Digital Publishing - Data Handling

Ross MacIntyre
Manchester Computing, University of Manchester, Manchester, UK
email: r.macintyre@mcc.ac.uk

Abstract

SuperJournal is a research project aiming to identify what factors will make electronic journals successful and why. The project brings together some 20 society, university press and commercial publishers, who are providing electronic versions of around 50 journal titles in discrete areas of academic research. The "why?" is being established via a formal evaluation study being conducted by Loughborough University. In order to undertake the study, an application is being developed by the University of Manchester which offers choice over both content and functionality to the users, who are based at targeted universities.

This report focuses on the technical development work undertaken so far to support the scalable distribution of digital journals in an academic network environment.

University of Manchester receives data files from the different publishers in a variety of formats: principally SGML, HTML, PDF and PostScript, plus GIF, TIFF and EPS; and via various transfer mechanisms. All SGML is 'synthesised' to conform to a DTD defined by the project, converted to HTML including enhancement. The associated scalability issues are discussed. Data is loaded into a number of datastores: an objectbase, a relational database and application-specific databases; and indexed via a number of search engines. The application is accessed over the WWW and consists of an integrated assembly of software, providing the core functionality to maximise use, e.g. searching, browsing, linking, screen presentation and personal preference setting.

1. Project Background

SuperJournal [1] is a project in the UK's Electronic Libraries Programme (eLib) [2] researching the factors that will make electronic journals successful and of real value to the academic community. The objective of the research is to answer the question: "What do readers and authors really want from electronic journals?" and to explore the implications for other participants in the publishing process. SuperJournal is a collaborative research project involving publishers, universities, their libraries, and academic researchers.

"What do you want from electronic journals?" is an easy question to ask, but difficult for readers and authors to answer unless they have hands-on experience using electronic journals to provide a context for their views and opinions. The project has therefore adopted the following method:

- To deliver clusters of journals to readers in an electronic application that can be changed over time
- To record usage with a view to identifying critical success factors and barriers
- To explore with readers and authors what they really want, in light of their experience using the journals

Some 20 participating publishers contribute the content of established refereed journals to form the journal clusters in different subject areas:

- Communication and cultural studies
- Molecular genetics and proteins
- Political science
- Polymer physics.

The journal content is made available to users at eight university sites in an easy to use application available from the WWW. Different search engines, user interfaces, viewers, and multimedia tools will be combined to give readers choices in a testbed environment that will enable us to learn their preferences. During the course of the project, each journal cluster is launched in turn, the electronic journal application is upgraded, and new features are added. Researchers at Loughborough University are conducting a formal evaluation to find out the features users do/don't like and why.

2. Role of Manchester Computing

Manchester Computing at University of Manchester is the project partner responsible for the technical development work, including the following elements:

- **The electronic journal application:** To develop an application environment with features and functionality that can be tested at user sites, and to develop the programs to record usage.
- **Network host environment:** To make the electronic journal application available to user sites via the UK academic network, SuperJANET
- **Data handling processes:** To convert the electronic files, supplied by the publishers, into the electronic journal application

- **Datastores:** To develop an environment to store the content, multimedia elements, and manage the application.

The current technical environment uses a single, central server, a Sun CS6400 running Solaris, housed at Manchester Computing. A move to multiple/distributed server architecture is planned during the project, with additional server based at publisher or other third-party sites. Access is via the main WebServer (Apache). On the Client-side, the application assumes a graphics-capable browser is being used (typically Netscape or Microsoft Internet Explorer) in a Windows, Macintosh or X-terminal environment. While vendor specific extensions have been avoided wherever possible, where they are exploited, they are clearly identified within the application. The same holds true where a certain level of functionality is required within the browser, e.g. Frames support. Helper applications and plug-ins will need to be included, by the end-user, to explore multimedia elements etc.

The SuperJournal project is funded for three years, from December 1995 to November 1998. By February 1996 the technical team was assembled:

- Technical Project Manager, who recommends technical direction
- Data Handler, responsible for developing all conversion code
- Software Developer, who has built the application.

In November 1996, the first journal cluster, Communications and Cultural Studies (CCS), was launched at three of the eight participating university sites. The remaining five universities had access by February 1997. The second cluster, Molecular Genetics and Proteins (MGP), was launched in May 1997 to all eight sites.

This paper describes the data handling processes that have been developed and includes some recommendations based on our experience. A key consideration in designing the data handling processes was that they should be scalable. In a project involving 50 journals from 20 publishers, scalability is important so that large quantities of data can be processed efficiently, with as little manual intervention as possible.

3. Data Analysis

Before designing the data handling processes, we needed to answer some basic questions about the data itself:

- What file formats can the publishers provide, both now and over the life of the project?
- Of these, what file formats should we accept?
- What is the best method for transmitting the files from publisher to MC?
- Once we have the files, what data should we store? In particular, what metadata should we store, sorting the production process data from the real article metadata.
- How can we create a consistent data set? What is that data set, and how can we derive/assemble it.

We therefore embarked on a data analysis exercise to understand the files that could be supplied, so that scalable processes could be designed around them.

3.1 Publisher Data Formats

The publishers' role was content creation, and MC's role was to deliver an application with maximum functionality, within the constraints of the content they could supply.

Each of the publishers participating in SuperJournal publishes journals in printed form, and the electronic files they could supply were generally a by-product of the print production process. The particular files available from each publisher depended on the nature of their production process, the suppliers they used, and whether they were already developing electronic journal products to accompany print.

The eLib Programme offered some general recommendations on file formats in their eLib Standards Guidelines [3], but these could not be forced on publishers. We needed to find out what files they could supply, evaluate the processing requirements, and assess the functionality that could be delivered in the SuperJournal application by using them. Then we could decide on submission formats.

In February 1996 a questionnaire was sent to each publisher to find out the files each could submit. Samples were sent to MC for testing and evaluation. Once the DTD analysis outlined in Section 3.2 was completed, we agreed on the following submission formats:

- **Standard Generalised Markup Language (SGML)**

From the start, virtually all Publishers were working with SGML. Most could supply "header" information in SGML format, where the header contains information about the article, eg author, title, volume, issue, date, abstract. The situation has moved on considerably, but in general the DTDs in use were either 'bespoke' or variants of Majour, Elsevier, and latterly SSSH [4]. Moves towards ISO 12083 standard DTDs are underway.

A few publishers could supply one or more journals with the full articles in SGML format, and several indicated plans to do so. In the case of full article SGML, each publisher developed their own DTD or was having one created. The use of SGML is explored further in the next section.

- **Portable Document Format (PDF)**

Most of the publishers could supply the journal articles in PDF format, typically generated by their print suppliers from PostScript files. The article can be viewed with the Adobe Acrobat Reader and looks exactly like the printed version it is derived from. It is cheap to produce and protects the 'look' of their journals, something they have invested serious money in designing.

- **HTML**

A few publishers could supply the articles of individual journals as HTML, typically generated when the publisher offers a Web version of the journal.

- **Graphics**

Encapsulated PostScript (EPS) & Tagged Image File Format (TIFF) are used widely in publishing, and most publishers can provide them. However, SuperJournal is a Web application, so we need graphic files that can be used with Web browsers. As more publishers develop Web applications, we now find they can supply graphics in Graphic Interchange Format (GIF), often multiple files of different sizes and quality. Joint Photographic Experts Group (JPEG) format files are few and far between.

- **Sound/Video**

Virtually none of the publishers generate sound, video, or other multimedia files for their journals as a matter of course. A notable exception is a music journal which includes a yearly CD-ROM which could be offered over the Web. This area is changing certainly, and we want to encourage experimentation. See Section 6 for details of our experience with one publisher's multimedia data.

- **Other Files Evaluated**

We also received files for testing in PostScript, tagged ASCII, and TeX/LaTeX format. In cases where the publisher can only supply articles as PostScript or TeX/LaTeX, this is converted to PDF. Where TeX/LaTeX is used for equations and formulae within an article, this can be accommodated by creating GIFs and displaying them in-line.

- **Comments on Submission Formats**

In May 1996 the project decided to accept SGML headers and PDF articles for the first journal cluster. It should be stressed that this was not a decision advocating one format over another. A 'preferred format' is a possible outcome of the project, but the preference would have to be demonstrated after providing users with different choice.

The initial decision to accept SGML headers and PDF articles was largely based on what most publishers could provide. One year later, in May 1997, the second journal cluster includes several journals with articles submitted full text SGML. Over the life of the project we envisage accepting new file formats as the publishers develop the capabilities to supply them.

3.2 DTD Analysis

To assist planning, we sought an expert view on the use being made of SGML by our contributing publishers: was the use of a project standard DTD a forlorn hope? Alden Electronic Products [5] performed a formal DTD analysis on the six DTDs then in use by the publishers (one applied to the header data only).

They concluded that:

- **Headers:** The DTDs varied considerably, but they contained basically the same data elements. It should be feasible to develop a single DTD to accommodate the variations.
- **Full text:** The DTDs were similar in many respects and could be harmonised, but only up to a point. Where the DTDs differ, the relative importance of each difference must be determined. It would be possible to totally harmonise the DTDs, but the consequence would be loss of document structure.

So the DTDs were compatible at a high level, but not at the detailed level. This did mean that a generic header DTD was possible, which was good news. However, it would be naive to expect the publishers to adopt a DTD of our choice. We would have to accommodate the mark-up that each individual publisher used, and design data handling methods to combine the data and process it.

3.3 SGML Data Mapping

Data mapping is a familiar activity in any data migration project. However, for this project it was unusual in that the source data descriptions were defined, but the target was not. One form of documentation which was used was a spreadsheet, indicating the mapping of DTD elements to each other. The target DTD evolved from this process. The rules for derivation were also noted, these being the starting point for the conversion code.

We started the activity with the six DTDs covered in Alden's analysis, but extended this to include SSSH, as this was recent, authoritative, and a move to developing a standard DTD for publishers. The same process was repeated to establish a target full-text DTD. The mapping has now been conducted for 13 different DTDs, including Dublin Core [6].

Of the 170 different tagged items from the header DTDs, we carried 70 forward into SuperJournal. When expanded to full-text, 100 were carried forward from around 300 tagged items.

4. Data Conversion

The objective of our data conversion is to process the data into a consistent format for indexing and loading into the SuperJournal application, not simply to generate HTML versions of the data for display on the Web.

4.1 SGML Conversion

Although we had accepted that we would have to deal with multiple input types, we sought to minimise any conversion code duplication. This lead us to treat header and full-text data in compatible but slightly different ways. The approach adopted is outlined below.

An SGML DTD was an appropriate language to describe what, in data modelling terms, could be called the synthetic data architecture: a synthesis of real world data, but remaining artificial. A bonus is that parsing offers syntax QA for free. The principle of treating conversion in a staged fashion is well established, see Alschuler and the Rainbow DTD [7]. The DTDs can be viewed as 'halfway houses'.

Microstar's Near & Far [8] was used to document the DTD development, albeit retrospectively. It was not used during the DTD creation, but has been for documentation and subsequent maintenance.

The project did not explore the use of DSSSL [9], which became ISO/IEC 10179 in April 1996, and may be well suited to this kind of activity. This may be examined in retrospect, together with JADE, its associated (public domain) engine.

4.1.1 SGML Header Data

First we developed a "generic" header DTD for SuperJournal which contained elements from all the various publisher DTDs. This is the input file definition for the conversion process, and all incoming SGML header files are parsed against it using the "sgmls" parser [10].

Next we developed a "SuperJournal header DTD" (SJ-DTD) which defines the data elements that are extracted for the electronic journal application. This can be viewed as the conversion output file definition, and the data definition is the basis for the objectbase and database design. The conversion process results in a subset of the header data provided by the publisher, which is then parsed, and produces files for input to the subsequent data load processes.

An advantage of this approach is that it is easy to add header data from a new source. The "generic DTD" is extended to accept input from another DTD, e.g. SSSH or Dublin Core. Having one SuperJournal header DTD means that only one filter needs to be written to produce output. So, to produce output for Dublin Core, no new filters need to be written.

It should be noted that the header data is not converted/stored as HTML.

The above processes are shown in the Data Conversion Schematic below.

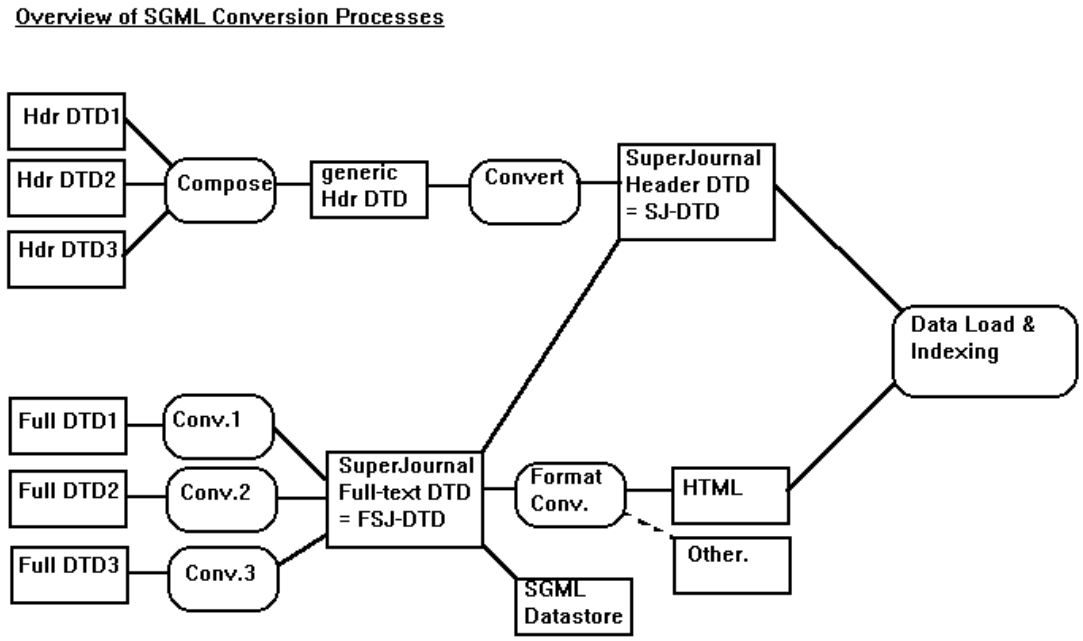
4.1.2 SGML Full-Text Data

It was not feasible to define a "generic DTD" for full-text, i.e. it wasn't possible to accommodate the differences in data description, given the current SGML mark-up

used by the publishers. A 'target' full-text SuperJournal DTD (FSJ-DTD) has been defined, but conversion requires more in the way of pre-processing and re-formatting. Therefore, for each DTD, separate conversion modules have been created to convert the data to conform to FSJ-DTD, which in then used to convert to HTML, though other formats may be included, e.g. RTF or XML [11].

The use of FSJ-DTD also means data can be supplied in a consistent format to SGML-capable datastores, indexers, etc.

The header elements of FSJ-DTD are consistent with SJ-DTD, supporting the processing of header data for articles submitted in full-text SGML format.



4.2 SGML Conversion Code

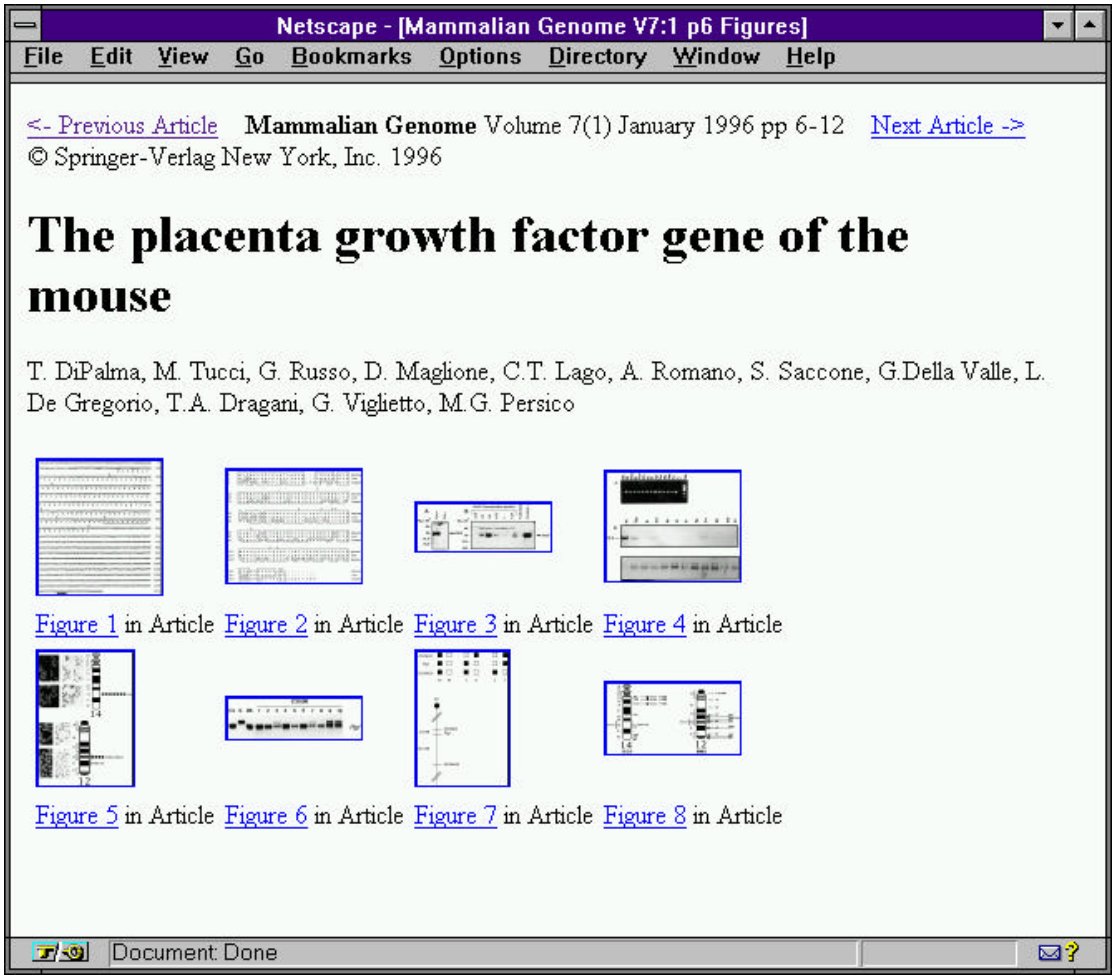
A number of options were considered for the conversion coding and OmniMark [12], an SGML aware 4th generation hypertext programming language, was selected. However, due to time constraints, the header conversion coding actually commenced with a UNIX utility (lex yacc) and C++. The conversion code runs to approximately 5,000 lines and the same for the DTD definitions. It currently copes with nine DTDs, including SJ-DTD. SJ-DTD was used by two publishers as a stop-gap measure for header mark-up while their own DTDs were being written.

Full text conversion is performed using OmniMark scripts, which consist of approximately 200 lines of code per DTD. The parsing of full-text SGML is done using the parser included with OmniMark. We have found the error messages more "developer friendly" than from sgmls.

As mentioned earlier, no header information is actually stored as HTML, it is converted for display 'on-the-fly'. Only full articles are converted and stored in HTML. The articles produced include:

- **Internal links**, e.g. from reference to bibliography, and from text to footnotes, figures, equations, and tables.
- **External links**, currently to MEDLINE [13], though others are under development. Where the references are tagged, the required data is extracted and sent via email to MEDLINE, who return, for matched citations, the unique identifier, which provides subsequent access. This link is then tagged as a MEDLINE reference within the FSJ-DTD and is converted to an embedded URL within the HTML file.
- **Illustrations**, converted to an appropriate format for display. Small GIF thumbnails are the default display within HTML, linked to the full-sized GIF image, which is displayed in a separate window. Variations on this will be explored during the project.
- **Hyperlinks**. Where URLs are either explicitly tagged, or identifiable by string-matching, they are tagged in FSJ-DTD as a URL, allowing a hyperlink to be created when converted to HTML.

We have also created what might be termed "graphical contents pages". We pick out "like objects", e.g. figures, which can be viewed separate from the article, allowing the user to scan quickly through. Images are expanded, or display the reference point within the article. A sample graphical contents page is shown below:



4.3 Other Conversion

- **HTML data**

Any HTML data files received are processed to conform to requirements of the application and enhanced where additional functionality can be added in a scaleable fashion as detailed above.

- **PDF data**

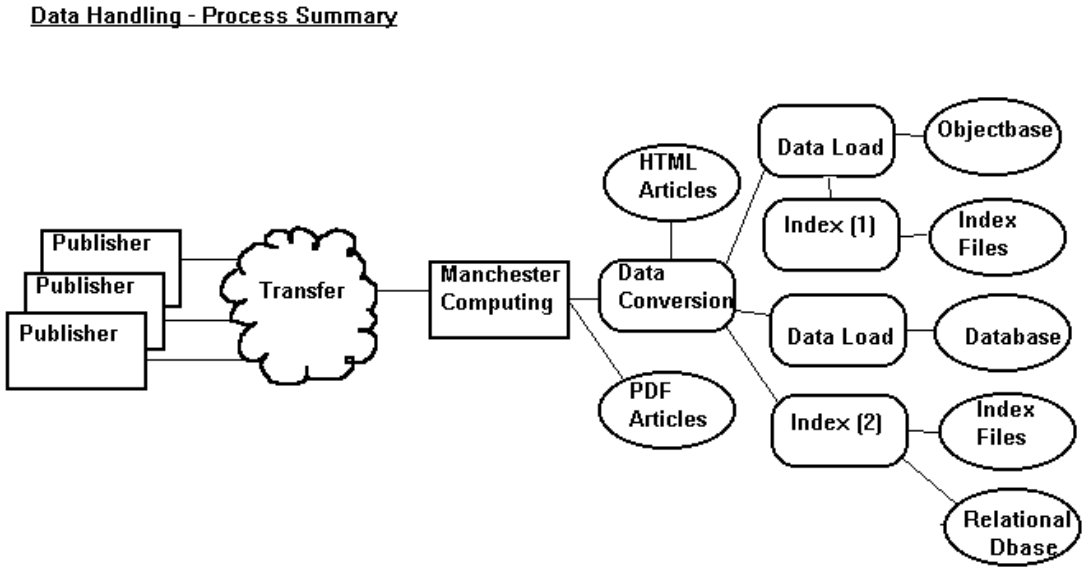
The PDF files are referenced as external data files, so are essentially ‘untouched’. There are exceptions, e.g. entering a base URL to enable weblinks.

- **Multimedia data**

The handling of multimedia data could not be considered scaleable, and no repeatable processing requirements have been established at this stage. Our experience with one publisher’s data is detailed in Section 6.

5. Data Handling Process Summary

The conversion processes described in detail above provide the input for the data load process. The schematic below provides an overview of the handling processes.



- The electronic source files are transferred from the publisher (or their typesetter) to MC. A part-time Production Co-ordinator schedules the transfers. The method of transfer is usually FTP, though variants have included: CD-ROM, 3.5" diskettes, SyQuest cartridges, and optical discs.
- On receipt, MC archives the file(s) and catalogues the data files, renaming where necessary.
- SGML is parsed against the relevant DTD, converted, parsed again (against SJ-DTD or FSJ-DTD) and formatted into load files. HTML and PDF files are sampled, but receive minimal processing.
- The data is then passed to the load and indexing procedures specific to the data storage software. These are not detailed in this report.

6. Multimedia Handling: The Sage Experience

Sage Publications publishes journals and books in the social sciences and humanities and is an active participant in the project. They have been keen to explore the potential for electronic versions of their paper journals, and have implemented comprehensive internal linking and weblinking in their PDF articles. In looking for supplementary, 'non-printable' material to attach, Sage felt it was important not to disadvantage the paper version. They felt multimedia should therefore be peripheral or amplificatory, not core. Sage targeted their media and communication studies journals looking for where:

- There was a link between the article content and outside agencies

- In the process of compiling the material for an article, audiovisual material had been produced or assessed
- There was a visual element in the content that could be enhanced by providing an image; still or moving.

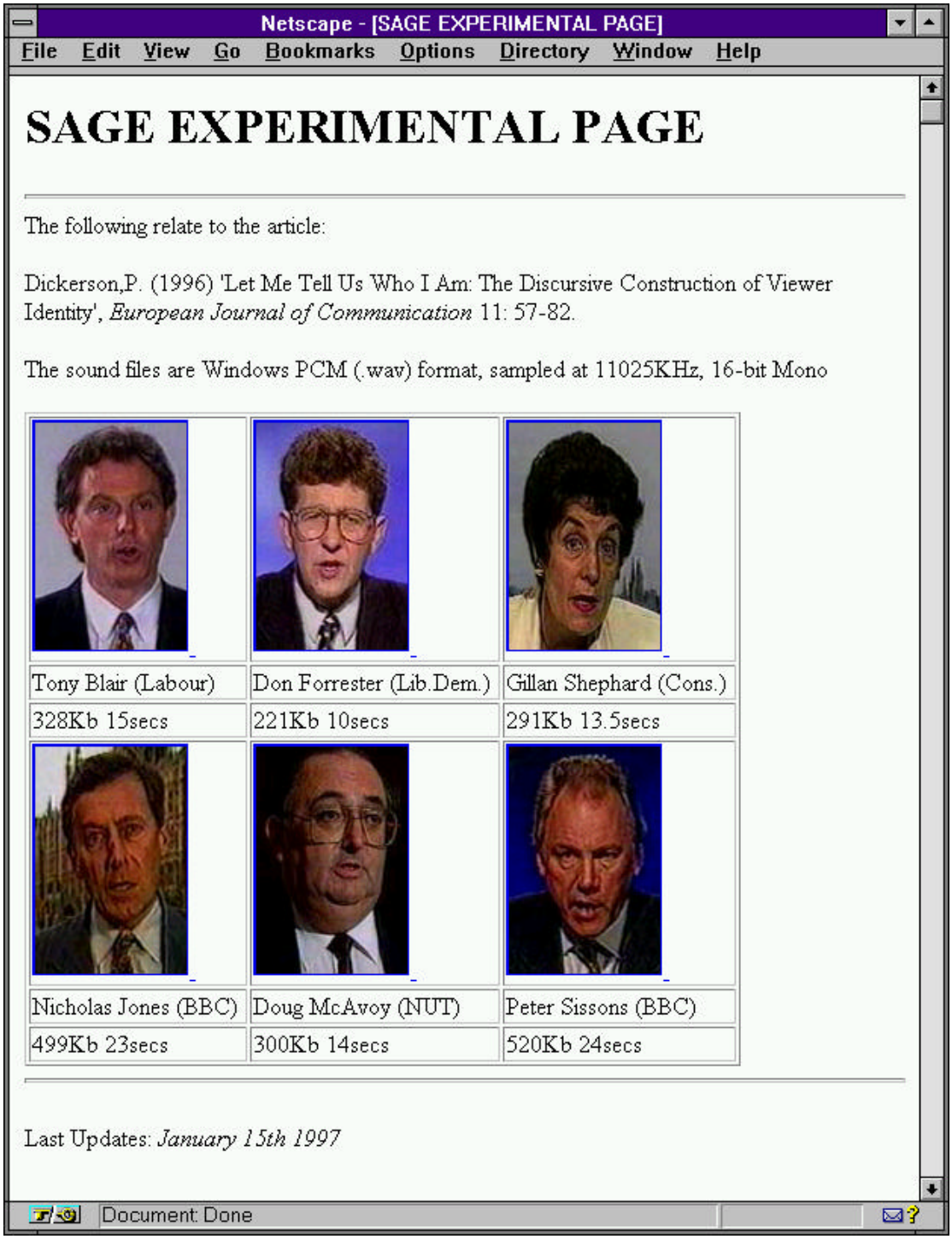
They were successful in identifying an instance of content enhancement. An article by Dickerson [14] had been published examining the presentational style of television presenters, interviewers and interviewees.

In preparing the article, BBC news footage had been recorded onto VHS video. Sage provided a list of the 6 items selected from the video and for each, a description, duration and supplied their video counter reading in minutes and seconds as from/to times. Initially, it was planned that the extracts would be converted into separate video files, so as to show what the research interviewees had seen on the TV. Manchester has a Graphics Unit that is a national centre for MPEG compression, including the ability to take input from a VHS player; though without (at that specific time) accurate time sequencing and detailed sound information.

Six 'rough' files were produced, containing 25 frames/second and stereo sound. The file sizes ranged from 1.8 Megabytes (10 seconds) to 4.4 Megabytes (25 seconds), i.e. between 160 and 180 Kilobytes per second of duration. The files were large, but that is not necessarily a problem. It does, however, raise the question "Was it worth the wait downloading?". Sage felt the answer was "No", having seen the footage played back using a software-only decoder, with the sound and images not in synch.

A fundamental question then arose: "What was the most important element of this additional material?". It was what was said, both at source and by the news staff. The movement was almost non-existent, often referred to as 'talking heads', but the images did help identify the person speaking. So, sound files were created from the MPEG files, using WAV format, at various levels of quality. WAV was used because that platform was available and the format is widely used. WAV format is produced by sampling analogue sound, so the frequency of sampling determines the quality and file size. The lowest level (11KHz, mono) was felt acceptable for the human speech. The resultant file sizes were 200-500 Kilobytes.

A web page was created containing six images sampled from the MPEG files (equal size, equally deadpan) linked to the six sound files. This was linked to the article concerned (in PDF format). All six were put on one page so the user would not have to go back and forth to hear all the samples.



We also tried MPEG to AVI conversion, which made the 4.5Mb file 10Mb. Quicktime may have been preferable, as it is an ISO standard and a mature technology with readily available (freeware/shareware) viewers. However, as sound seemed adequate, this was not taken further.

The experience suggests the following:

- Format is instance specific. Sound was the key attribute here, but in another case movement may be key, or the quality of the sound may be key.

- Why try and choose just one format? There is a difference between the requirements of material intended for immediate consumption and that for prolonged study. Why not choose two formats, a representative sample and the 'real McCoy'? The sample might be considered a multimedia thumbnail, smaller in size, lower in quality, and lower in expectation of end user's resources (eg hardware, software, technical prowess). The real McCoy would be larger, higher quality, and require greater resources from the user.

For audiovisual material, "streaming" is also an option, where the playback starts as the file is downloading, rather than waiting. Though, as this requires sequential viewing during streaming, a sample may still be appropriate, if the opening sequences are not fully representative of the whole.

7. Recommendations on Data Handling: A "Cut Out & Keep" Section

The following recommendations and observations are made based on our experience processing data from the publishers using the data handling methods described above. All should be found in the realm of common sense and are consistent with the concept of "self-identifying data", well established in data processing.

A key objective in designing the SuperJournal data handling processes was to ensure that they were scaleable, to allow large volumes of data to be loaded into the application. They should also be as automatic as possible, minimising the need for manual intervention. Most of the issues raised below focus on manual intervention. In some cases we've been able to write scripts to overcome problems, but this means it is necessary to pre-process each publisher's data with a different script before running it through the main data conversion process.

It should be remembered that the files we received were not necessarily produced for creating electronic journals. In general they were produced as a by-product of the production process for getting the journal into print.

Our top four recommendations are as follows:

- Filenames should be unique within a journal issue
- Use appropriate file extensions to indicate the file type
- Graphic or multimedia files should indicate the article to which they belong.
- Ideally filenames should reflect the sequence of articles within the journal issue

7.1 Transfer Method

The preferred method of delivery would be CD-ROM. (This is consistent with the experience of TULIP project [15].) CD-ROM is resilient, does not get corrupted in transit and provides a physical archive of the original data received. However, in practice, FTP is more readily accessible at present. The process is easiest if publishers FTP data to SuperJournal.

7.2 Matching Files to Articles

When publishers send more than one issue at a time, ideally a directory structure should be included, so it is easy to see what files belong with each issue and article. Without this, it is necessary to manually inspect the content of the SGML header files to match files to articles.

For a given journal issue, we would expect to receive the same number of SGML headers and full articles (either as PDF or SGML full text). Where discrepancies occur these have to be investigated. Where there are more headers than PDF files, typically one PDF file contains several smaller ones. Where there are more PDF files than headers, we may have been sent unneeded PDF files, or some headers may be missing. We cannot load articles into SuperJournal without a header, so in some instances we have manually created them.

7.3 File Naming

Where possible we use the filenames assigned by the publisher, but in some cases it has been necessary to rename files. We require files to order under UNIX in page number order within an issue so that they are loaded into the application in the correct sequence. Where this is not the case, scripts have been written to rename files using volume number and start page number.

Some publishers use filenames longer than eight characters. Where filenames are contracted to eight characters for transmission, significant characters may be lost, resulting in duplicate filenames.

7.4 PDF File Problems

- **Damaged PDF Files**

Always ensure that PDF files are FTP'd in binary mode. Some PDF files appear to have been damaged during FTP transfer, and we think this is caused by ASCII FTP transfer. Do not rely on "automatic", as this can result in the ASCII transfer mode being selected.

- **Encrypted PDF Files**

Some publishers' PDF files have security settings, to prevent files being changed, and security settings cause the PDF to be in encrypted format. One of the search engines used for the SuperJournal application is unable to index encrypted files. It is possible to remove security settings (provided we are supplied with any password) using Acrobat Exchange, but this is a manual operation. (It would also be an option to join the Acrobat Developers Association and develop batch programs in-house.)

- **Base URLs and Multimedia Links**

To include multimedia links in a PDF file to be displayed over the web, requires the link to be created as a weblink. We intend to use relative file names for these links. But

for display with some versions or set-ups of Acrobat, it is necessary to set the Base URL in the PDF document. We also need to set this Base URL so that we can log when users access this multimedia link. Setting the Base URL is a manual operation.

7.5 SGML Parsing Problems

SGML requires strict adherence to its syntax, and errors show up when parsing the SGML file against the DTD. The following problems have all been observed and require correction for parsing to be successful. In some cases a script can be used, but each still requires manual intervention:

- Stray punctuation (often between author names, and within affiliations)
- Identifier and identifier reference attributes which don't match, or "null" identifiers
- Newlines in inappropriate places (eg in the middle of an SGML tag), or no newlines at all
- Non-ASCII characters (SGML character entities should be used)
- Missing lines of text (in full text articles)

7.6 SGML Header Elements

- **Minimum Data Requirement**

Our minimum requirement for data elements supplied in the headers is:

- Journal Title or Identifier
- Volume and Issue numbers
- Cover Date
- Article Title
- Page Numbers

Although this information may be obvious to a person, it is not obvious to an automatic script and we cannot load an article which has no title. We also prefer the following to be tagged, but it is possible to generate them automatically:

- Publisher name
- ISSN
- Copyright statement

Some publishers do not supply the cover date as a data element, or they supply only the year. In these cases, we supply the cover date as an argument to a pre-processing script. There are many variations of format for cover dates and copyright statements. We have had to include data conversion coding to accommodate them all and to provide some degree of consistency of their display in the SuperJournal application.

- **Author Names**

To index author names within the application, they need to be tagged to indicate each author's surname. We have seen SGML which lists all the authors in one field, with no

obvious way of splitting them automatically. We manually edit these author fields. Sometimes author affiliations are run together, but it is usually possible to deduce where to split them.

- **Capitalisation**

The author names and article title in some headers are entirely in lower case letters. These are changed to upper case letters by the data conversion process.

- **Duplicate Headings**

In some cases publishers have included a heading within the data element itself, e.g. “Keywords” or “Abstract”. As the SuperJournal application generates such headings automatically based on the data type, we have to remove these unnecessary headings as part of the data conversion process, so they will not be duplicated.

7.7 SGML Full Article Elements

- **Internal References**

If internal references are tagged, e.g. from a figure reference to the figure, or from a citation to the bibliography, hypertext links can be generated in the HTML making the article more user-friendly. Some publishers are not supplying this tagging. It is difficult, particularly with the citations to generate them with a global edit.

- **Bibliographic References**

There are variations in the way bibliographic references are tagged. In some cases each bibliographic reference is just a text string with no internal tagging. In other cases there is a fine-grain tagging from which details such as title, author names, and journal can be ascertained. Where bibliographic references are tagged in this way, we can build external links, e.g. to MEDLINE abstracts.

- **Figure and Table Positions**

In many cases, there is no indication in the SGML where a figure should appear. Some publishers put all the figures in a “bucket” at the end. Some put the figure at its first reference, which could be in the middle of a sentence. If figure positions are not defined, we display the figure at the end of the paragraph in which it is first referenced.

- **Tables**

There are variations in the way tables are defined. Some are supplied as graphics, so their display is the same as for figures. Some tables are tagged in SGML, which necessitates their conversion to HTML. This conversion is non-trivial.

We have come across some strange tagging of table footnotes and legends, e.g. tagged within the last row of the table, in a cell which spanned the entire table. The data conversion has to extract these and tag them correctly. Where the same footnote

identifiers (e.g. “a”) are used in each table, this causes parsing errors if there is more than one table in an article. To overcome this we prepend the identifier with the table number.

- **Special Character Entities**

Within SGML it is possible to include a large number of special character entities. At present HTML browsers display very few of these correctly. Where possible special character entities are converted into keyboard characters. In other cases, particularly Greek letters, the word is displayed surrounded by square brackets. In the future we shall look into better ways of displaying these characters.

7.8 Graphics files

- **Thumbnail and Full Size GIFs**

We request a thumbnail size GIF file to display within an article, and also a larger size GIF file to display when the user clicks on the thumbnail. So far, publishers have supplied these files. We use ImageMagick [16] to convert between graphics formats and sizes. ImageMagick has a command line interface so we set up batch conversions where necessary.

- **Tagging of Graphics Files**

We have to rename graphics files and their paths from those used by the publishers for consistency within SuperJournal. Problems have been encountered where links to graphics files are missing or wrongly tagged. There have also been cases where figure files were missing.

7.9 Conclusion

This section may seem rather negative, because it highlights the problems encountered when setting up the data conversion and load process. In fact, we have been successful in creating a data handling process which is to a large extent scalable. Most problems are encounter only once, because code is written to overcome them. The cases where manual editing of the supplied data files is necessary are in the minority.

8. Future

In the area of data handling, SuperJournal will (continue to) devote resources to:

- additional SGML DTD conversions for new journal titles
- adding additional features via HTML, MathML [17], XML,...
- converting to formats other than HTML
- adding metadata to HTML, including mapping to Dublin Core
- providing full-text search facility to all articles, including PDF

- enhancing PDF articles received
- offering more choice of data formats

As far as the publishers are concerned, it is highly likely that, as all will have moved on in terms of their production capability with regard to electronic journals, many of the scalability problems discussed will be addressed and removed. Time will tell.

Acknowledgements

SuperJournal is a project funded by the Joint Information Systems Committee (JISC) of the Higher Education Funding Councils, as part of its Electronic Libraries Programme (eLib).

References

1. SuperJournal Project - http://www.superjournal.ac.uk/sj

2. Electronic Libraries Programme, eLib - http://ukoln.bath.ac.uk/elib/

3. eLib Standards Guidelines - http://ukoln.bath.ac.uk/elib/wk_papers/stand.html

4. SSSH - http://cobham.pira.co.uk/sssh/

5. Alden Group Ltd - http://www.alden.co.uk/

6. Dublin Core - http://purl.org/metadata/dublin_core

7. Alschuler, Liora. *ABCD....SGML. A User's Guide to Structured Information*. International Thomson Computer Press, 1995. ISBN 1850321973.

8. Microstar, Near & Far - http://www.microstar.com

9. DSSSL - http://www.jclark.com/dsssl/
This has links to a number of relevant resources.

10. sgmls - http://www.jclark.com
Note that a replacement for sgmls has now been made available at this site.

11. XML - http://www.w3.org/XML

12. OmniMark Technologies, OmniMark - http://www.omnimark.com

13. National Centre for Biotechnology Information, MEDLINE - http://www.ncbi.nlm.nih.gov/

14. Dickerson, Paul. "Let Me Tell Us Who I Am: The Discursive Construction of Viewer Identity", *European Journal of Communication*, Vol. 11, No. 1, March 1996, pp 57-82.

15. Elsevier Science, “TULIP Final Report”, Elsevier Science 1996, ISBN 0444825401
Also see http://www.elsevier.nl/inca/homepage/about/resproj/tulip.shtml

16. ImageMagick - http://www.wizards.dupont.com/cristy/ImageMagick.html

17. MathML Project - http://www.w3.org/MarkUp/Math/