Personalising Electronic Books

James Ohene-Djan
Department of Math. & Computing Sciences
Goldsmiths College, University of London
New Cross, London SE14 6NW, UK
email:j.ohene-djan@gold.ac.uk

Alvaro A.A. Fernandes
Department of Computer Science
University of Manchester
Oxford Road, Manchester M13 9PL, UK
email: a.fernandes@cs.man.ac.uk

Abstract

This paper addresses the issue of how hyperdocuments, accessible via electronic books (E-books) which are read using the World Wide Web, can be endowed with features that enable the personalisation of the interaction process that takes place between the reader and the E-book itself. We introduce a novel, abstract approach to modelling the personalisation of hyperdocuments. This approach aims to make available to readers, features that allow them to interact with these documents in a manner much closer to that of owners of paper-based ones. This research is based upon a formal characterisation of personalisable hyperlink-based interaction. This characterisation is unique in formally modelling a rich set of user-initiated personalisation actions that allow users to come closer to satisfying their specific, often dynamic, information retrieval goals.

1 Introduction

The first speculative discussion of the concepts underlying the notion of utilising technology to display books to readers can be found in Bush's proposal in 1945 for a system called *memex* ("Memory Expander") [12]. Bush described the memex as "a sort of mechanised private file and library". After describing the mechanics of memex, Bush stated that

> *All this is conventional, except for the projection forward of present-day mechanisms and gadgetry. It affords an immediate step, however, to associative indexing, the basic idea of which is a provision whereby any item may be caused at will to select immediately and automatically another. This is the essential feature of memex. The process of tying two items together is the important thing. [12]*

This quotation highlights the importance Bush placed on the user in the process of constructing associative relationships between units of information. An important conclusion that can be drawn is that the memex system was envisaged to be personalisable, in that a user could make an associative relationship or link between two units of information. It can be observed that even today, the vast majority of hyperdocuments [1] which are read using the World Wide Web (WWW) as E-books, provide each user with the same associative relationships (retrieval possibilities) disregarding user-specific information goals and histories. Furthermore, all decisions as to the association of information units are decided by the designer of the system, with users being given no opportunity to interfere with these decisions.

In this paper we define the content of an E-book to be a network of digital information units which may comprise text, graphics, video, animation and/or sound. When these information units are rendered, they provide the user with optional links to other information units. Such links can provide context-based, non-linear navigation between information units. In this paper a rendering unit in a E-book is viewed as a hyperpage, and a collection thereof as a hyperdocument read using a WWW browser or handheld device. The core of this definition is content, a digital object containing an electronic representation of a book whose pages are interconnected by hyperlinks. More generally, an E-book is viewed as an electronic version of a paperback or cloth-bound book. In its digital form the content of an E-book may be dynamically generated on the fly. It may be accessible world wide, via the Internet

[1] In this paper a rendering unit in a E-book is viewed as a hyperpage, and a collection thereof as a hyperdocument

and read using a E-book reader such as a WWW browser. An E-book reader is defined to be a electronic device capable of displaying E-books. E-book software operates on an E-book reader providing book display functions. Personal computers, PDAs, pocket PCs and dedicated readers such as Rocket E-book [22] and Cybook [16] are E-book readers, while WWW browsers such as Internet Explorer [23], MS Reader for the pocket PC [37] and Palm Reader Pro [42] are examples of E-book reader software.

Although there have been many impressive proposals for the development of E-book technologies [26, 43] and much research has been conducted into their deployment [31] and design [49, 32, 34, 35], a significant limitation of many E-books read using the WWW is their inability to address users' information needs on an individualised basis [6, 10, 3]. Most E-books provide distinct users with the same information retrieval possibles and present that information in the same manner. If one considers the question of how users interact with a book over extended periods of knowledge acquisition, and the actions that they are able to take that would be difficult or impossible if the book were a hyperdocument, it can be argued that users of hyperdocuments are limited in the functionality available to them [8]. For example, when a user interacts with a paper-based book, they are given the opportunity not only to read its content in a linear manner, but also to, annotate it, insert and delete content, bookmark and highlight particular sections and to make, in context, cross references to parts of the book and to other books. Such functionality is rarely found in E-books read using the WWW.

Personalisable, adaptive hyperlink-based systems (PA-HLBSs), more often referred to as *adaptive hypermedia* (AH) [8, 7] constitute an area of research that aims to enhance the functionality of hypermedia systems, such as hyperlink-based E-books [32], by making the user interaction process personalisable. The approach taken is to endow systems with personalisation features which may be initiated by the users or by the system itself. AH are assumed to be useful in areas, such as learning, where users have different information seeking goals, histories and preferences. AH aim to use knowledge provided by (or captured about) specific users to tailor the information and the links presented to each specific user. By applying the knowledge of users they amass, AH can support users in navigation by limiting the options for traversal to information units, suggesting relevant links to follow and providing additional information on links and information units. In this paper we propose a system-independent approach to personalising hyperdocuments as a value-adding strategy for E-book technology. This work is based upon research [41] that has contributed a formal model of personalisable, adaptive hyperlink-based interaction. The main contributions of that model were a formal definition of core hyperlink functionality and a formal definition of hyperlink-based personalisation through extensions to this core. Such extensions allow for the specification of hyperdocuments that facilitate user-initiated personalisation of their content.

The remainder of this paper is structured as follows. To place our work in context, Section 2 provides a brief background to our research. To motivate the contributions of the paper, Section 3 contrasts the actions that users of paper-based books can perform with those that users of hyperdocuments cannot, and vice-versa. Section 4 introduces a framework for the personalisation of hyperdocuments. Section 5 uses an example to show how the framework proposed allows the desiderata identified to be met. Section 6 contrasts our results with those of other researchers. Section 7 draws conclusions.

2 Background

In this section we briefly outline some issues, concepts and principles of our research. The section begins by describing why hyperlink-based interaction, as a model for information retrieval, is a natural candidate for modelling and deploying E-books. Following this, several fundamental issues relating to the design of personalisable hyperdocuments are outlined, together with some principles for personalisation, that have emerged from preliminary research.

2.1 Why Personalisable Hyperdocuments?

In order to understand PA-HLBSs research and its potential importance to the design of hyperdocuments accessible via E-book technologies, it is useful to consider first how users seem to read and write documents. Human intelligence depends upon models with which people communicate. Broadly, it is believed that one associates pieces of information with other information to create complex knowledge structures or networks of information. Such a network forms a *mental model* which may be viewed as a semantic network comprised of units of information represented as nodes, and associations between units represented as links [45]. Communicators must be able to persuade the audience that acceptance of their communicated mental model is in the best interests of the audience.

During this process, the typical medium (i.e., paper, video) can only capture a sequential presentation. An author must transform their non-linear communication (mental model) into a linear form. Authors have to linearise their

mental model before it can be communicated. As the linear representation is not natural, authors may provide additional information (i.e., tables of contents and indexes). Readers, in turn, transform the linear message into their own non-linear models in their minds. Generally this process involves the reader breaking up the information into smaller parts and then rearranging these parts based on their information goals. This process is shown in Figure 1.

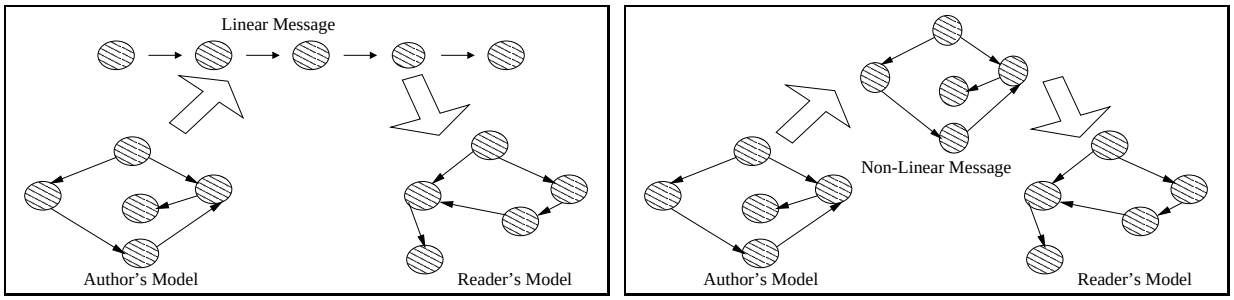

Figure 1. Reading and Writing using Traditional Linear Media (left) and HLBSs (right)

In [47] a writing model is suggested which consists of three cognitive phases: an *exploring phase* during which knowledge is formulated, initial drafts are made, and ideas are grouped into different perspectives; an *organising phase* during which initial drafts are organised into sequence and an outline is made; an *encoding phase* during which the document is written.

Although these phases may be undertaken in sequence, generally authors like to move freely from one phase to another. In [21], the following activities are described: *Recording relevant ideas*, where initial drafts of ideas are made; *Exploring relationships among ideas*, which is the associative linking process; *Structuring ideas*, which is the process of organising drafts and creating an outline; *preparing the document*, during which the document is written, linearised, formatted and printed.

Research into PA-HLBSs is motivated by their potential to enable authors to partially mimic the processes of writing described above. The logical model of HLBSs is largely that of a semantic network based on the associative linking of units of information attached to the nodes of the network. The analogy of a semantic network to a hypernetwork is straightforward and has long been recognised [15, 44]. PA-HLBSs allow authors and users to directly create networks that communicate their mental models through the use of computer supported links and annotations. Authors and users may not only recreate their mental models as non-linear information structures (hypernetworks) but also tailor these structures, overtime to satisfy there information retrieval needs on an individualised basis, as depicted in Figure 1.

2.2 Designing Personalisable Hyperdocuments

It is now generally accepted [6] that users of HLBSs may differ in their information goals insofar as they may have preferences as to what information is provided and which links are used to navigate the information space. Users may also differ in their histories insofar as they are likely to have different knowledge of the information contained within the HLBS, of the information space and how it may be navigated.

Most of the interaction a user might experience with a hyperdocument is determined by the design decisions that shaped the hyperdocument in terms of its content, rendering and navigation possibilities (i.e., its links). Because these decisions are unilateral and irreversible (i.e., cannot be overridden by users as they interact with the hyperdocument), it can be said that the designers *own* the hyperdocument.

This paper proposes one approach to overcoming this impediment by extending HLBSs with formally defined personalisation actions that effect a transfer of ownership from former designers of the hyperdocument to each of its users, thereby enabling the latter to redesign the former according to their specific information goals and histories.

The challenge is, therefore, how to model, at a suitable level of abstraction, the space of possibilities for personalisation actions that could be made available to beneficiaries of PA-HLBSs. Such a model should characterise the notion of "transfer of ownership" and should avoid being technology-driven. Furthermore, the choice of personalisation actions should fall out from this abstract model of interaction and should ultimately be subject to empirical tests for effectiveness gains, although this last desideratum is not tackled in this paper.

In answering this challenge, the following derived problems present themselves:

1. How to model interaction with a hyperdocument?
2. How to transfer ownership of the process of interaction with a hyperdocument from the designers to its users?
3. How to make design decisions more explicit?
4. How to distinguish, model and implement personalisable hyperdocuments?
5. How to distinguish, model and implement personalisation and adaptivity?

A further goal is to incorporate the following principles for personalisation, which emerged from preliminary research:

1. Personalisation should, to some degree, represent the measured transfer of ownership of the process of interaction with a hyperdocument from designer to user;
2. All hyperpage design decisions should, in principle, be able to be the subject of personalisation requests;
3. A model of hyperlink-based personalisation should aim to accommodate all recognised personalisation actions (see [6] for a comprehensive review);
4. Personalisation actions should be clearly defined, explicit and capable of being formally defined;
5. All personalisation actions should be consistent, repeatable and revisable;
6. Any choice of personalisation actions should ultimately be subject to empirical tests for effectiveness gains.

3 Motivation

This section discusses the motivation for ownership transfer via personalisation as a value-adding strategy. It does so by contrasting the kind of personalisation actions that owners of traditional learning materials (such as printed books) and users of HLBSs can carry out.

Assume users are Computer Students. As part of their degree, students are required to take a course on e-commerce, for which the textbook is *Electronic Commerce* by G Schneider and J Perry [46]. A small section of the book is depicted in Figure 3.

Consider the question as to how students interact with a textbook over extended periods of learning. Furthermore, consider interactions which would be difficult or impossible if the book were a hyperdocument. Some of the personalisation actions that fall into this category include:

- *Selective Reading*: To open and read some page(s) only (possibly consulting the table of contents or index) and perhaps only partially. In current hyperdocuments, options for traversal are defined by the links the designer of the hyperdocument has embedded in it. Users are not provided with facilities to make links between pages and therefore define their own paths through the material. Also, when a user does want to traverse a link defined by the designer there is no metadata about the target thereby making it impossible not to commit to the traversal. Since traversing links can incur high computational costs, frustration sometimes ensues;
- *Annotation*: To add to the original printed text handwritten material that, for the student, enriches the meaning of the text. Examples include underlining or highlighting a passage, writing comments in the margin, cross-referring to some other page of the book or some other information resource (e.g., a website, another book, etc.). At present, such features are rare;
- *Content Insertion and Deletion*: To add content (e.g. copied from another book, or a website) at specific points and to delete (or cross out, not necessarily physically) parts of the text. Features to enable the users of a hyperdocument to tailor its content have also not been implemented widely. In particular, users are not often given the ability to remove parts of a hyperdocument which they deem inappropriate for study at a point in time;

- *Bookmarking*: To leave markers at particular pages (e.g., to indicate that they are more important in some respect than others). Note that marked pages are easily identifiable for selective reading. Although WWW browsers allow users of hyperlink-based learning systems to bookmark pages, such bookmarks are often limited to recording only the Internet address (location) of a page. Users are not normally given the opportunity of indicating the importance of some bookmarked page over another or why a particular page is bookmarked at all.

Each of the above actions are admissible on paper-based materials that the student owns but markedly difficult on hyperdocuments because the student has no means of claiming ownership of the material nor of personalising the latter (even if ownership could be transferred from the designer to the student, because the student may not have the designer knowledge required for personalisation to be carried out).

Now consider the question as to how students would interact with a textbook implemented as a hyperdocument. Furthermore, consider interactions which would be difficult or impossible if the book were *not* a hyperdocument. Some actions that fall into this category include:

- *Context-Based, Nonlinear Navigation*: To benefit from the availability of contextual links at points deemed appropriate by the designer of the hyperdocument and traverse the material nonlinearly;
- *Dynamic Information Management*: To make use of computational resources to react and respond to requests dynamically (e.g., to benefit from a web of information comprising many, possibly remote, independently generated and maintained interrelated hyperdocuments).

Each of the above actions are admissible on electronic materials that the designer prepares, but markedly difficult on paper-based documents even if the student owns them.

In summary, although users of hyperdocuments benefit from the concept of linking and from the fact that electronic media typically exist in a computing environment, there are many useful actions that the owner of paper-based documents can take that are not open to users of hyperdocuments.

4 Framework

This section describes a framework for the personalisation of hyperdocuments that aims to make available to users of, e.g., electronic learning materials, features that allow these users to interact with the hyperdocuments in a manner much closer to that of owners of paper-based ones, while still benefiting from the ability to traverse contextual links and to exploit the computational environment in which the material is embedded.

A general, open architecture for HLBSs, of the kind depicted in Figure 2 as a simplified data flow diagram, is assumed. Here we propose one view of the interior of the shaded oval in Figure 2.

It is assumed that hyperlink functionality is provided as a client technology dependent on loose couplings to (at least) a user-interface server (UIS) and a database server (DBS). The classical example of a UIS is a WWW browser (e.g., Netscape Communicator). Among other functions, browsers broker requests and have rendering capabilities. Browsers can render formal texts (e.g., rendering expressions authored in HTML or XML). Examples of DBSs are database management systems (DBMSs) which support client server architectures (e.g., Oracle, or MySQL).

Broadly, the dynamics associated with Figure 2 are as follows. The UISs capture requests for desired hyperpages. The UISs channel requests for hyperpages into the HLBS proper. If a request is for a hyperpage which resides in a remote HLBS, then the core of hyperlink functionality interacts with it to obtain the requested hyperpage in the form of a rendering expression that the core can pass back for the UISs to render. If the request is for a local hyperpage (e.g., one which is known to the core) then the latter responds by returning a rendering expression to the UISs, possibly after querying one or more DBSs to fetch some or all of the content specified for the requested hyperpage.

Implicit in Figure 2 is the assumption that personalisation actions in HLBSs should not, and need not, be compounded with personalisation actions that might be provided by user-interface and database components in HLBS architectures. The shaded oval in Figure 2 is responsible for what users experience as hyperlink-based information retrieval. Notwithstanding the fact that users may well want to personalise database and user-interface features, clearly it can be argued that whatever is in scope for personalisation actions *in HLBSs* resides in the shaded oval.

To model adaptive, personalisable hyperlink-based interaction, a framework is proposed in which the shaded oval in Figure 2 is partitioned into regions. Non-adaptive, non-personalisable HLBSs are modelled by a group of functions referred to as the *H-region*. Personalisable HLBSs require the addition to the H-region of the functions

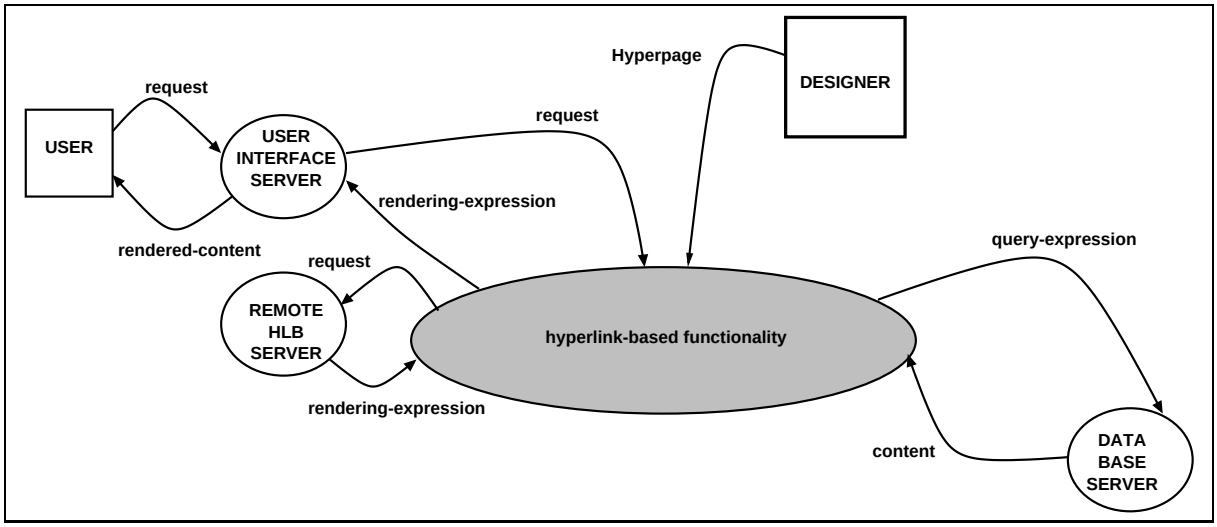

Figure 2. An Architecture for HLBSs

provided by the *P-region*. Adaptivity, seen as user-specific, system-initiated personalisation, comprises another region of functionality but is not within the scope of this paper (see [41] for details).

4.1 Core Hyperlink-Based Functionality

The H-region models a core of hyperlink-based functionality. Conceptually, the H-region behaves as a *composer of hyperdocuments from specifications*, i.e., what the designer of a hyperpage designs is not a hyperpage, but rather a specification of how to build the hyperpage upon request. Hyperpages are modelled as formal specifications and a formal language has been defined for this purpose [41].

Within the H-region, users can only request for hyperpages to be rendered. The decisions that the designers of a hyperdocument have made with respect to content, navigation and rendering cannot therefore be overridden. Upon a request arriving from a UIS, a composition function parses the hyperpage specification into a series of actions that, when executed, convert the specification into renderable text that is sent to the UIS as the response to the original request.

The semantics of a hyperpage specification (of which an example is shown in Figure 4) have been formalised as a program which, when interpreted, typically fetches content from a DBS, composes the content into a renderable text (making use of template variables as a binding mechanism) and finally responds to the original request with renderable text. Figure 5 indicates how the hyperpage specified in Figure 4 might be rendered.

4.2 Personalisable Hyperlink-Based Interaction

The P-region comprises a group of functions that are non-disruptively added to the H-region in order to model personalisable hyperlink-based interaction. Personalisation is viewed as the process of handing over to the user the ability to *annotate specifications or rewrite them or both*, thereby allowing the user to override, in principle, each and every designer specification. Therefore, within the P-region users can not only request a hyperpage, but also annotate or rewrite it, thereby creating their own version of it. The decisions that the designers of that hyperpage have made with regard to content, navigation and rendering of the hyperpage can therefore be overridden by users and this kind of event characterises ownership transfer.

When personalisation functionality is layered over the core, a designer can annotate a hyperpage in preparation for differences in users' goals and histories. A user can personalise not only such annotations (of which an example is shown in Figure 6), but the hyperpage specifications as well. Personalisation requests (of which examples are shown in Figure 7) allow users to specify which hyperpages are to be personalised and how they should be transformed.

The kinds of personalisation actions modelled are based on annotating and rewriting the hyperpage specifications. Annotation pairs a hyperpage specification with notes of interest to the user and, by doing so, assumes that versioning takes place. Such notes take one of the following forms. Firstly, a note can assign user-specific values to user-generic attributes of interest (e.g., that the level of difficulty of a given page or component part is high, or

that ‘planets’ is a keyword of relevance to a given page). Secondly, a note can specify a rewriting action over the renderable text after it has been composed by the H-region, i.e., after content has been fetched and made ready for display (e.g., to map American into British spelling forms). This form of post-composition rewriting can also be conditional on the environment (e.g., replace images with captions if the display unit is text-only).

The existence of annotations on hyperpages allows for: personalisation of a specified hyperpage; the specification of *alternatives* to a specified hyperpage; the specification of *comparable* hyperpages to a specified one and the recording of information about a hyperpage (i.e., what are the current values of attributes set by previous annotations).

Annotations are not operations on hyperpage specifications, i.e., they do not alter the latter. Rather, they give rise to a user-specific pairing with a hyperpage specification. Rewriting also causes versioning and can be characterised simply as the editing of hyperpage specifications. If a user edits the hyperpage specification as conceived by its designers, ownership is thereby transferred. If, subsequently, the same user edits that hyperpage specification again, adjustment takes place.

A formal language for annotations has been defined, as has a formal language for the personalisation requests used to maintain annotations and hyperpage specifications [41]. Personalisation requests allow users to take the following actions to enhance their interaction process:

1. *Content Tailoring*: The insertion or deletion of hyperpages and, recursively, their page parts;
2. *Selective Content*: The hiding from users of parts of the content of hyperpages which they wish not to see;
3. *Content Rewriting*: The rewriting of specified phrases in hyperpages;
4. *Prerequisite Content Selection*: The fetching and displaying of prerequisite content for a particular hyperpage or page part;
5. *Linking*: The insertion of hyperlinks between hyperpages and their page parts;
6. *Link Annotation*: The annotation of links made between hyperpages and their page parts;
7. *Link Hiding and Blocking*: The hiding or blocking of links on a hyperpage.

5 Personalisation at Work

This section uses an example to show how the desiderata identified in Section 3 are met by the functionality arising from the framework described in Section 4. Consider Figure 3, where an excerpt from [46] is transcribed.

IMAP, the Internet Message Access Protocol, is a newer protocol that has advantages over POP and that may, one day, replace POP. IMAP, like POP, can download e-mail, delete mail from the server, or simply ask if there is new mail. IMAP, however, improves upon some of the shortcomings of POP. For instance, it defines how a client program asks a mail server to present available mail. IMAP can request that the server download only selected e-mail messages rather than all messages. Your e-mail client can view just the heading and the name of the sender of the e-mail message and then decide whether to download the message. IMAP, through your client e-mail program, allows you to create and manipulate mail folders or mailboxes on the server, delete messages, and search for certain parts of a note or an entire note – all without down-loading mail from the server to your PC.

Figure 3. An Excerpt from [46]

To specify a hyperpage to correspond to Figure 3, the approach taken is to divide a page into a sequence of segments called *chunks*. Each chunk comprises a content specification (C-spec) (i.e. either hardwired text or data returned as the result of queries sent to databases, figures, etc.) and a specification of how this content should be rendered (R-spec) for presentation (e.g. via HTML tags). A chunk may contain template variables. Conceptually, a template variable is associated with bits of text or else acts as a place holder for content which becomes available after evaluation of the query associated with it. The rendering part is specified in a language which the UIS can render, except that this renderable text may be interspersed with template variables. After content is retrieved and the template variables are assigned, textual replacement takes place substituting values (i.e. retrieved content) for references (i.e. occurrences of template variables). The overall result is renderable text (e.g. HTML).

In addition, a chunk may be associated with two sets of identifiers. The first set is the set of entry points to the chunk and the second is the set of its exit points. An entry point enables the chunk (as an anchor in the page) to be referenced in a request. An exit point enables a chunk to establish navigable links to the anchor denoted by the exit point. In WWW parlance, an entry point can be thought of as a *URL* and an exit point as a *hyperlink*. A detailed formal descriptions of chunks and there component parts can be found in [41].

```
page{
 chunk{
  entry{
    [<a name="Email Message Protocol"></a>]  }
  content{
    A := 'IMAP',
    B := ', the',
    C := 'Internet Message Access Protocol',
    D := 'is a newer protocol that has advantages over',
    E := 'POP',
    F := 'and that may, one day, replace POP. IMAP,
          like POP, can download e-mail, delete mail
          from the server, or simply ask if there is
          new mail. IMAP, however, improves upon some
          of the shortcomings of POP. For instance,
          it defines how a',
    G := 'client program',
    H := 'asks a',
    I := 'mail server',
    J := 'to present available mail. IMAP can request
          that the server download only selected  e-mail
          messages rather than all messages. Your e-mail
          client can view just the heading and the name
          of the sender of the e-mail message and then
          decide whether to download the message.  }
  rendering{
    [<b><a href="imap.html">          A </a></b>] B
    [<b><a href="imap.html">          C </a></b>] D
    [<b><a href="pop.html">           E </a></b>] F
    [   <a href="clientprogram.html"> G </a>    ] H
    [   <a href="mailserver.html">    I </a>    ] J  } }
 chunk{
  entry{
    [<a name="IMAP"></a>]  }
  content{
    A := 'select definition
          from   glossary
          where  term = 'IMAP'  }
  rendering{
    A  } } }
```

Figure 4. Fig. 3 as a Hyperpage Specification

Figures 4 shows one way in which Figure 3 might be written as a hyperpage specification. It is assumed that database queries for retrieving content are written in SQL and that HTML is used as a mark-up language in the construction of a renderable text. For more detail on how the evaluation of Figure 3 into renderable text takes place, see [41]. Figure 5 depicts the hyperpage after it has been rendered by a UIS (in this case, an HTML browser).

5.1 Hyperpage Annotations

A hyperpage annotation pairs a hyperpage specification with notes of interest. An annotation is, roughly, a collection of contextualised attribute-value pairs. An example of an annotation for the text specified in Figures 4 is given

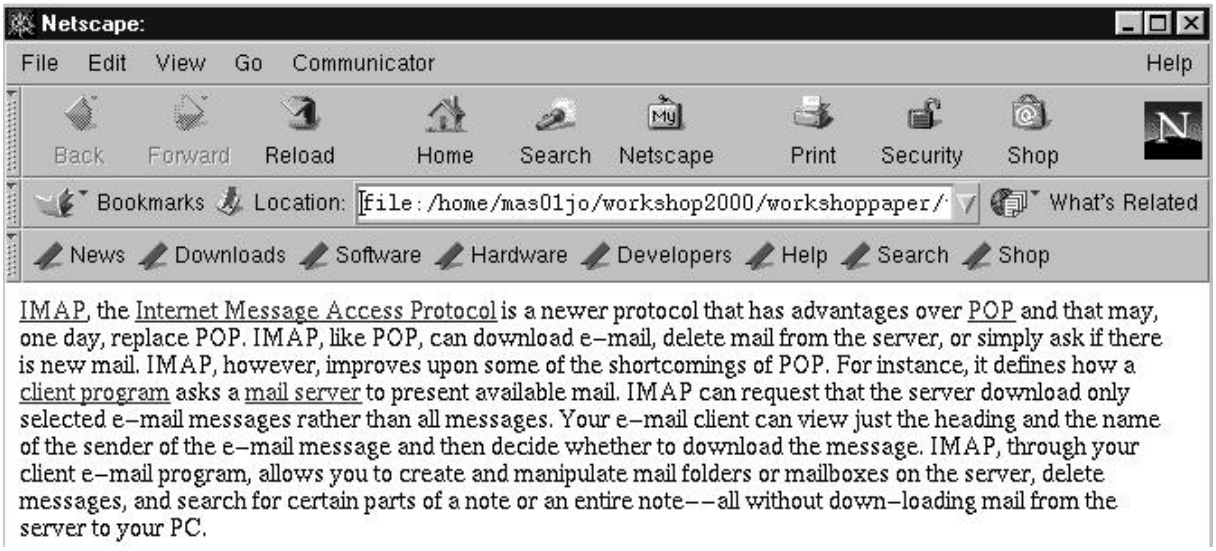

Figure 5. Possible Rendering of Fig. 4

in Figure 6. The attributes used are *description*, which lets the student or designer provide a summary or abstract; *keyword*, which lets the student or designer tag content and thereby clarify the information contained; *level*, which lets the student or designer attach a measure (e.g. of difficulty) to the text; and *see-as-well*, which lets the student or designer specify hyperpages that describe comparable information. Again, [41] provides more details.

The annotation in Figure 6 will be associated with the hyperpage specification in Figure 4. Conceptually, this versions the page specifically for the annotator.

```
annotation{
 page:
  description := 'defines and contrasts IMAP with
                 other Internet protocols for
                 e-mail services';
  level       := 1;
  see-as-well := [http://www.internetmail.com/];
  keyword     := 'e-mail', 'electronic mail';
 [1, chunk]:
  description := 'Internet Message Access Protocol'
  keyword     := 'IMAP', 'POP',
                 'Internet Message Access Protocol';
  see-as-well := '[page-60.html]'
 [2, chunk]:
  description := 'Definition of IMAP
                  retrieved as dynamic content';'
 [2, chunk (R-spec)]:
  keyword     := 'Rendered using HTML'; }
```

Figure 6. Annotating Fig. 4

5.2 Personalisation Requests

Personalisation requests allow a user (e.g., a student) to generate annotations, to update annotations (e.g., those provided at source by the designer), and to update (by versioning) the hyperpage specifications themselves. A few examples of personalisation requests (in a language, and with a semantics, formally defined in [41] that a student might issue (to the hyperdocument of which Figure 4 is a page) are depicted in Figure 7.

Example 1 in Figure 7 is a request to tailor the content found in the book. The request applies to all hyperpages (because the selection condition is vacuously `true`). Its effect is to insert the string, `"These pages belong`

```
select-page-if            % Example 1
  true
hp-then-do {
  insert [1, chunk (R-spec)]
     "These pages belong to student X"  }
                          % ----------
select-page-if            % Example 2
  [5, chunk] contains "electronic mail"
hp-then-do {
   insert [1, chunk]
     chunk {
       content  { X := 'Many of the latest
                       electronic mail systems
                       now provide support for
                       sound and video files.'  }
       rendering { [<I>] X [</I>]} } }
                          % ---------
select-page-if            % Example 3
  true
ann-then-do {
  insert page :
       "http://www.ebook.com"
    -> "http://www.ebook.co.uk"; }
```

Figure 7. Personalising Fig. 4

`to student X"` into the rendering specification of the first chunk of each page. Example 2 is a request to add content. The request is applied to all hyperpages that contain the string ”electronic mail” in its 5th chunk. The effect on each selected hyperpage is the insertion of a new 1st chunk. Example 3 in Figure 7 depicts a request to rewrite a specified string found in the hyperdocument. The request applies to all hyperpages and has the effect of inserting into the annotation of each hyperpage a request (as denoted by the arrow ‘`->`’) that all occurrences of the string `"http://www.ebook.com"` in the composed text be rewritten as `"http://www.ebook.co.uk"` in the rendering text.

The example in this section, in spite of space limitations, illustrates how hyperpage specifications, that can be annotated and maintained via personalisation requests, enable users of hyperdocuments to claim ownership by actions that version pages by means of annotations and rewriting requests.

6 Comparison

This section compares and contrasts work that is related to the framework proposed in this paper (whose origins lie in the formal model of personalisable, adaptive hyperlink-based interaction proposed in [41]).

6.1 Hypermedia Models

6.1.1 The Hypertext Abstract Machine (HAM)

The HAM [13] was the first attempt to define an abstract model within which HLBSs could be expressed. The model describes a transaction-based storage system. The definition of the HAM consists of a description of HAM objects and the operations that can be applied to them. The HAM describes a lower level machine tied closely to the storage (file) system while having a looser connection to applications and user interfaces. Although the approach taken in devising the HAM is related to the work reported in this paper, it differs from it in several important respects.

The HAM views a hypernetwork as a database of objects and the operations described are primarily those to manipulate this database (e.g., create, delete, get, filter). This approach can be contrasted with the model proposed here, which attempts to identify and characterise those operations which are unique to HLBSs as opposed to those operations present in HLBSs *qua* database applications.

Although the HAM views a core of hyperlink functionality (the HAM itself) as being loosely tied to applications and user interfaces, it views the host file system's structure and contents as being tightly coupled to the HAM. This view can be contrasted with the work reported here, in which both user-interface and database functionality are viewed as being loosely coupled to possibly remote servers whose internal structure may be unknown. How such coupling takes place is viewed as a side effect of the query specification languages and mark-up languages used to specify the content and presentation of hyperpages.

It can be seen from the specification of the HAM that its authors realised the need for features to allow personalisation of hypernetworks. The filtering mechanism described in the HAM allows subsets of HAM objects to be extracted from hypernetworks (HAM graphs), thereby allowing a user to specify a subset of objects to be presented. However, this approach to personalisation is primarily data-oriented and could be broadly understood as the specification of logical views over an underlying database. In contrast, the work reported here models personalisation of a core of hyperlink functionality, as opposed to specifying particular (i.e., personalised) database queries.

6.1.2 The Dexter Model

The Dexter model [24, 25] has become the foundation for much research into HLBSs modelling and is a reference against which many HLBSs are compared. The model is divided into three layers; the storage layer, the within-component layer and the run-time layer. The main focus of the model is the storage layer which describes a database that is composed of a hierarchy of data containing components, called nodes, interconnected by links. The storage layer, therefore, models the basic node/link network structure of a hypernetwork.

The Dexter model is similar to the work reported in this paper insofar as both attempt to characterise the concepts and operations that make HLBSs distinctive from other information systems. For example, the Dexter model describes an interface mechanism for addressing (referring to) locations or items within the content of an individual component. This mechanism, known as anchoring, provides a clean separation between the storage and within-component layers. Anchoring provides the functionality to allow links to be made between documents and also parts of documents.

In the framework proposed in this paper a content specification, plays a comparable role to the Dexter model's anchoring mechanism. A C-spec defines content which is to appear in a hyperpage. A C-spec takes the form of data values or requests to DBSs (i.e., query expressions which DBSs can evaluate into data values which are served back). A C-spec may be as simple as a number or a string and as complex as a sequence of complex queries which are to be sent to a variety of DBSs.

In the Dexter model, presentation specifications provide an interface between the storage layer and the run-time layer (i.e., UIS). Such specifications are the mechanism by which information about components presented to the user can be encoded into the hypernetwork (at the storage layer).

In the framework proposed here rendering specifications (R-specs), define how content is to be rendered by a UIS. An R-spec takes the form of formal text (e.g., mark-ups) in a language which the intended UIS can render. This renderable text may be interspersed with template variables that act as place holders for content as defined by the C-spec with which the R-spec is paired. An R-spec can therefore be used to specify how content is to be presented in a manner similar to that of presentation specifications in [24, 25].

The Dexter model does not address the issue of tailoring the interaction process that takes place between users of HLBSs and the system itself. No aspects of personalising the interaction process are explicitly modelled and no consideration is given to users of HLBSs that may have differing information retrieval needs, user goals and backgrounds.

In the Dexter model the storage layer models a hypernetwork as a known database of nodes and links. As such, the size, scope and form of the hypernetwork can be said to be determined at design time, thereby making the HLBS closed. In contrast, the work reported here views a hypernetwork as a collection of specifications to be evaluated at runtime, i.e., the size, scope and form of the hypernetwork is not known in advance, thereby making the HLBS open.

The Dexter model sees HLBSs as being layered and including inter-connected user-interface and data storage layers. An implicit assumption is made that these layers are aware of and are tightly bound to each other. In contrast, the proposed model does not view the functionality provided by these layers as being tightly bound. Instead this functionality is provided by a loosely-coupled core of hyperlink functionality served by an open collection of possibly remote user-interface and content servers. As such the proposed architecture reflects more accurately those of many HLBSs currently being deployed on the WWW.

The Dexter model implicitly assumes that applications making up the within-component and run-time layers are

known. This monolithic view cannot always be assumed to be true. The array of different presentation, link anchoring and data storage mechanisms that could potentially be employed by HLBSs implies that their behaviour and computation should not be modelled as an integral part of a HLBS. The architecture for HLBSs proposed in this paper may be extended to support such an array of presentation, link anchoring and data storage mechanisms.

In summary, although the Dexter model has provided a solid foundation for research into closed HLBSs, it is less appropriate for representing dynamic interaction patterns across different categories of user. Furthermore, it does not explicitly address issues of personalising the user interaction.

6.1.3 HyTime, XML and OEBPS

The ISO standard HyTime (Hypermedia/Time-based Structuring Language) specifies the representation of hyperdocuments in a presentation independent format. HyTime allows for the representation and synchronisation of static and time-based hyperlinked information [40, 39, 20]. HyTime is defined as a subset of the Standard Generalised Mark-up Language (SGML) [27]. HyTime supports addressing by name, by position in the document and by semantic construct. Links can be established to documents that conform to HyTime as well as those that do not. HyTime also provides a document query language.

Although the aims of HyTime and of the work reported in this paper differ, several similarities can be drawn. HyTime and the work reported here model hyperpages as specifications in which external content may be referenced. The level of openness that HyTime allows, in terms of the kinds of media and technologies that can be referenced by hyperpage specifications, is comparable to that which is allowed by the model proposed in this paper.

In both models, addressing of hyperpages and their component parts can be specified using semantic constructs. Furthermore, the ability to incorporate links which are specified using an external formal text is also shared.

Work on XML [4] bears some resemblance with ours. Both approaches seek to divide a hyperdocument into units which either specify data or delineate regions where data may be obtained. The use in XML of arbitrary-value pairs associated with a document's structure, can be compared to our notion of note. The functionality of an XML-based HLBS is not expected to be dissimilar from that of the H-region. However, we propose an abstract model of hyperlink-based interaction and not just of specification. Also, the specification elements in our framework may, without loss of expressiveness or generality, be recast in SGML.

In this paper, we argue that the hyperlink paradigm is one that is well suited to the design of E-book material. This view is supported by the proliferation of hyperdocuments formatted using the Open eBook Publication Structure (OEBPS) [48], a XML-based specification for the content, structure, and presentation of electronic books. However, it is still rare to find E-books authored in a manner that enables them to be annotated and subsequently personalised. Our future work will aim to address these issues using the results first reported in [41]

6.2 Personalisable HLBSs

Little research has been conducted into formalising personalisation in HLBSs themselves. Most work has concentrated on providing personalisation features via data querying mechanisms (e.g., [28]). Although these contributions show how personalisation may be experienced, it can be argued that they merely exploit DBMS functionality rather than accounting for how hyperlink-based interaction itself can be tailored in a principled manner.

Since we have formalised (in [41]) a complete space of possibilities for personalisation actions, the adoption of our model enables a system to support all the personalisation actions described in [6] and many of those in [7]. Implemented systems such as Adaptive HyperMan [36], ELM-ART [9], Hypadapter [29] and AHA [2] can all be represented using our model.

In its aims, our model subsumes the work on the adaptive hypermedia reference model AHAM [19, 2]. However, the explicit aim of the AHAM is to represent techniques that have been implemented and hence serve as a primary reference for comparative studies. In contrast to AHAM, the work reported here aims to induce a set of personalisation actions from a formal definition of a core of hyperlink functionality. In this way, we provide a methodological approach to the area that goes beyond the effort embodied in the AHAM.

The work presented herein concurs with that in [30] in which the Munich reference model (MRM) for adaptive hypermedia applications is introduced. However, both the AHAM and MRM models are based upon the notion that hypermedia applications are closed in nature and can be described using the framework of the Dexter Model [38].

The framework for personalisation of hyperdocuments introduced by this paper can be contrasted with the MRM proposed in [30] in that both seek to model personalisable hyperlink interaction. However, whilst the MRM is a reference model, that characterises the architecture required for hyperlink-based personalisation, the model

that underpins this research [41] characterises a set of formally defined personalisation actions. Furthermore, whilst the model described in the MRM views hyperlink-based systems as comprising user interface and database components, our work views these components as outside the scope of hyperlink-based personalisation. Our view that the design aspects of documents contained within E-books are the central ones to their usability is shared by [32, 49] and [35]. However, we aim to use personalisation as a value-adding strategy rather than, for example, visual clues [33].

Several customisable hypermedia systems have been developed as commercial products. Representative examples of which are Microcosm and its Web version, Webcosm [17, 18]. These products are concerned with the integration of hypermedia, database, and information retrieval technology in order to tackle the problems involved in providing access to multimedia repositories and providing protocols for the interoperability of hypermedia systems.

Although these systems work characterise the functionality which hypermedia systems can provide to hyperlink-aware and -unaware applications (i.e., applications that have no notion of linking), in contrast, the work reported here views such a characterisation, at a suitable level of abstraction, to be an essential starting point from which to induce a set of comprehensive P&A actions.

7 Contributions and Conclusions

In conclusion, it is clear that current E-book technology has some way to go before it can provide the richness of interaction that owners of paper-based books benefit from. However, personalisable hyperdocuments, accessible via E-books, can be designed to possess characteristics that is some ways supersede those of paper-based books.

The challenge, which the research [41] underlying this paper aims to meet, is how to model, at a suitable level of abstraction, the space of possibilities for personalisation actions that could be made available to readers of hyperdocuments.

Through personalisation of the interaction process that takes place between the reader and the E-book itself, we have shown that it is possible to perform actions that enable users to come closer to satisfying their individual information retrieval needs.

In Section 5 we have exemplified how our framework is a step towards bringing together the complementary benefits, highlighted in Section 3, in paper materials owned by a student and hyperdocuments that the student can only use but not personalise. The framework constitutes, therefore, a contribution to the goal of making E-books better able to address users' information needs on an individualised basis, as advocated, e.g., in [1, 5].

References

[1] D. Benyon, D. Stone, and M. Woodroffe. Experience with Developing Multimedia Courseware for the World Wide Web: The Need for Better Tools and Clear Pedagogy. *International Journal of Human-Computer Studies*, 47(1):197–218, 1997.

[2] P. D. Bra, A. Aerts, D. Smits, and N. Stash. Aha! meets aham. In P. D. Bra, P. Brusilovsky, and R. Conejo, editors, *Aptive hypermedia and Web based systems: Second International Conference, AH 2002*, LNCS, pages 213–222, Malaga, Spain, May 29 - 31 2002. Springer.

[3] P. D. Bra, P. Brusilovsky, and R. Conejo, editors. *Aptive hypermedia and Web based systems: Second International Conference, AH 2002*, volume 2347 of *LNCS*, Malaga, Spain, May 29 - 31 2002. Springer.

[4] T. Bray, J. Paoli, and C. M. Sperberg-McQueen. Extensible Markup Language (XML) 1.0. Technical report, W3C, http://www.w3.org/TR/REC-xml, February 1998.

[5] P. Brusilovsky. Intelligent Tutoring Systems for the World-Wide Web. In R. Holzapfel, editor, *Proceedings of Third International WWW Conference*, pages 42–45, 1995.

[6] P. Brusilovsky. Methods and Techniques of Adaptive Hypermedia. *User Modeling and User-Adapted Interaction*, 6(2-3):87–129, 1996.

[7] P. Brusilovsky. Adaptive Hypermedia. *User Modeling and User-Adapted Interaction*, 11(1-2):87–110, 2001.

[8] P. Brusilovsky, A. Kobsa, and J. Vassileva, editors. *Adaptive Hypertext and Hypermedia*. Kluwer Academic Publishers, Dordrecht, February 1998. ISBN 0-7923-4843-5.

[9] P. Brusilovsky, E. Schwarz, and G. Weber. ELM-ART: An Intelligent Tutoring System on World Wide Web. *Lecture Notes in Computer Science*, 1086:261–269, 1996.

[10] P. Brusilovsky and J. Vassileva, editors. *Adaptive Hypertext and Hypermedia*. Kluwer Academic Publisher, feb 1998. ISBN 0-7923-4843-5.

[11] J. F. Buford. Evaluating HyTime: an examination and implementation experience. In ACM, editor, *Hypertext '96, Washington, DC, March 16–20, 1996: the Seventh ACM Conference on Hypertext: Proceedings*, pages 105–115, New York, NY, USA, 1996. ACM Press. ISBN 0-89791-778-2.

[12] V. Bush. As we may think. *The Atlantic Monthly*, pages 101–108, 1945.

[13] B. Campbell and J. M. Goodman. HAM: A general-purpose hypertext abstract machine. In *ACM Hypertext'87 Proceedings*, pages 21–32, 1987.
[14] L. A. Carr, D. W. Barron, H. C. Davis, and W. Hall. Why use HyTime? *Electronic Publishing Origination, Dissemination, and Design*, 7(3):163–178, Sept. 1994.
[15] J. Conklin. Hypertext: An Introduction and Survey. *IEEE Computer*, 20(9):17–41, 1987.
[16] Cytale. Producer of Cybook ebook hardware. http://www.cytale.com accessed 24 August 2002.
[17] H. Davis, W. Hall, I. Heath, G. Hill, and R. Wilkins. Microcosm: An open hypermedia environment for information in tegration. Technical report, Southampton University, 1992. Tech Report:CSTR 92-15.
[18] H. Davis, G. Hutchings, and W. Hall. Microcosm: A hypermedia platform for the delivery of learning materials. Technical report, Southampton University, 1993. CSTR 93-10.
[19] P. de Bra, G.-J. Houben, and H. Wu. AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. In *Proceedings of the 10th ACM conference on Hypertext and Hypermedia*, Darmstadt, Germany, February 21-25 1999. ACM.
[20] S. J. De Rose and D. G. Durand. *Making Hypermedia Work: A Users Guide to HyTime*. Kluwer, Boston/Dordrecht/London, 1994.
[21] N. M. Delisle and M. D. Schwartz. Collaborative writing with hypertext. *IEEE Transactions on Professional Communication*, 1(32):183–188, 1989.
[22] Gemstar eBook Group Limited. Rocket e-book pro. http://www.ebook-gemstar.com accessed 24 August 2002.
[23] R. Grauer. *Exploring Microsoft Internet Explorer 4.0*. Prentice Hall (a Pearson Education company), 1998. ISBN 0138616752.
[24] F. Halasz and M. Schwartz. The Dexter hypertext reference model. In *Proceedings of the Hypertext Standardization Workshop*, volume 500–178 of *NIST Special Publications*, pages 95–133, Gaithersburg, MD 20899, January 1990. National Institute of Standards and Technology.
[25] F. Halasz and M. Schwartz. The Dexter hypertext reference model. *Communications of the ACM*, 37(2):30–39, 1994.
[26] B. L. Harrison. E-books and the future of reading. *IEEE Computer Graphics and Applications*, 20(3):32–39, May/June 2000.
[27] A. Heimburger. Introduction to Standard Generalized Markup Language (SGML). *Microcomputers for Information Management*, 11(4):239–260, Dec. 1994.
[28] C. Hockemeyer, T. Held, and D. Albert. RATH — A Relational Adaptive Tutoring Hypertext WWW–Environment Based on Knowledge Space Theory. In C. Alvegård, editor, *Proc. CALISCE 98*, pages 417–423, 1998.
[29] H. Hohl, H.-D. Boecker, and R. Gunzenhaeuser. Hypadapter: An Adaptive Hypertext System for Exploratory Learning and Programming. *User Modelling and User-Adapted Interaction*, 6(2-3):131–156, July 1996.
[30] N. Koch and M. Wirsing. The Munich reference model for adaptive hypermedia applications. In P. D. Bra, P. Brusilovsky, and R. Conejo, editors, *Aptive hypermedia and Web based systems: Second International Conference, AH 2002*, LNCS, pages 213–222, Malaga, Spain, May 29 - 31 2002. Springer.
[31] M. Landoni, F. Crestani, and M. Melucci. The visual book and the hyper-textbook: Two electronic books one lesson? In *Proceedings of the RIAO 2000 Conference*, pages 247–265, Paris, France, April 2000.
[32] M. Landoni, R. Wilson, and F. Gibb. From the visual book to the web book: the importance of good design. In *Proceedings of the Fourth European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2000)*, pages 18–20, 2000.
[33] M. Landoni, R. Wilson, and F. Gibb. From the visual book to the WEB book: The importance of good design. *Lecture Notes in Computer Science*, 1923:305–314, 2000.
[34] M. Landoni, R. Wilson, and F. Gibb. Looking for guidelines for the production of electronic textbooks. *Online Information Review*, 25(3), 2001.
[35] C. C. Marshall, M. N. Price, G. Golovchinsky, and B. N. Schilit. Designing E-books for legal research. In *JCDL'01: Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries*, Digital Libraries for Education: Technology, Services, & User Studies, pages 41–48, 2001.
[36] N. Mathe and J. Chen. User-centered Indexing for Adaptive Information Access. *User Modeling and User-Adapted Interaction*, 6(2-3):225–261, July 1996.
[37] Microsoft. Ms reader. http://www.microsoft.com/reader/default.asp accessed 24 August 2002.
[38] J. Moline, D. Benigni, and J. Baronas, editors. *Proceedings of the Hypertext Standardization Workshop*, volume 500–178 of *NIST Special Publications*, Gaithersburg, MD 20899 USA, January 1990.
[39] S. Newcomb and V. Newcomb. Some background information about HyTime (hypermedia language). *Journal of the Institute of Image Electronics Engineers of Japan*, 21(5):459–467, Oct. 1992.
[40] S. R. Newcomb, N. A. Kipp, and V. T. Newcomb. The "HyTime": hypermedia/time-based document structuring language. *Communications of the ACM*, 34(11):67–83, Nov. 1991.
[41] J. Ohene-Djan. *A Formal Approach to Personalisable, Adaptive Hyperlink-Based Interaction*. PhD thesis, Department of Mathematical and Computing Sciences, Goldsmiths College, University of London, University of London, 2000. Available from http://homepages.gold.ac.uk/johene-djan/pa-hlbs.ps.zip.
[42] Palm Digital Media. Palm reader pro. http://www.peanutpress.com accessed 24 August 2002.
[43] L. Press. Personal computing: from P-books to E-books. *Communications of the ACM*, 43(5):17–17, May 2000.
[44] R. Rada. Hypertext writing and document reuse: the role of a semantic net. *Electronic Publishing*, 1990.
[45] R. Rada and W. Wang. Experiences with semantic net based hypermedia. *International Journal of Human-Computer Studies*, 43(3):419–439, 1995.
[46] G. Schneider and J. Perry. *Electronic Commerce*. Thomson Learning, 2000. ISBN 0-7600-1179-6.
[47] J. B. Smith, S. F. Weiss, and G. J. Ferguson. A hypertext writing environment and its cognitive basis. In *ACM Hypertext'87 Proceedings*, pages 195–214, 1987.
[48] The Open eBook Forum Publication Structure Working Group. Open ebook publication structure specification : Version 1.0.1. Technical report, The Open eBook Forum, 2001.
[49] R. Wilson. EBONI: Designing effective electronic textbooks. *Library Hi Tech News*, 19(4), May 2002.