature): The greeting line in a letter to the editor.
- `shortabs` (PMC): Short abstract
- `speech` (UCP): Used to markup dialog/interviews
- `cartoon` (Wiley): The Wiley DTD distinguishes between different figure types with unique elements rather than an attribute. DTD elements are `cartoon`, `chart`, `diagram` , `drawing`, `exhibit`, `fig`, `illus`, `map`, `plate`, `scheme`, and `workflow`

Within the scope of this report, it is not possible to examine every element in every DTD to determine its importance in an archive DTD, and we are not providing recommendations on the elements presented above. This task is best left to the document analysis for and development of an archive DTD. Criteria based on the mission of an archive DTD, rendering and linking, should be used to evaluate each element. In addition the tradeoff between tolerable loss and DTD complexity should be evaluated for each publisher-specific element that is added to the DTD.

Specific elements not withstanding, Inera recommends an archive DTD fall between the intersection and the union of structural elements found in the surveyed publisher DTDs. It must be broad enough to capture the key structural elements common to most publishers, however it will exclude many elements that are unique to specific publishers because the content of those elements can be adequately rendered through tagging with more generic (i.e. less restrictive) elements. Exclusion of many publisher-specific elements will simplify the DTD and processes for SGML conversion and rendering.

The elimination of some publisher-specific elements when creating the archive DTD will cause lossy conversions when publisher SGML is transformed to archive XML. Inera believes that a carefully constructed DTD can limit the loss to tolerable loss rather than significant loss within the scope of the archive mission.

This loss, while insignificant for the vast majority of archive clients, may be significant to a small number of future researchers. For example, researchers who study the history of the transformation from print to electronic publishing may find certain data in publisher SGML files that is significant to their research, but insignificant to the archive mission of rendering intellectual content.

The needs of these researchers can be met by depositing two manifestations of each article in the archive. The first manifestation will be an XML file created in the archive DTD. The second manifestation will be the publisher's original SGML/XML file. The instance created with the archive DTD will allow full and easy access to the intellectual content of the article. The original instance will provide the full resolution granularity and metadata of the publisher's original file. Our expectation is that researchers will provide their own tools to study the publisher's files. We do not anticipate the archive will build systems to render these files.

The inclusion of two manifestations for each article in the archive requires attention to additional issues beyond the scope of this report including completion checks for file delivery, naming conventions, and resolution of figure names so that only one set of figures need be deposited. These issues not withstanding, Inera recommends the inclusion of two manifestations of each article in an archive to allow use of full-resolution SGML by a small number of researchers for whom this difference will be significant.

9 DTD Documentation

Consistent high quality SGML starts with clear and comprehensive documentation. The documentation combined with sample files provided by the DTD author(s) serves as interpretation of the DTD.

By way of comparison, a DTD, like the Bible, is a relatively short document that is rich in meaning. However, each religion that reads the Bible has its own interpretation and that commentary is far more voluminous than the Bible itself. Similarly, many organizations may use similar DTD structures but interpret them in different ways, requiring significant documentation.

For example, a look at the Elsevier and Highwire DTDs shows some elements with the same data model. However if we compare SGML created with these DTDs when elements have the same model (e.g. Table 6 and Table 7), we find differences according to each publisher's interpretation. So we have come to find that solid DTD documentation to explain these interpretations is often far larger than the DTD itself.

Table 23 summarizes the documentation that was received from each of the participating publishers. The documentation defines how to interpret the DTD and allows suppliers to provide more consistent SGML. By setting publisher standards, documentation avoids the issue of "personal preference" for tagging of elements. By removing personal preferences, SGML will be more consistent.[33]

Table 23: DTD documentation provided by publishers.

DTD	Available Documentation
AIP	1 Word file (34 pages)
BioOne	(none provided)
Blackwell	DTD website (see bibliography) or HTML Help file (1,017 MB)
Elsevier	Published book "Tag by Tag", 312 pages
Highwire	Web page (118k)
IEEE	13 PDF files (146 pages)
Nature	1 Word file (67 pages)
PMC	(none provided)
UCP	5 PDF files (225 pages)
Wiley	1 PDF file (56 pages) and 3 RTF files (558 pages)

During our years of implementing SGML systems, we have read, interpreted and divined a wide variety of documentation. We have found the following documentation components are especially helpful:

- A graphical view of the DTD hierarchy
- Element-by-element documentation including:
  - A description of each element
  - A definition of how the element and its attributes should be applied
  - One or more examples of the element as used in actual documents
  - Discussion of unusual circumstances and how they should be handled (e.g. author name tagging when the given/surname distinction is unclear such as "Leonardo da Vinci")
  - A list of related elements
- Sections devoted to special topics such as special character entities or reference structuring

Based on a review of the documentation provided by suppliers and Inera's experience using publisher-provided documentation to implement SGML creation systems, we recommend comprehensive documentation as an essential part of developing an archive DTD. This documentation will help insure that the DTD is interpreted correctly and consistently to produce high quality SGML files.

[33] Inera has implemented systems to create SGML according to the Elsevier DTD for 6 years. During that time, we have watched the documentation for the DTD grow and mature. The current version, *Tag by Tag*, is very comprehensive. We have used it routinely since it was published to resolve questions of DTD interpretation without the need to send a query to the authors of the DTD. We believe that the combination of this documentation and the Elsevier QCTool have been the foundation for high quality SGML at Elsevier.

10 The Highwire Experience

The experience of Highwire Press at Stanford University over the past six years has many similarities to the issues that will be faced by libraries that maintain archives of electronic journals. For this reason, we present a somewhat extensive review of their activities and experiences.[34]

Highwire Press started to place full text journal content online in May 1995 with the Journal of Biological Chemistry (JBC).[35] Initially, Highwire worked with providers of SGML to develop or modify DTDs that would allow online presentation and full text indexing.[36] For each DTD, Highwire built a custom parser that converted the SGML to HTML for online presentation.

After several years of building custom parsers for SGML to HTML conversion, Highwire developed their own DTD. Ideally, all content is delivered in this DTD, however if a customer already creates content in another DTD, Highwire converts the customer's content to the Highwire Press DTD.[37]

The Highwire DTD was designed with these key goals in mind:

1. Highwire's main goal is to present full text content online. In addition, Highwire provides archive services to their publisher partners.[38]
2. As part of online presentation, Highwire allows sophisticated searches of full text SGML. The Highwire DTD excludes metadata (such as keyword classifications from the Elsevier DTD) that cannot be accessed via the search mechanisms that they provide. Such metadata may be of interest to archiving institutions.
3. Online presentation provides rich linking opportunities unavailable in print publications. Where possible, Highwire creates electronic links that will enrich the research experience of the full-text user.

Highwire derived their DTD from the Elsevier 4.1.0 DTD.[39] The changes made include:

1. Elimination of some Elsevier DTD features that Highwire did not need (e.g. keyword classifications, appendices)
2. Simplification of some features of the DTD (most notably the reference section where references are encoded as formatted text with only the minimal amount of tagging required to create external links)
3. Addition of some new features that Highwire requires for online presentation and linking (e.g. the <addart> element used to create links between articles in a journal via the volume and page number).

All content delivered to Highwire is now converted into the Highwire DTD[40], and Highwire encourages new customers without a DTD to adopt the Highwire DTD from the start.

[34] The information in this section is based on a conference call with Highwire Press representatives Maureen Phayer and Diana Robinson on September 24, 2001. It is supplemented by Inera's experience working with providers of SGML content for online publication by Highwire.

[35] JBC was placed online with the Keton DTD. Today JBC content is still delivered to Highwire in the Keton DTD.

[36] Early DTDs developed or modified for Highwire Press content delivery were the Capital City Press DTD (developed by Inera) and the Keton DTD (developed by Cadmus Journal Services). Both DTDs were derived from the Elsevier 3.0 DTD.

[37] Some publishers do not provide SGML to Highwire. In most cases, the non-SGML files are typeset files from systems such as XyVision, Quark, FrameMaker or Miles 33. Highwire arranges for conversion of these files to SGML.

[38] Although it was not their practice initially, Highwire now archives all original content delivered by publishers. Highwire has also added the original high resolution TIFF and EPS image files to this archive.

[39] At the time Highwire developed their DTD, almost every Highwire partner was using one of the Elsevier DTD versions or the Keton DTD (an Elsevier derivative). This experience with the Elsevier DTD and its derivatives led Highwire to stick with this model rather than chose a significantly different model.

[40] There are a few exceptions, notably content for which custom parsers had been previously built. Highwire decided that the existing systems to handle those DTDs did not need to be updated. At the current time, Highwire converts content into the Highwire DTD from approximately a dozen DTDs including: Blackwell,

This new approach for Highwire has generally been successful, however it has not been without problems. In addition, we believe there are several limitations of the Highwire approach that need to be addressed in the development of a full text DTD for library archiving. The following sections examine some of the issues Highwire has faced.

10.1 The Startup Process

When a journal first submits SGML to Highwire, a formal validation process is conducted. At least two issues of the journal receive careful proofing. Problems are reported to the publisher, and Highwire works directly with the publisher or SGML provider to facilitate changes that may be required in their SGML production processes. The goal of validation is to insure that deliveries will be consistent and correct.

Tables and math receive special attention during the startup process. Tables, which may be submitted in either CALS or Elsevier format, sometimes have problems with column headers and spanning cells. Graphics embedded in tables may cause alignment problems, and Highwire recommends scanning those tables with graphics that require alignment to specific cells. In some situations, publishers submit all tables as scans rather than full text SGML. Highwire discourages this practice because the scan files are large, slow to load in browsers, and cannot be indexed for searching.

Unfortunately HTML has more limited presentation capabilities than CALS or Elsevier SGML. For example, decimal or character alignment is unsupported in HTML. As a result, Highwire sometimes sacrifices a degree of table formatting in online presentation. Their primary goal is to insure the table is readable.

Math is usually tagged according to the Elsevier or MathML DTDs, or encoded as TeX. Because most browsers are unable to render math, the equations are converted to GIF images. Highwire converts all equations from their SGML format to LaTeX, and then converts the LaTeX to GIF images. Sometimes math presents formatting problems, especially in long equations that require line breaks. As a result, some equations receive hand massaging even during the regular production process.

When validation is complete and all problems have been resolved, Highwire moves the journal to a more automated process for regular issues. The importance of this validation process for quality of the final presentation cannot be overstated. Without this step, the overall quality of journals hosted by Highwire would be much lower.

10.2 DTD Updates

The Highwire DTD has undergone regular updates since it was first developed. Typically these updates are driven by the need to add features for the online presentation needs of new customers. For example, one recent upgrade added several elements to support online quizzes for Continuing Medical Education needs. Other updates have added features back into the DTD that were originally removed from the Elsevier DTD.

Updates are usually new features and are typically backward compatible with existing content. While they have not been disruptive for most Highwire processing, Inera believes it is important to minimize the number of updates in an archive DTD in order to maintain stable processes.

10.3 Content Conversion Issues

Highwire faces a number of challenges when converting content from other DTDs to the Highwire DTD. Some of these challenges include:

1. The format of reference citations in the source DTD may be difficult to convert into the Highwire DTD. Ranges of numbered citations (e.g. "[3-5]") can be especially problematic. For more information, see Section 5.1.2.
2. Highwire often finds appendix section headers, table titles, or figure captions are incorrect in SGML. As a result, correct placement and linking of appendix sections, figures, and tables can be difficult.

Elsevier, Lippincott Williams and Wilkins, New England Journal of Medicine, Oxford University Press, OvidBase, and Radiological Society of North America.

3. Many journals now include magazine-style news content in the front section. In many cases, the design and layout of this material is a freeform style that differs significantly from the standard article layout. This content may have uncited or uncaptioned figures and extra design elements that make them difficult to reproduce online.
4. Many publishers do not tag content required for links to external databases. Please see the Section 10.5 for more information.
5. Special issues present some extra challenges, and supplements in particular, because of inconsistent or unwieldy numbering standards. Publishers may restart page numbering for supplementary issues at page 1, creating ambiguous situations that today's reference linking applications cannot resolve.
6. Although the main portion of a journal may be in SGML, some Other Content or Static Content may not be delivered in SGML. For example, tables of contents are not delivered in SGML, and they may sometimes contain necessary information not found in the SGML files (e.g. the caption for a cover image).

10.4 Quality and Consistency Problems

All of the problems discussed up the to this point would exist even in a perfect high quality SGML production system. Unfortunately real world production of SGML has shown that consistency and high quality are not always achieved.[41]

Highwire has found several kinds of quality and consistency challenges:

1. Files are submitted that fail to parse.

   All organizations that create SGML parse the files. However, sometimes they fail to re-parse files after making final edits. In addition, sometimes SGML files become corrupt in transmission, which may cause parse errors. In such cases, the files are returned directly to the provider for appropriate repairs.

   For this reason, Inera recommends that, at a minimum, all SGML files must be re-parsed when they are delivered to an archiving institution. Failure to re-parse all files may result in unusable archives. If files fail to parse, a replacement request must be sent immediately to the supplier.

2. Use of SGML elements may be inconsistent.

   Usually Highwire resolves consistency problems during the startup process. Because some SGML files are created by manual keying rather than software-driven processes, and the people who create them may change over time, the new people may not always apply designated tagging standards.

   As a general rule of thumb, the more people who create SGML according to a DTD, the more variation you will find because each person, in the absence of specific rules (or even in the presence of such rules), will find a unique way to tag the same article.

   Inconsistent tagging can be benign, but in other cases it may effect either presentation or searching of content. Consistency can be enforced with constant manual checking or a custom application can be developed to report consistency errors to the organization that creates the SGML.

3. The content between the tags is incorrect.

   This issue is the most difficult one that Highwire and most other publishers working with SGML face.

   When SGML files are created, the degree of quality assurance conducted varies widely. Some organizations check that the file parses but little more. More sophisticated organizations may have developed manual or automated systems for quality validation (see Section 11).

   These problems arise from several sources:

   a) The content may have been incorrect in print as a result of author, editorial or production errors. The most common errors appear in the reference section, resulting in failed links to databases such as Medline.

[41] This experience is consistent with that reported by NEDLIB, "the number of suppliers able to produce SGML of a sufficiently high quality for reliable and consistent publication remains restricted."

b) The SGML supplier may misinterpret the content, possibly because it is ambiguous. For example, when applying given name and surname tags, it may be unclear how to handle some non-Western names, or names from societies where surnames are not commonly used. For example, the names "Ho Chi Minh" and "Leonardo da Vinci" would likely receive inconsistent tagging.

c) The publishers may be inconsistent in their own production processes. For example, publishers are inconsistent in their use of the dochead, docsubj, and doctopic elements for Non-Article Content. Because the table of contents is built from these elements (few publishers create SGML for their TOCs), these inconsistencies cause errors in the issue online presentation

d) Suppliers may fail to tag certain elements that Highwire feels are essential for online presentation. Typically this information is used to create links, and it is discussed in Section 10.5.

e) In some cases, SGML suppliers create sub-standard work due to internal issues.

Minimization of quality and consistency problems requires startup validation and ongoing quality assurance processes. Because content sent to Highwire is put online almost immediately, most of these problems are caught at some stage of the production process. Furthermore, because Highwire is organizationally focused on quality and customer satisfaction, they have created feedback loops to customers to insure that systemic problems with SGML quality are addressed.

10.5 Linking Problems

One of the most compelling reasons for electronic journals versus print journals is hyperlinks. The ability to rapidly follow hyperlinks through research materials allows new forms of discovery. For this reason, Highwire pays special attention to creation of links when placing journals online.

SGML Highwire has received from customers shows that there are consistent issues with linking:

1. SGML from some suppliers may not provide links between floating objects (e.g. figure, tables) and their citations.

   All DTDs reviewed by Inera support such links, so missing links are likely to occur due to SGML supplier error.

   In some cases, links might be missed because they are in unusual places. For example, if Table 4 was scanned and Figure 5 is cited only from the body of Table 4 there is no place in the SGML file to create a link.[42]

   Missing links make it difficult to correctly place floating objects.

2. Links from citations to references may not be created if the citation was incorrectly formatted.

   A common example is a name-date (Harvard style) citation with a typo in the author's name or the year that was unchecked during copy editing. In such cases, no link is created from the citation to the reference.

3. Linking to full text available at Highwire, Medline, ISI, or other external databases fails.

   This is one of the most common quality problems encountered. Reference link failures are most commonly the result of author error. For example, if the author incorrectly transcribed a first page number or date, the reference will fail to link.

   Mistakes by SGML suppliers also cause linking failures, most often because a supplier did not tag the elements in a reference required to create the link.

4. Links to the Genbank, PDB, or Swisprot databases fail.

   Links to these external databases fail for one of the following reasons:

   a) The database number was untagged because the DTD did not support it.

   b) The DTD supports database links, but the SGML supplier failed to tag it.

[42] Many editors would submit that this is a case of incorrect authoring or editing, however we have seen examples of such cases in SGML instances.

c) The link was tagged in the SGML, but it fails to resolve because of a typographical error. One of the most common errors is using a capital letter 'O' rather than a digit '0', or using a lower case letter 'l' rather than a digit '1'.

Highwire uses regular expression pattern matching to automatically identifying untagged Genbank numbers with a relatively high success rate. PDB and Swisprot numbers are harder to automatically tag.

5. Links to web addresses (HTTP and FTP) and email addresses fail.

   These web links fail for one of the following reasons:

   a) The supplier failed to tag the information in the SGML file.

   b) The information was tagged as a link in the SGML file, but the type of link was unidentified.

   In cases where the link was untagged, regular expression pattern matching works in some cases. This automatic tagging is frustrated by problems such as spaces in the middle of web addresses. Typesetters often insert spaces in web addresses to create better looking text flow in narrow print columns. The spaces are not removed during the process that creates SGML from the typeset file, and the result is a visually pretty but semantically incorrect web address.

6. Links to related articles may fail.

   Every publisher has a unique standard for creating links between related items. For example, an erratum should have a link back to the article to which it is related. These links may fail for the following reasons:

   a) The volume and page information to create a link was untagged because the DTD did not support it.

   b) The publisher failed to tag the appropriate volume and page information in the item.

   c) The publisher uses a system to create the link that is not easily converted to HTML. For example, The Elsevier DTD maintains these relationships in the refers-to attribute of the <art> element. The location of this information in the SGML file does not conveniently create a location in the article for a logical hyperlink to the related content.

Highwire tries to compensate for incorrect or missing link information whenever possible. However they deliberately try to undershoot rather than overshoot when automatically creating links because they believe it's better to miss a link than to create an incorrect link. In some cases, they limit their regular expression pattern matching to specific sections of the document. For example, they only search for untagged email addresses in the front matter of an article.

10.6 Conversion Location

Highwire encourages customers to deliver content in the Highwire DTD, but they cannot require publishers who have their own DTD to deliver in Highwire format. When a DTD conversion is required, Highwire does the conversion, rather than the publisher, for several important reasons:

1. Publishers may not want to switch to the Highwire DTD as their primary DTD because they use their SGML for additional purposes.

2. Most publishers are not interested in bearing the burden or expense of creating SGML in a second DTD.[43]

3. Highwire charges publishers a fee associated with setup of the conversion to the Highwire DTD, however this fee is less than the cost of having the publisher build or buy a system to convert the SGML themselves. It costs Highwire less than other organizations to build this conversion because:

   a) They are familiar with their own DTD

[43] This rationale is consistent with the experiences noted in the NEDLIB report. Most publishers have limited technical staffs, and the cost of compliance with an additional DTD is not sufficiently low to make it more attractive than the cost of Highwire converting the content.

b) They can reuse pieces of other conversions (e.g. regular expression pattern matching to identify linking elements)

c) They know the mission of their conversion intimately, and can create the most efficient conversion to achieve exactly that mission.

4. Highwire has greater control over the quality of the conversion if they do it rather than someone else. Highwire uses a small team to create software-driven conversions. By using a small team under one roof, there is likely to be less variance in the SGML created ("fewer hands, fewer errors"), allowing for a smoother flow of the resulting SGML into Highwire's online presentation systems.

While there are many advantages to converting SGML at Highwire rather than at the publisher, there is one significant disadvantage: if a publisher significantly upgrades their DTD, a full-fledged parser update, coupled with extensive integrity testing, is required at Highwire.

Communication of DTD upgrades to Highwire, whether major or minor, is critical. In many cases, Highwire has only learned of a DTD upgrade through the failure of a file to parse, rather than through proactive communication from a publisher.

10.7 Highwire Summary

Highwire successfully converts SGML from many DTDs to a common DTD, although they have faced significant issues to achieve this goal with high quality. The lessons that archivists can draw from the Highwire experience are:

1. Successful conversions can be completed only if the mission of the conversion and data use is clearly understood before the DTD development and conversion process is started.
2. Some content may be more difficult to convert from one publisher's DTD to an archiving DTD due to how the same structural element may be handled differently in DTDs.
3. The quality and consistency of incoming SGML cannot be insured. However, it can be significantly improved through an initial validation process and ongoing feedback loops.
4. Quality and consistency are maintained on an on-going basis because the content is immediately placed online. Content that is converted to an archive DTD without adequate quality checks may be unusable, in part or in full, when it is accessed at an unknown future date.
5. Certain elements, if they are untagged by the publisher and critical to the mission of the archive, may require tagging with pattern recognition during the DTD transformation.
6. Ongoing production at Highwire requires some manual intervention. It is possible that the degree of manual intervention required for successful use of an SGML archive may exceed the capacity of an archiving institution.
7. Significant publisher DTD upgrades can cause costly rework of the SGML conversion system. Additionally, one cannot rely on publishers to communicate information about DTD upgrades.
8. Highwire controls the conversion from other DTDs to the Highwire DTD. This control probably results in the lowest cost and highest quality conversion

11 SGML Quality Control Tools

The Highwire experience illustrates that high quality results can be maintained through clear standards, continual monitoring, feedback mechanisms, and appropriate levels of investment. However, Highwire is not the only organization to have addressed this problem directly. Elsevier Science provides another valuable case study.

When Elsevier Science first required full-text SGML, they provided a DTD and documentation for the DTD. Even with a 90-page reference manual, Elsevier found that the quality of the SGML they received from suppliers was not as high as they had hoped. Files parsed (most of the time), but high quality SGML required more than parser validation.

Elsevier found that the interpretation of the DTD was inconsistent, sometimes because desired interpretations were undocumented, and sometimes because those people creating SGML had not memorized the documentation. Certain policy decisions that could not be encoded in a DTD needed clarification. Some common author mistakes appeared in SGML because editors did not catch them; sometimes editors made mistakes that affected SGML as well.

When these problems were encountered in 1996, Elsevier was archiving a lot of the SGML - they were not yet placing most of their SGML online. In this regard, the problem Elsevier faced with deferred-use of the SGML was similar to the problems that archivists will face.

Elsevier realized that these errors would cause long-term problems if they were left unchecked and unfixed. In order to catch these and a wide range of other errors, Elsevier started to build a quality control application, known simply as "QCTool".[44]

QCTool validates that a file parses, and then it reports three classes of problems: errors, warnings, and notifications. Errors are problems that indicate misuse of the DTD or violations of Elsevier policies (for example, a non-EMPTY element that has no content). Warnings are lesser issues that probably indicate incorrect SGML (for example, the text "Smith (2001)", if untagged, will cause a warning to be issued that a possible citation needs tagging). Notifications are warnings that might indicate a problem, but are just as likely to be a false warning.

Initially QCTool was used inside Elsevier to examine SGML files for errors. As more errors were caught with the tool, a small team known as "the repair shop" started to fix the most egregious ones. It did not take long to realize that the capacity of this team to fix the errors reported by QCTool would soon be exceeded. In addition, Elsevier decided that the responsibility to fix these errors lay with the supplier, not with Elsevier.

In mid-1997, Elsevier distributed QCTool to suppliers and asked them to run SGML files through it in the hope that suppliers would fix the problems that were reported. In some cases, though, the number of problems reported was so large that suppliers found it more expedient to ignore QCTool. In addition, some errors were the result of author or editorial mistakes, and sometimes it was unclear how they should be fixed.

By mid-1998, most of the early questions about how to use and respond to the QCTool were resolved. But because a large number of errors were still being repaired in the repair shop, Elsevier no longer *requested* that suppliers run QCTool and fix the problems. New policies were instituted that *required* suppliers to run QCTool and fix all errors. Warnings and notifications were to be heeded, but they need not be fixed.

To reinforce the importance of the QCTool, Elsevier announced that any files with errors would cause the entire issue submission to be rejected by Elsevier and returned to the supplier for repair and resubmission (we call this a negative feedback loop). To insure that suppliers adhere to these policies, Elsevier runs QCTool on all files when they are received by Elsevier Science. Elsevier trusts suppliers to run QCTool, but they verify that they have run it.

The lessons that archivists can draw from the Elsevier QCTool experience are:

[44] Elsevier is not the only publisher to create an integrated quality control tool. UCP is another publisher with tools to check content inside elements.

1. SGML that is archived and not put to immediate use online will probably have errors because there is no quality checkpoint.
2. No matter how complete the documentation, strict DTD interpretation can best be insured with software-based quality control tools.
3. Suppliers will not use quality control tools and fix reported problems without negative feedback loops.
4. Publishers must actively monitor and enforce quality standards, even after providing tools to suppliers.

Inera recommends an archive develop quality assurance tools for use by those organizations that create files for archival deposit. We believe the development and deployment of such tools will have a significant impact on the quality of files deposited in the archive.

12 Conclusions

This report, commissioned by the Harvard University Library under a grant from the Mellon Foundation, has examined the feasibility of creating a DTD that can be used to reasonably represent the intellectual content of journal articles published by a range of different publishers.

12.1 DTD Design and Development

Based on the findings in this report Inera believes a DTD or Schema can be developed that will allow successful conversion of significant intellectual content from publisher SGML and XML files into a common format for archival purposes.

We recommend the DTD and accompanying use policies to have the following characteristics:

- The archive should use XML, not SGML, because there is a wider range of tools available for XML.
- The archive DTD should be less restrictive in structure and more streamlined in element selection than the specific DTDs created by individual publishers because it must accommodate a wide range of journal styles.
- The archive DTD should fall between the intersection and the union of structural elements found in the surveyed publisher DTDs. It should be broad enough to capture the key structural elements common to most publishers, however it will exclude many elements that are unique to specific publishers because the intellectual content of these elements can be adequately rendered through less specific markup.
- The archive DTD should make use of public standards rather than proprietary standards wherever possible because it is more likely that work based on public standards will be easier to access at such time as articles are retrieved from the archive.
- Archive XML files should include generated text and face markup. We believe that this solution presents the most effective method for an archive to render content.
- The archive DTD must include comprehensive documentation to insure that the DTD is correctly and consistently interpreted. In addition, the archive should consider providing quality assurance tools to organizations that transform content into the archive DTD.
- Quality control tools should be developed in conjunction with the archive DTD. They should be used to validate all content submitted to the archive. Creators of archival SGML should be encouraged (if not required) to use the tools.

12.2 Transformation Deposit and Retrieval

While we are confident that the design and development of an archive DTD can be successfully completed, we believe there are significant challenges to be faced with its deployment and use. These issues include:

- The quality and consistency of incoming SGML cannot be insured. Some content that is converted to an archive DTD and archived without immediate use or adequate quality checks may prove to be unusable, in part or in full, when it is accessed at an unknown future date.
- Quality can be significantly improved through a validation process and ongoing feedback loops to archive depositors. Setup and maintenance of quality checks will impact the archive budget.
- Some manual intervention may be necessary during SGML transformation. It is possible that under some circumstances the degree of manual intervention required may exceed the capacity of an archiving institution.
- Publishers should not be asked to add granularity when transforming SGML to an archive DTD because it places an increased burden on their ability to deposit content. However in cases when adding structure is critical to enable linking, and it can be done with pattern recognition, this step should be encouraged.

- Tables of contents and indexes must be created by the archive upon acceptance of the content because they are an integral part of the search and retrieval mechanism. We expect the capabilities for such systems will improve over time, allowing the archive to improve access to content. The degree of sophistication of these systems will have a significant impact on the successful use of the archive, however they will require significant investment by the archiving institution(s).

12.3 Final Thoughts

HTML utilizes a relatively simple DTD to present many kinds of content. HTML however, is too simplistic for archival storage. When SGML is transformed to an HTML manifestation, semantic information is lost that is critical for link creation and therefore research discovery.

Publishers have invested significant resources building systems that convert SGML to HTML. Transformation of SGML to a streamlined E-Journal DTD that is more complex than HTML may require investments of at least a similar magnitude.

Some publishers have sufficient budget and technical resources to setup high quality conversion systems that can be used to convert their SGML to an archive DTD. Smaller publishers may lack budget or technical resources to build these systems, requiring assistance or even conversion services from an archiving institution.

The technical problem of DTD design and development can be solved, however we believe the greatest challenges will arise from implementation of the systems for transformation, deposit and retrieval of XML files in a cost effective and high quality manner.

13 Acknowledgements

We would like to thank the following people who participated in this study, provided materials, responded to our queries, and offered invaluable feedback on draft versions of this report:

Bob Hollowell	American Institute of Physics
Kristine Schnebly	BioOne
David Sommer, Richard O'Beirne	Blackwell Science
Karen Hunter, Jos Migchielsen	Elsevier Science
John Sack, Maureen Phayer, Diana Robinson	Highwire Press
Stephen Cohen, Ken Rawson	IEEE
Y Kathy Kwan, Ed Sequeira	Pubmed Central
Howard Ratner, Heather Rankin	Nature Publishing Group
Evan Owens, John Muenning	University of Chicago Press
Margaret Wallace	John Wiley & Sons

We extend our thanks to Dale Flecker, MacKenzie Smith, Stephen Abrams and especially Marilyn Geller, at Harvard University for many stimulating conversations about the mission of an E-Journal archive, and for their support and guidance. We are grateful to Maureen Phayer, and Diana Robinson at Highwire Press for taking significant time to discuss Highwire's experiences, to G. Paul Bozuwa at Capital City Press for permission to reference Inera's experience implementing their SGML systems for scholarly publishing, and to the dedicated team at Inera. Without the help of all these people, this report could not have been written.

Bruce Rosenblum & Irina Golfman
Inera Incorporated
Newton, MA
December 5, 2001

14 Bibliography

Bide, Mark. Standards for Electronic Publishing. 2000. http://www.kb.nl/coop/nedlib/results/e-publishingstandards.pdf. Accessed on August 28, 2001.

Blackwell Publishing. Blackwell Publishing DTD 4 documentation. 2001. http://www.blackwellpublishing.com/xml. Accessed on September 18, 2001.

DeRose, Steve. The SGML FAQ Book. 1997. Kluwer Academic Publishers: Norwell, MA.

Diaz, Andrew, *et al*. W3C's Math Home Page. 2001. http://www.w3c.org/Math/. Accessed on October 31, 2001.

Goldfarb, Charles F. The SGML Handbook. 1990. Oxford University Press: New York.

Kennedy, Dianne. ISO 12083 Survey. 1998. http://www.xmlxperts.com/12083.htm and http://www.xmlxperts.com/survey98.htm. Accessed on October 4, 2001.

Megginson, David. Structuring XML Documents. 1998. Prentice Hall: Upper Saddle River, NJ.

Owens, Evan. SGML and The Astrophysical Journal: A Case Study in Scholarly and Scientific Publishing. 1996. http://www.journals.uchicago.edu/sgml96.html. Accessed on October 11, 2001.

Pepping, Simon and Schrauwen, Rob. *Tag by Tag*. 2001 Elsevier Science: Amsterdam.

St. Pierre, Margaret and LaPlant, William P., Jr. Issues in Crosswalking Content Metadata Standards. 1998. http://www.niso.org/crsswalk.html. Accessed on June 23, 2001.

Poppelier, Nico, *et al*. The STIX Project Home Page. 2001. http://www.ams.org/STIX/. Accessed on October 30, 2001.

van Herwijnen, Eric. ISO 12083 Math DTD. 1993. http://www.ams.org/html-math/iso12083.html. Accessed on December 3, 2001.