Book reviews online

Philip Barker[1]

Human-Computer Interaction Laboratory, University of Teesside

As the number of new academic books published each year continues to rise, such that it becomes evermore difficult to keep abreast of them in one's discipline, the book-review procedure takes on an increasing importance. This paper outlines the design and development of an automated system for handling book reviews. Descriptions are given of some prototypes that have been developed for use on an intranet server and/or the Internet. These systems, based on SGML and HTML, are briefly discussed and compared.

Introduction

Many thousands of books are published each year, and even specialists find it difficult to keep abreast of new books in their disciplines, learning technology being no exception – indeed, in our subject-area the situation is beginning to reach saturation point. The book-review procedure facilitates selection: a good review will capture the essential content of a book, and will comment on its quality, style, level of presentation, appropriateness, and perhaps value for money. Figure 1 shows the review process, its relationship to the production of books and learned journals, and the functional similarity between book reviews and abstracts of papers published in learned journals. Abstracts are often archived in online databases or on CD-ROM, in this way acting as an alerting and brief reference service. Book reviews can be treated in the same way. This paper discusses the use of servers (Internet or intranet) as a means of making them available to a global population. It also describes how such a facility could fit into a more general infrastructure for soliciting potential reviewers and drawing their attention to publications available for review.

Analysing book reviews

In order to describe a system and the relationships between its component parts, it is useful to employ some form of meta-language or meta-notation in which each of the components of the system is represented as a meta-variable (Alexander, 1987). The relationship between the meta-variables that define the top-level moieties of the system is then expressed using a series of meta-rules. The purpose of a meta-rule is to define a given meta-variable in terms

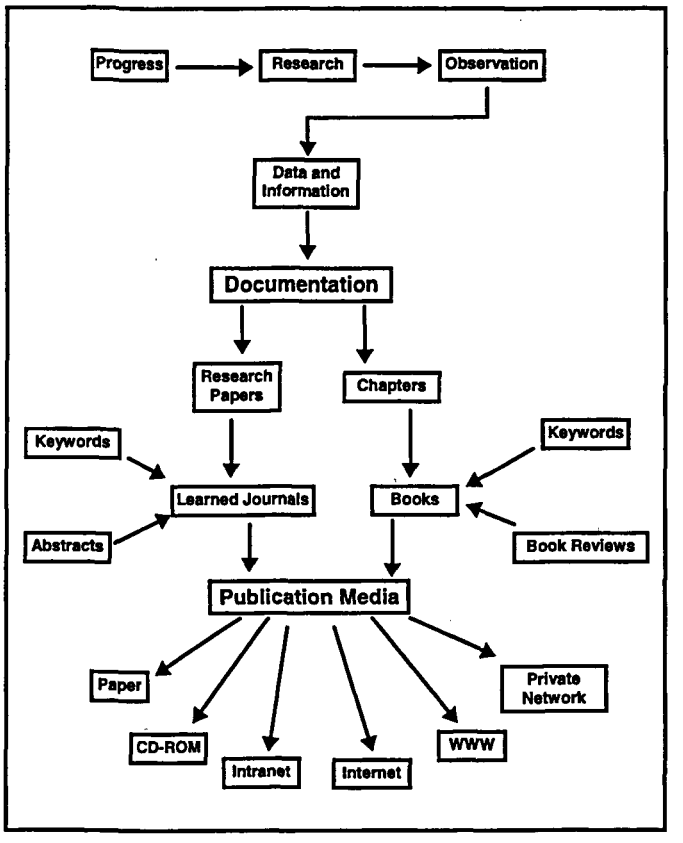

Figure 1: The context of book reviews

of more elementary units, or indeed the axiomatic entities of the system being described. Suppose, for example, one wished to describe the logical structure of a book using this technique. The first step would be to identify the book's building blocks. The second step would involve formulating a series of meta-rules that specify the relationships between the basic components identified. The structure of a text-book could thus be formulated in the following way:

```
<text-book> ::= <front-cover> <body> <back cover>
<body> ::= <contents-pages> <chapters> <index-pages>
<chapters> ::= ... <chapter>
<chapter> ::= ... <page>
<page> ::= ... <text-line>
<text-line> ::= ... <word>
<word> ::= object (X) : X isin dictionary (Y)
```

In this example, the meta-variables that identify the basic building blocks are each contained within angle brackets (for example, <body>, <chapter>). Each line defines a meta-rule, and within meta-rules various operators show how one meta-variable relates to others. Three primitive operators have been used: the composite symbol ::= means 'is defined to be'; the ... operator means 'is a replication of one or more'; the : operator means

'such that'. Logical predicates are also used. A predicate takes a proposition about an object or relationship and determines its truth or falsity. In the book definition given above, two predicates have been used: 'object' and 'isin'. The 'object' predicate checks that a candidate object, X, exists and that it is of the correct type (that is, a <word> object). The 'isin' predicate checks for membership of the instance X in the dictionary Y. If X is not found in the dictionary, the rule fails and the object X is rejected as a valid instance of <word>. This meta-notation is similar to that defined by the British Standards Institution (BSI, 1981).

The definition of <text-book> presented above is not complete: the meta-rule sequences that define the meta-variables <front-cover>, <back-cover>, <contents-pages> and <index-pages> have been omitted. In a complete definition, these objects would need to be defined in terms of lower-level meta-variables or terminal (axiomatic) entities. But despite its incompleteness, the definition presented is sufficient to illustrate the basic principles involved. Expressed in plain English, the definition of <text-book> given above would read as follows:

> A text book is made up of a front cover, a back cover and an inside (or body). The body is made up of a collection of pages. The pages at the front of the book identify its contents, those at the back make up the index, and those in the middle are organized into a series of chapters. Each of the pages within a chapter is made up of a sequence of lines containing words. All the words used in any given line must exist within a dictionary whose name is Y.

How can this system be applied to the definition of a book review, which contains not only an assessment of the content and worth of a book, but also important items of information (attributes), such as the names of the author(s)/editor(s), the publisher, the place of publication, the series of which the book may form a part, its ISBN, its cost, the number of pages it contains, its format (hardback, paperback), and so on? The meta-notation approach used for the definition of a text-book above can be used to express relationships between these attributes, as follows:

```
<book-review> ::= <book-details><keywords><review-
   body><reviewer-details>

<book-details> ::=<title><author-list><publisher-
   details><other-details>
<publisher-details> ::= <publisher><place-of-publication>
<other-details> ::= <date><ISBN><book-type><size><cost>
<book-type> ::= 'hardback' | 'softback'

<keywords> ::= ... <keyword>
<keyword> ::= isaword (X) & X isin dictionary (Y)

<reviewer-body> ::= ... <paragraph>
<paragraph> ::= ... <line>
<line> ::= ... <word>
<word> ::= isaword (X) & X isin dictionary (Z)

<reviewer-details> ::= <reviewer-name><reviewer-affiliation>
```

The process of specifying the structural units of a book review using a meta-notation paves the way for tagging a document using a markup language such as SGML (Standard Generalized Markup Language) or HTML (HyperText Markup Language). These markup tools form the basis for numerous online retrieval systems that can be made available on servers.

The review process

In order to achieve the development of an automated system for handling book reviews, it is necessary to consider in more detail some of the ancillary processes relating to the review procedure. A schematic representation of some of the more important supporting actors and processes involved in a reviewing operation, and their inter-relationships, is depicted in Figure 2.

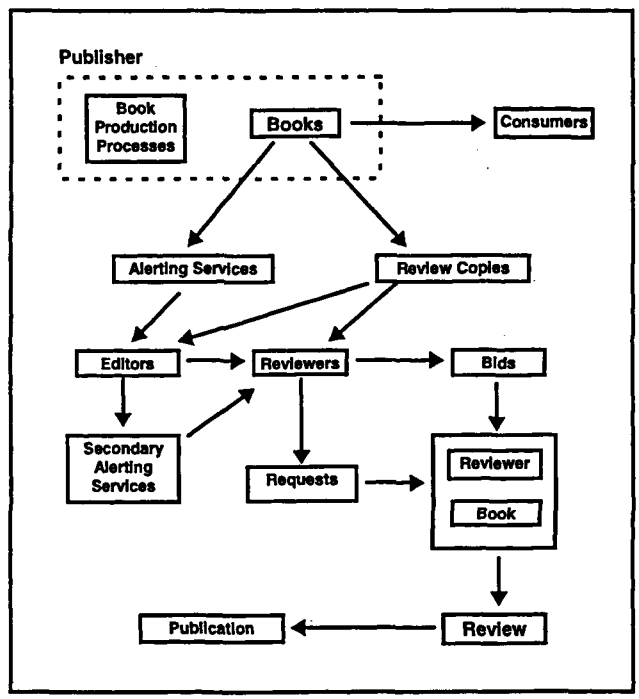

Figure 2: Principal actors and processes involved in book reviews

Most journals have a person responsible for handling review material – the reviews editor – who is usually chooses and acquires review copies of books. Sometimes these copies arrive unsolicited from publishers. More often, review editors have to ask to be supplied with copies of books relevant to their journals. Review editors can build up a buffer stock of books for review, and reviewers have to be found for these and matched to appropriate books. This process can be initiated, for example, by listing titles available for review in the reviews section of the journal in question, or by sending out a printed supplement as an insert to that journal. Alternatively, various electronic alerting mechanisms can be used, such as an email facility linked to a listserver. This is the case with *ALT-J*, where a database list of the books received from publishers for review is built up and emailed to potential

reviewers whose names are held in a listserver. People interested in reviewing books on the list then reply (by email) using appropriate code words to identify the book(s) they want to review. Invariably, many bids are emailed back, and a selection mechanism, based on the judgement of the reviews editor, is then used to allocate a title to a particular bidder.

Another electronic alerting/awareness facility we have been exploring is based on the use of the Web. A list of books available for review is maintained on a Web server which potential reviewers can access. Details of bidders and the books they select are automatically emailed back to the reviews editor. The user-interface also enables reviewers to make suggestions about other books that should be included in the list, and to make offers to review new books they are aware of. The main problem with this Web-based approach is that someone has to be responsible for maintaining the online list of books for review. We are investigating ways of automating the process. One advantage of using the Web for electronically processing book reviews is that the reviews can also subsequently be made available online to all those interested.

Towards an online review system

In order to explore the possibility of creating an online database of book reviews which could be accessed via Web, we created three prototype systems, one developed using SGML, the other two using HTML. The structure for <book-review> (see above) formed the foundation for each of the online implementations. It served as a basis for identifying search keys, stimulating thinking about system organization and screen layout; and specifying the display styles to be employed. The other main purpose of the meta-linguistic representation of <book-review> was as a means of identifying the various elements and entities that were subsequently to be used in the formulation of the SGML Document Type Definition (DTD). For example, meta-variables such as <title>, <author>, <review-body> and <paragraph> mapped directly across to corresponding SGML tags, namely </title>, </author>, and so on. The importance of the DTD lies in the fact that it forms the underlying basis for consistency-checking when marked-up instances of <book-review> are admitted into an electronic library-cataloguing system.

During the initial stages of the project, three perspectives were identified as being important to the design process. First, that of system administrators who would need to set up and maintain an instance of the system at any given site. Second, that of book reviewers who might use it to gain indirect access to material for review purposes. Third, that of readers who would access the reviews. The ways in which these different perspectives influenced the system design and the functionality embedded in the user-interface are discussed below.

Basic system design

As a result of surveying various potential users from each of the three categories identified above, we found that typical users of the system would wish to be able to retrieve a review from the database, do a library search to see if a book of interest was available in a local library, request a book either for purchase or using an inter-library loan service, see a list of book titles waiting to be reviewed, and perform system-maintenance operations. Obviously, the detailed system implementation instigated at any given site would depend on the overall context in which a given instance of the system was to be used and the

population it was intended to serve. It was therefore imperative that the design and implementation should lead to a flexible approach.

One of the most important design tasks was choosing the search mechanisms to be used for retrieving reviews. From a needs analysis, we found that typical users would most likely want to search for a review based on a specification of relevant keywords. In a review, these might occur in the title of the book, in the body of the review, or as part of its <keyword> section if one has been used (we do not use one for *ALT-J*, although we may do in future). In addition, some users suggested that retrieval by ISBN and/or author(s) might also be useful.

Implementation

Three implementations of our online book-reviews system were undertaken, referred to here as P1, P2 and P3.

P1 was implemented using an integrated SGML development and display system called IADS (Interactive Authoring and Display System) produced by the Automated Publications Division of the Integrated Material Management Centre of the US Army (Templeton, 1996). The system consists of three facilities: (1) a visual interactive editing system for authoring SGML documents; (2) a style sheet for specifying the visual appearance of documents; (3) reading and navigation tools (hyperlinks). We used this system to create an electronic library of book reviews and, in so doing were able to assess the merits of SGML for this type of application. Our findings are briefly discussed later in this paper.

P2 was created using HotMetal Pro (Version 2), a low-cost HTML editing system. It was implemented on a notebook computer as a proof-of-principle experiment, the main purpose of the prototype being to explore the potential of HTML markup techniques for this type of application. It was also used as a demonstrator to show potential users the kinds of facility a full implementation of the system would make available, and it provided the opportunity to obtain some valuable feedback from users. The implementation used simple HTML hyperlinking techniques in order to create a three-level menu-driven structure. The top level contained an introductory page which gave basic instructions and listed the search options available, and which embedded hotlinks to three second-level pages that were responsible for handling retrieval by author, title and keyword. The hotlinks embedded in the second-level HTML modules pointed directly to entries at the third level – the book reviews themselves (which were all marked up in HTML). This prototype had a number of inherent limitations (see below) and a third prototype, P3, was developed in order to overcome them.

P3 also used HTML but with more sophisticated techniques involving the use of the Common Gateway Interface (CGI) standard (Gundavaram, 1996). The design approach underlying the creation of this prototype was driven by the need to realize four basic objectives: (1) to avoid any requirement for editorial users to have any knowledge of markup languages; (2) to provide an easy-to-use system; (3) to add a search engine; (4) to produce a system that was easy to maintain. A series of CGI scripts and Perl (search-engine) programs were mounted on a Unix-based intranet server. The suite of programs had to perform the following processing tasks: (1) extract a series of keywords entered (by the user) by means of an online HTML form running on a client computer; (2) search a textual database of book reviews (held on the server) for instances of the keywords (using

AND or OR logic); (3) build a list of reviews containing the keywords entered as search terms; (4) deliver the list of candidate reviews to the client computer for perusal by the user; (5) allow users to look at individual reviews in their hit-list.

As a result of the formative evaluation that was undertaken using prototype P2, a number of other facilities were identified as possible additional requirements for the P3 system. These included the provision of an online help facility, making available mechanisms for searching the University's online book catalogue (to check for the availability of books), and the provision of facilities to enable the electronic submission of inter-library loans and book-purchasing. Each of the five primary functions mentioned above that users might wish to perform was represented by a iconic button embedded in the user-interface. Various levels and types of help were available, depending on the type of user requesting it (editor, administrator, reviewer or reader) and his/her level of experience. In order to control the basic search process, users could enter a list of keywords into a text-entry box. The interface controls also allowed users to set various search parameters (such as case sensitivity and search logic) by choosing options from pop-down selection boxes. Once all the search values had been set, a submit button transmitted the query to the server for processing. The hit-list resulting from a search was subsequently shown in a dynamically created HTML page which acted as a menu providing access to the reviews associated with the entries in the current hit-list. Because P3, like P2, was a prototype, its interface also contained a feedback button, allowing users the option of sending back comments.

Throughout this project we were conscious of the need for evaluation of what was being developed. Different methods were employed in order to undertake both formative and summative assessments of the P3 system including the ease-of-use, look and feel of the user-interface, the responsiveness of the system, and users' attitudes and perceptions about the potential utility of book reviews as an online information retrieval facility. The feedback button generated some simple questions relating to ease of use, levels of satisfaction with the search results and search time, and general usefulness of the system. The responses were aggregated in order to produce a cumulative response file that could be used to create dynamic histograms representing the views of those users who chose to submit some feedback. Details of the evaluation have been documented in two student dissertations (Bouvier, 1997; Tan, 1997).

Discussion

The results obtained from the evaluation, and our own experience of all three prototypes, allowed us to see a number of their shortcomings.

SGML is undoubtedly a very powerful markup facility, but in contrast to HTML, we did not find it at all easy to use. Furthermore, we found that we could not transport the <book-review> DTD (associated with the IADS system) to other SGML browsers such as SoftQuad's Panorama Pro. We also experienced substantial difficulties during our attempts to set up the types of search facility we wanted to use.

P2 was rejected because it was a very simple implementation that did not contain any search mechanism, and all reviews had to be retrieved by menu selection. It was also awkward to maintain, and would undoubtedly have become cumbersome to use when a substantial number of book reviews were made available.

There were three main shortcomings associated with prototype P3. First, the book reviews held in the online database were stored as plain text files, so lacked any attractive layout attributes. Second, the underlying search engine that was used to retrieve reviews had to search all the files sequentially in the database for occurrences of the keywords associated with a query. We did not find this prohibitive in terms of search time, but with a very large database it is an issue which we may need to address. Third, the system made extensive use of server-side processing involving a considerable number of data transfers between the server and the clients it serviced: the response time to queries could be subject to delays caused by other traffic on the network.

Although the use of a proprietary database system (such as Oracle or dBase) might have been a way to proceed, we decided against the use of such packages, primarily because we wanted to generate implementations that were very easy to operate and install without being dependent on the availability of commercial software.

Future possibilities

As we gain more experience with our online book reviews system and expose it to a greater number of potential users, so we begin to perceive more ways in which it could be used. We also begin to realize some of its current limitations, and foresee the need to review its design and implementation.

It is anticipated that as larger numbers of book reviews need to be handled, we may have to redesign the search engine. A related issue will be the provision of mechanisms to enable users to handle large hit-lists. It is possible that this could be achieved by allowing users to specify how many hits they want to see, and/or by providing a range of selection algorithms (from which users can choose) that can be used to determine membership of the hit-list. For example, a user might wish to see only five reviews and that the hit-list should be based (say) on selecting those reviews which have been most often referenced by other users.

An important overall future development of our system will be its integration into both the new digital library infrastructure at Teesside, and our existing conventional library system. With this in mind, we intend to make three ancillary support services available: (1) mechanisms to search online catalogues in order to assess the availability of books; (2) facilities for the electronic submission of inter-library loans; and (3) methods to enable users to submit electronic library-purchase orders.

Other future development work we hope to undertake will include a re-assessment of the potential of SGML in environments other than that provided by IADS, and an investigation of the potential role of Java for the implementation of applets that will embed some of the system functionality described above.

Conclusion

Because of their importance as an information storage, dissemination and retrieval mechanism, it is all but unthinkable that books will become obsolete in the foreseeable future, even though, with the advent of low-cost electronic publishing mechanisms (particularly the Internet), it is likely that some changes in their form, appearance and

properties will take place (electronic books, for example, are now starting to become more widely available). Given the richness of books as an information/knowledge resource, effective mechanisms are needed in order to retrieve details of them. Although much more evaluation work remains to be undertaken, the initial studies outlined in this paper suggest that an online book-reviews system could form an effective mechanism for handling book-based material. It would of course also be feasible to apply the concept to other types of publication such as sound recordings, films, videos, and multimedia productions.

Note

1. Professor Philip Barker is *ALT-J* Reviews Editor.

References

Alexander, H. (1987), *Formally-based Tools and Techniques for Human-Computer Dialogues*, Chichester: Ellis Horwood.

Bouvier, E. (1997), *Book Reviews Online*, Final-year BSc Computer Science Dissertation, University of Teesside.

BSI (1981), *BS-6154:1981 – Method of Defining Syntactic Meta-language*, Bracknell, Berks: British Standards Institution, Technical Standards Ltd.

Gundavaram, S. (1996), *CGI Programming on the World Wide Web*, Sebastopol CA, USA: O'Reilly.

Tan, C.M. (1997), *Hypermedia Electronic Books*, Draft PhD Thesis, University of Teesside.

Templeton, R.S. (1996), 'The Interactive Authoring and Display System (IADS)', downloaded from *http://shodan.redstone.army.mil/index.htm.*