ELECTRONIC PUBLISHING, VOL. 7(2), 89–111 (JUNE 1994)

The Individualized Electronic Newspaper: an example of an active publication

ANJA HAAKE, CHRISTOPH HÜSER AND KLAUS REICHENBERGER

Integrated Publication and Information Systems Institute (IPSI)
Gesellschaft für Mathematik und Datenverarbeitung (GMD)
Dolivostrasse 15, D-64293 Darmstadt, Germany

SUMMARY

During the last four years the PaVE department at GMD-IPSI experimented with the Individualized Electronic Newspaper, an active publication that is individualized and composed on demand for a reader, and then delivered electronically. The work concentrated on the user interface design for active electronic publications and, in particular, on the investigation of publishing system architectures supporting the preparation and production of active electronic publications. The paper introduces two alternative interfaces for an electronic publication showing the potential of the electronic medium for publication design. The main part of the paper presents our approach to making such publications possible: a combination of structured documents and knowledge-based techniques based on a sound publishing model. This approach guided the design of an integrated publication environment for the preparation and production of active documents.

KEY WORDS Interface design Structured documents Publishing models Publishing architectures

INTRODUCTION

The paper introduces the Individualized Electronic Newspaper (IEN), an example of an active publication. The IEN is part of the European RACE (Research and Development in Advanced Communications Technologies in Europe) project 1075 TELEPUBLISHING [33], which aims to investigate and illustrate the potential usefulness of broadband technology in Europe. The IEN is an experimental publication that is individualized and composed on demand for a reader, and then delivered electronically. The content of the IEN is automatically selected according to a profile of the reader's particular interests from a continually developing pool of "up to the minute" articles and features. To exploit fully the possibilities for electronic delivery, the reader can receive the IEN both as a printable and as a hypermedia product. Thus, the IEN comprises both paper and interactive presentation.

The difference between the IEN and other newspaper projects (see the section on differences from related work) is that the IEN is intended as a potential publication that is designed by a publisher with respect to a certain reader group or market. The IEN may thus be considered not just an active electronic document,[1] i.e., a document designed by an author with a particular purpose in mind whose presentation and content depend on computation, but rather an active electronic publication. In particular, both the IEN content and presentation are responding to aspects of the surrounding world's state. On the one

[1] Active documents can be intended for interactive and paper presentation. The former are often called interactive active documents. We prefer the term active electronic document or, for short, electronic document.

CCC 0894–3982/94/020089–23

Received 29 March 1993
Revised 2 September 1993

hand, the content of an individualized newspaper issue is composed on demand according to a profile of the reader's particular interest. On the other hand, since potential readers cannot be forced to buy equipment just to be able to subscribe to different active publications provided by different publishers, the IEN must be delivered to different target systems, depending on the printing or hypermedia reading software available at the reader's site.

In this paper, we will focus on the contribution of GMD-IPSI to the IEN project. The PaVE (Publication and Visualization Environment) department at GMD-IPSI is mainly interested in concepts and techniques needed to realize active electronic publications such as the Hypermedia IEN. This interest includes two major topics: interface design and system design for active electronic publications.

First, little is known about how electronic publications should look. Good design rules for electronic documents are still missing. The presentation of information communicates its function to the reader. Such presentation forms for communicating knowledge have been developed over hundreds of years for conventional publications (for example see [37] for newspaper layout rules), but still are missing for electronic documents. So first of all, the design and functionality provided by potential active documents have to be defined.

Second, based on a more precise idea how electronic publications look, the publishing process and the publishing technology have to be reviewed and then enhanced with respect to the requirements imposed by the preparation and production of such active publications.

Taking the IEN as an example, this paper discusses the two aspects of interface design and system design for active publications. We introduce two alternative interface designs of the Hypermedia IEN showing the potential of the electronic medium for document design. Then, we discuss requirements imposed on the publishing process and technology by this kind of active publication. Next, we introduce our approach to satisfy these requirements, the application of structured documents to electronic publishing. This leads to the definition of an extended publishing model and an architecture for an integrated publication environment (IPE) aimed at the production of active publications. We show how the facilities offered by an IPE are being applied to make the production of the IEN possible. Finally we discuss differences to related work and conclude on first experiences gained from our approach.

1 EXPLORING METAPHORS FOR ACTIVE DOCUMENT INTERFACES

We introduce two alternative designs of the Individualized Electronic Newspaper interface that show the potential of hypermedia and general options for the design of active document interfaces. Aiming at diminishing the known hypertext problems of disorientation and getting lost in hyperspace [10], we explored two different metaphors to guide the designs. These metaphors each imply a different presentation language that communicates the document's functionality to the reader. First, we show the exploration of the newspaper metaphor and then introduce an interface taking into account information presentation and processing mechanisms known from electronic mailtools.

Both interfaces have been implemented in HyperNeWS [38]. HyperNeWS is a stand-alone, extensible, object-oriented interface system based on the hypertext paradigm. HyperNeWS combines the NeWS System [16] and the idea of open access computing. Based on a client-server architecture, it provides, in addition to programming by direct manipulation, a high-level interface to a variety of languages, e.g., C, Lisp, and Prolog. HyperNeWS itself is programmed in PostScript and all internal as well as asynchronous or synchronous inter-process communication is done via message-passing between objects.

Following the hypertext paradigm, similar to HyperCard [15] HyperNeWS offers stacks, backgrounds, cards, textfields, canvases and buttons as predefined object classes. Each instance of an object class, called prototype, can be associated with individual property values describing its presentation on the screen and its potential activities. The presentation covers aspects such as font characteristics of text, the color of an object, or even the shape of an object. Since HyperNews is based on PostScript, an object may have an arbitrary shape. The potential activities of an object are described by a set of individual PostScript procedures. The execution of these procedures is triggered by either messages sent by other objects or events triggered by the user. The latter comprise the opening and closing of stacks and different types of mouse-clicks and mouse movements. If an object is modified in direct manipulation mode, the cutting, moving, copying, etc., of objects also may trigger the execution of a specified message.

With these characteristics, HyperNEWS supports active documents in the sense of EmbeddedButtons [7] and Tioga [36] or Interleaf [12] active documents, i.e., an active document has a program attached to it that allows the document to act on its environment. Within this meaning, the two IEN prototypes introduced below are active documents.

1.1 Exploring the newspaper metaphor

We designed the customized IEN as a fully integrated hypermedia product for access on a multimedia computer terminal or notebook. In addition to enhancing articles with high-quality images, animations, and video, the IEN increases the current newspaper functionality. It offers access to background material and to databases of classified advertisements, or enables the publisher to provide extended news related services.

Figure 1 shows an IEN front page composed for a reader with interest in current affairs and a special interest in science. While the front page is made to look much like a traditional newspaper, its functionality is enhanced to provide access to the content of a hypermedia newspaper: for example, its different multimedia contents (in the right lower corner, left of the contents bar, the reader can activate a news video clip), the background information, and the extended services.

According to his/her mood and information needs the reader can choose to flip pages by clicking on the arrows. He/she may reach sections of his/her interest directly or make use of the lexicon look up by pressing the bullet buttons in the contents bar.

The article on “Fusion Fraud” (containing new developments on the issue of cold fusion) provides the reader with different background material: clicking the buttons here will reveal a chronicle of events, e.g., a series of published articles dealing with this topic (Background), a transcript of an interview, which may eventually be shown as a video (Interview), a collection of bibliographic references (Literature), and a number of controversial contributions acquired directly by electronic mail (Netnews).

“A Cheap and Easy Second Shot” is an example of a scrollable text with embedded buttons. Clicking on a sensitized phrase will bring up a lexicon entry containing its translation (e.g., into German) and explaining its meaning. Actually, ipsiLex [13] has been integrated into the IEN as a lexicon service. The sensitized string in the article has attached a procedure[2] that queries a database (see the section about “The structured document

[2] To realize the degree of activity we required for the IEN, we extended the HyperNeWS class hierarchy by a so-called "HyperTextField". In a "HyperTextField" each string may be marked up and can be associated with presentation properties (e.g., color) and a procedure that is executed when the string is double-clicked.

Figure 1. Example of a front page of the Hypermedia IEN (newspaper metaphor)

base") containing the most recent ipsiLex dictionary entries and triggers the generation of a rapid-lookup of the entry. Figure 2 shows the rapid-lookup of an ipsiLex lexicon entry.

The figures show the application of document metaphors to the presentation of electronic documents in order to avoid disorientation and getting lost in hyperspace. Rather than defining the appearance of hyperdocuments on the basis of the system's technical capabilities, we exploit the technical features of the hypertext target system to realize the best presentation form for each kind of electronic publication. For example, employing the newspaper metaphor means that newspaper articles are not mapped straightforwardly onto single HyperNeWS cards. Rather, a set of articles is arranged on a card representing a newspaper page. In contrast, background information pops up (e.g., the lexicon) and is presented according to the needs of the kind of publication.

The evaluation of the interface has shown that the idea of using different presentations and layouts for different publications diminished disorientation [19]. The readers liked to be aware of the context switch when invoking the additional information and services. At any time the respective presentation elucidates the kind of publication, i.e., it was obvious that the window on the screen shows a newspaper, a lexicon, or some background information.

However, it turned out that the electronic medium creates requirements for functionality to process the information contained in an electronic document. Users want to know what they have already read and what remains to be done. They want to annotate the content, send interesting content around to people that should be aware of the news, or want to cut and paste the content into their working environment. To fulfill these requirements,

Figure 2. Rapid-lookup of an ipsiLex lexicon entry (on term "bookmark") popping up

we designed a completely new interface adopting information processing concepts known from electronic mailtools.

1.2 Exploring the multimedia mailtool metaphor

When entering the IEN interface, the reader is confronted with the front page shown in Figure 3. The purpose of the front page is to display the content and structure of the newspaper issue. Here, all articles contained in the issue are represented with their headline and are grouped according to the different sections they belong to. Either a reader may select a section to begin with by clicking on the section icon or if one headline calls his attention the reader can open the respective section with the selected article "in focus" (see Figure 4) by clicking on this headline.

A section will cover the whole front page (see Figure 4) except the bar at the bottom of the screen, which is always visible regardless of the section read at the moment. This bar contains global functions like going from one section to another, getting online help and messages of the IEN for the user, and returning to the front page. On the section level, the screen is divided into two main regions of approximately equal size. One region, the focus, contains only one article at a time. This article is displayed in black on a light background, the headline in a larger font. Subtitle and byline (date and author information) are displayed separated from the text. When the text does not fit entirely in the focus, scrolling buttons are displayed.

Additional buttons are displayed if there is additional information available. The shape

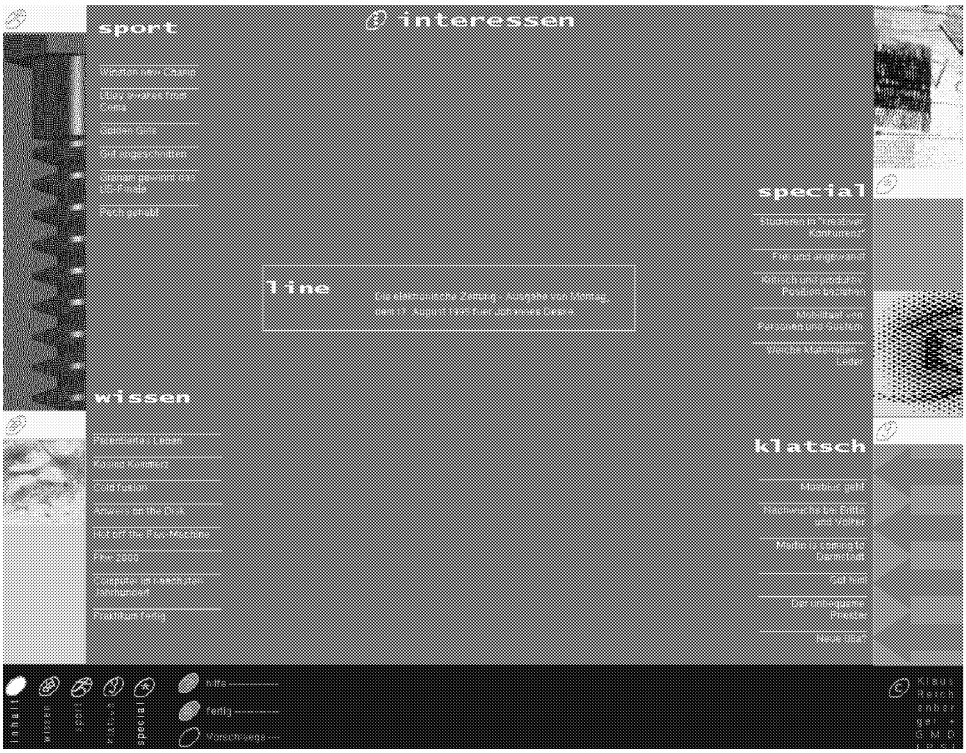

Figure 3. Example of a front page of the Hypermedia IEN (mailtool metaphor)

of the button depends on the category of the information (background information, original material, discussion) available. When the reader operates one of these buttons, the additional information is displayed at the cost of the main text of the article. The additional information can be text, picture or video. Thus the reader can consume all information clinging to the article in the focus, starting from the headlines and the main text.

The other main region of the screen, the pool, provides information for navigating the content of a section. The pool contains two lists of the articles in that section. The headline list shows the headlines of all articles of the section. It provides an overview of the content similar to the content lists of the front page. The abstract list shows head, subhead and abstract of each article, as far as they fit on the page.

At any time the reader has the following possibilities to work through the section. He can save or throw away the article that is in focus by clicking on the “archive” or the “wastebasket” button or by dragging the article towards the respective button. This will cause the topmost article to move to the focus side of the screen and the abstract list to be updated. So the abstract list only contains unread articles. The reader can also drag any article from the pool into the archive or the wastebasket.

Another possibility to change the article in focus is to drag one of the articles from the pool to the focus. The recent article is then automatically thrown away and the articles that would have been in turn before the dragged one are marked as skipped and will be appended to the end of the abstract list.

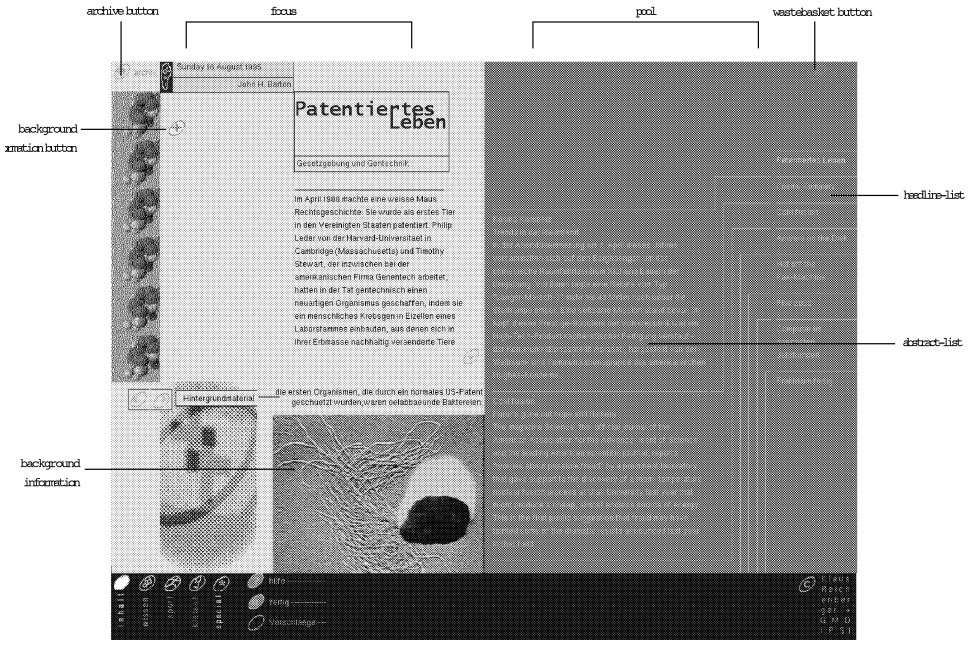

Figure 4. Example of a section page of the Hypermedia IEN (mailtool metaphor)

This implies that any article has one of five different states. It can be the current article, it can be archived or thrown away, it can be skipped and it can be among those that are still to be read. These different states are reflected in the headline list by different colors.

The abstract list shows only articles still to be read. That way the abstract list literally gets consumed during reading. When the reader has decided for all articles whether he has further interest or not—either by archiving them or throwing them away, substituting or bypassing them—the abstract list remains empty. However, new articles may be incorporated. Whenever the headline list is updated and there is still space to show further headlines, the textfield representing the headline list may execute a procedure that looks for further articles available in the environment.

The reader can also extend the focus to a full-screen view of the current article, which leaves only the headline list and reformats the article (see Figure 5). In this view, operations to comment an article and to mail these comments are provided.

The basic principle behind features of this interface such as, for instance, the drag and drop interaction or the color coding of the headline list, is that the layout should reflect the user's actions and the status of the articles, instead of just being a fixed presentation. Technically, this is achieved by triggering appropriate procedures updating presentation and content of the electronic document when articles are clicked and moved.

2 REQUIREMENTS ON PUBLISHING TECHNOLOGY

The electronic newspaper issues described in the previous section are complex, well prepared active electronic documents. The different kinds of constituents of the issue, i.e., articles, visual information, and the various kinds of documents needed for the

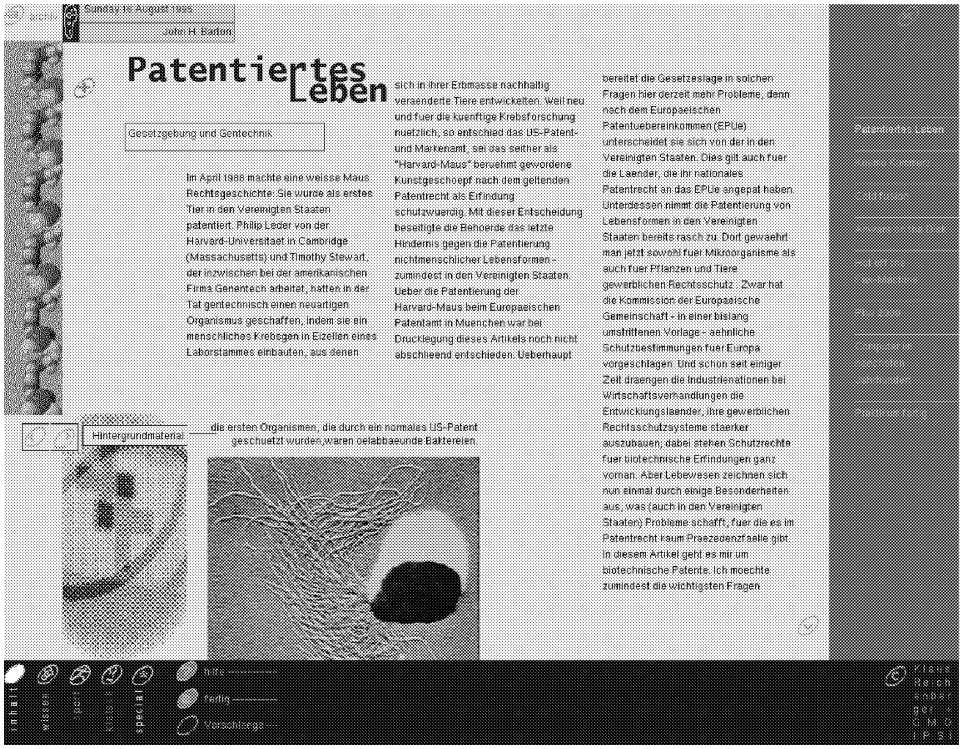

Figure 5. Example of a full-screen article (in German) of the Hypermedia IEN (mailtool metaphor)

news-related services (background information, interviews, literature, net news, lexicon service etc.), the relationships between them, and the presentation of both to the user together make up the value of these electronic publications. The automatic composing and delivery of such individualized publications from a continually developing pool of information imposes new requirements on the overall publishing process that raise new challenges for publishing technology.

The individualization of content has to be performed automatically (Req. 1) and requires that the interests of the reader be mapped successfully onto the information contained in the pool. The resulting newspaper issue should meet the expectations of the reader as well as possible.

The delivery onto various systems, i.e., the individualization of the presentation of electronic publications, imposes the problem of automatic presentation of hyperdocuments (Req. 2). Previewing of hundreds of individual issues for different delivery systems is not possible. The electronic publications have to be produced without manual corrections.

To compose such complex publications on demand from continually changing content, the content has to carry additional information needed for the production and delivery process. Such information comprises explicit modelling of relationships between the different kinds of content in general, the classification of content for the selection of content, and presentation information for the subsequent delivery to different target reading systems. At any time, the information contained in the pool of articles has to be consistent with the

expectations of the production and delivery tools. The information source has to maintain both the constraints and rules for all kinds of entities involved in the production of an individualized newspaper issue, and the constraints and rules on meaningful links between those different entities (Req. 3).

The creation of complex publications is a distributed process involving the cooperation of various experts. In publishing, cooperation is based on the exchange of documents. Usually the experts use specialized tools supporting their specific tasks. In general, these tools rely on different data formats and the problem of document exchange is hard to solve. As for the IEN, information provided by external sources such as news agencies or information used to realize the innovative news-related services (e.g., net news information) as well as content prepared by several journalists and editors using different systems (PC at home, lap-top for live reports) have to be moved into the publisher's editorial system and format. To make the document exchange possible during the publishing process, a system- and device-independent data representation format for electronic documents (Req. 4) is required.

It is the task of the editorial staff to choose from the various information sources mentioned above and to turn this content into well-prepared constituents carrying all information needed for the production and delivery tools. To do so, in addition to the tools for the editing of content (text, audio, video), the editorial office needs tools for editing and checking the document structure, for classifying content and creating meaningful links between the different newspaper constituents. To control the effect of the added classification and links, and to check the amount and quality of content actually contained in the pool of articles and features, the editors need adequate previewers. In general, tools supporting the creation of consistent electronic documents containing all information needed by the production and delivery tools (Req. 5) have to be developed.

In addition, the possibility to update the pool of articles continuously requires adequate version support for hyperdocuments (Req. 6). On the one hand, editors want to keep track of their incremental changes. On the other hand, an individualized newspaper makes it possible to prepare the same content at different levels of detail and abstraction in separate newspaper contributions for various groups of readers.

Considering the required tools for document creation (Req. 5) and the tools performing the automatic individualization and presentation (Req. 1 & 2) possibilities for document exchange (Req. 4) are not sufficient to support the publishing production process. All these tools supporting cooperating experts in the publishing process or performing the document delivery without human assistance share the same documents and require general functionality to manipulate, retrieve, and navigate these documents. Therefore, shared databases for electronic documents including meaningful access functionality are also required (Req. 7).

3 THE DESIGN OF AN INTEGRATED PUBLICATION ENVIRONMENT FOR THE PRODUCTION OF ACTIVE PUBLICATIONS

A closer look at the requirements suggests that careful design and planning will gain more and more importance for the production of active publications compared to the production of non-active publications. In general, since the content, structure and presentation of active documents react to aspects of the surroundings world's state, the possible and meaningful reactions have to be planned in advance. Our approach is to base the active document preparation and production on a sound model of the publishing process (see Figure 6).

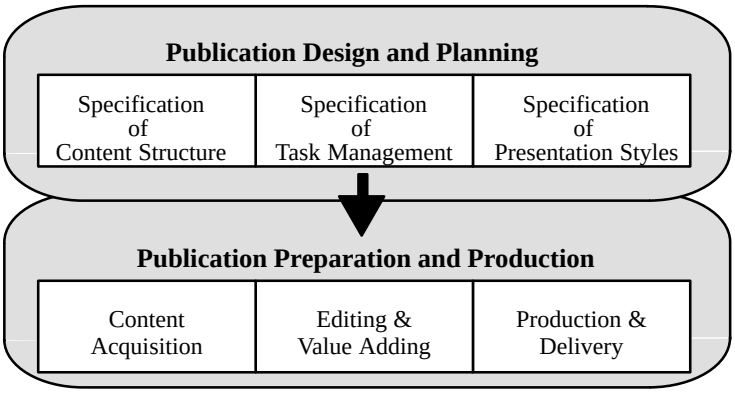

Figure 6. General publishing model for the production of active documents

The model combines the concept of structured documents with knowledge-based techniques. The idea of the publishing model is to separate the logical document structure from all kinds of semantics, as they appear for printed and electronically delivered documents. The semantics beyond the logical structure are covered by declarative rules, so-called style definitions, that present knowledge about presentation or other task management in general and can be associated with the logical document structure. Thus, explicit structure and declarative styles defined during publication design and planning will guide users and/or the decisions of knowledge-based production tools during publication preparation and production.

Our implementation of this model is based on SGML, the Standard Generalized Markup Language [1]. During the publishing process documents to be produced and published are marked up to identify the functional components of the publications, for example chapters, sections, headline, footnotes. These components serve as reference points during discussions and constitute the basic units for layout design of paper documents. The basic idea of SGML is to exploit the markup of hierarchically structured documents for the exchange of documents between different systems by standardizing the notation of markup. SGML defines a syntax to describe both document type definitions (DTDs) defining the legal structure of a certain kind of document and the concrete markup of the documents themselves. Therefore, SGML can be employed to express the constraints and rules of entities involved in the publishing process in a system- and device-independent representation format for document exchange and thus serves both requirements 3 and 4.

Consequently, our approach covers the logical document structure by SGML document type definitions. Concrete electronic documents are represented by SGML document instances conforming to a certain document type definition. The style definitions are currently done in specially designed rule-based languages. As far as possible, we adhere to adequate processing information standards (see the section on “The Hypermedia Presentation Composer”).

To realize shared databases for electronic documents including meaningful access functionality (Req. 7), we offer database management support for SGML-conformant documents. The analysis of the production tools required for the IEN has shown that access functionality has to be provided at the level of structured documents. It takes a considerable amount of work to define (and years to standardize) DTDs. The application tools should be

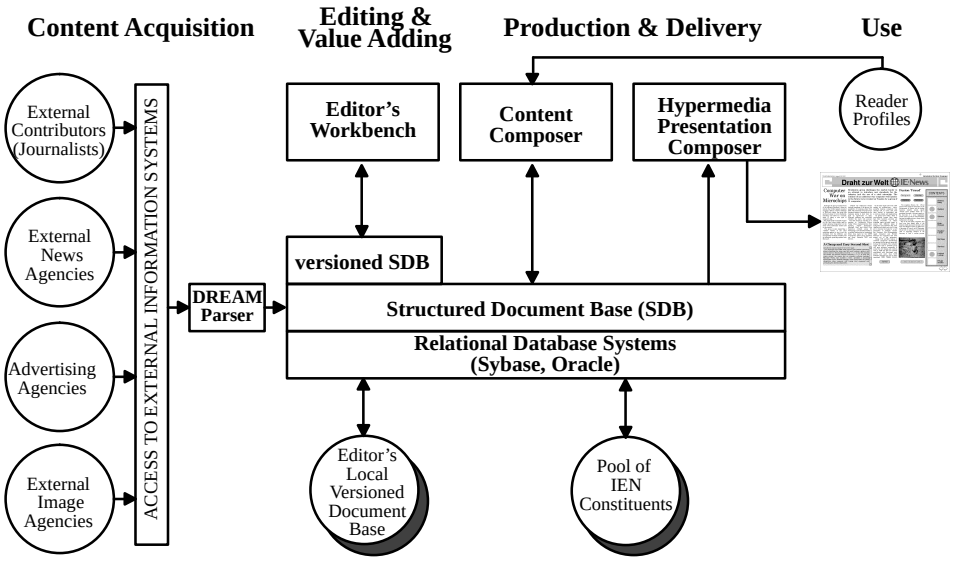

Figure 7. Publishing process oriented architecture of an integrated publication environment

programmed on the basis of this knowledge and should not be forced to deal with complex (relational) database schemas. Therefore, we built the Structured Document Base (SDB) [20], a database application offering SGML-conformant storage, manipulation, navigation and querying of documents (see Figure 7). Requirement 6, version support for documents, implies also the employment of general concepts that can be extracted from the application tools. We developed the versioned Structured Document Base (vSDB) [40], an extension of the SDB, offering version support for structured documents.

To support the creation of consistent documents (Req. 5) and the subsequent automatic individualization of content (Req. 1) as well as the automatic presentation of hyperdocuments (Req. 2), we have designed three task-specific tools, the Editor's Workbench, the Content Composer and the Hypermedia Presentation Composer, respectively. The three tools mentioned above incorporate knowledge about the specific document processing tasks they have been designed for. They are built on top of the SDB to share documents.

In the following we explain the integrated publication environment (IPE) for the publishing of the IEN shown in Figure 7. Circles indicate data repositories relevant in the production of the IEN; boxes represent publishing tools and components manipulating data. Arrows show the flow of data starting from content acquisition finally leading to the production and delivery of the electronic IEN to the user. We discuss the components shown in the architecture, starting with the central components, the SDB and the vSDB. Then, following the production process of the IEN, we describe the task-specific tools, explain their relationships to the SDB, and illustrate how they make the IEN possible.

3.1 The Structured Document Base

The Structured Document Base (SDB) [20] is a database application offering persistent storage, manipulation, navigation and querying of documents modeled in SGML. The SDB

implements the basic SGML standard and offers extensions conforming to the standard SGML syntax to cope with hypertext links, i.e., machine-supported cross-references, that are inherent in electronic documents.

The SDB manages SGML-conformant documents in separate pools. The pool concept corresponds to the database concept of database management systems and allows applications to realize distinct data repositories.

The SDB allows the user/application programmer to store any SGML-conformant document into any pool. To do so, the DTD contained in an external standard SGML file has to be declared to the respective pool. After this, any SGML document belonging to this type stored in an external file can be imported into the pool through an import operation.

Any document stored in the pool can be manipulated by the application programmer interface of the SDB. The SDB considers each document as a document instance and knows about every document element and its attributes. This is attained by considering DTDs as object-oriented schema definitions. Each document instance and each element are considered objects, and the elements' attributes are considered the properties of the objects. Thus, the SDB offers object-oriented operations to create and manipulate whole documents (e.g., rename), to create and manipulate new document elements, to navigate along the document structure (up, down, succ, pred), and to create and manipulate the elements' attributes. For all update operations, the SDB assures conformity of the document with the associated DTD. At any time, each document can be exported in standard SGML format.

Moreover, the SDB offers a path-oriented query language that allows access to each document and each element on the basis of the document structure and the associated attributes. The query language has been designed following the Document Style Semantics and Specification Language (DSSSL) [2], a companion standard of SGML specifying document processing. In the SDB, queries can be posed against a whole pool, can be restricted to documents of a certain type, or can be restricted to a single document. In any case, the SDB returns a compound element, or a so-called query element, that contains all result objects as sub-elements. In this sense, the query language of the SDB is closed. The query element can be navigated, manipulated and interrogated like any other structured document maintained by the SDB.

It is a basic design principle of the SDB to handle all information necessary for document processing as structured documents. In particular, DTDs themselves are documents obeying certain rules and thus conform to a DTD. This DTD describing DTDs, the so-called super DTD, is the central concept of the realization of the SDB. So, DTDs are declared to pools of the SDB by an import operation and may be updated, navigated and queried like any other document. In addition, all information needed to implement the SDB functionality, as for example the query element, are considered structured documents and conform to specially designed DTDs. These DTDs are automatically contained in any of the SDB pools.

A difficulty in applying SGML to electronic document production is that SGML lacks an explicit notion of hypertext links. A common approach of SGML applications to overcome this problem is to define link types as explicit elements that carry two attributes of type IDREF to refer to the source and destination object of the link. Additional attributes may be defined depending on the type of link, for example an attribute containing the link label, or attributes modeling link anchors (starting point and offset). In this way, the Text Encoding Initiative [3] has defined a general reference link. Also the Grif editor [32] follows a similar approach to integrate nonlinear references into a hierarchical structured document model. Such a kind of link representation is inherent in the SDB. Moreover, the SDB generates

globally unique object identifiers for each element (Internet address of the element creator[3] plus name of the pool plus unique number). Since the SDB offers full access and query functionality with respect to these identifiers, they may be used by the applications as consistent ID and IDREF attribute values.

Recently, the HyTime [31] standard extending SGML towards hypermedia applications has defined five standardized link types including their processing semantics. Aiming at maximum support of hypertext applications, we plan to implement these links and their semantics in the SDB. In addition, HyTime proposes to implement a hypermedia document model by associating meta-interpretations to SGML element declarations. To support these HyTime facilities concepts proposed by object-oriented database management systems should be exploited for structured document database applications. For example VODAK [30] offers a homogeneous integration of the concept of metaclasses with a separation of types and classes.

Furthermore, reference support of shared objects is implemented in the SDB. The SDB interprets the reserved SGML word ANY (expressing that the sub-element of an element may be any type of element declared in the DTD of the document) in the scope of whole pools. Consequently, the sub-element may be any element declared in any DTD in the pool.

The SDB has been implemented in C++ on top of the Sybase relational database management system. The SDB only depends on standard SQL. This modular internal architecture made it possible also to move the SDB to Oracle. The SDB makes use of the Amsterdam SGML Parser [39] to realize the import functionality. Providing this functionality, SDB meets the requirements 3, 4, and 7. It supports requirements 1, 2 and 5 by offering consistent update, navigation and retrieval operations for electronic documents.

The SDB has in particular been applied to implement the Pool of IEN Constituents that stores the information contained in and related to a daily newspaper as a set of structured documents. It holds the current articles, pictures, and background information for topics discussed in the newspaper, and reference documents such as dictionaries and encyclopedias.

Considering the actual IEN demonstrator, GMD-IPSI has developed DTDs for different types of articles (e.g., report, interview, commentary), visual information (including their classification attributes), as well as for an overall newspaper issue [20]. Regarding the Hypermedia IEN, the links are also part of the documents and different types of links (e.g., reference link, explanation link) have been defined. Moreover, ipsiLex [13], a hypermedia lexicon implemented by PaVE on the basis of the SDB is connected as a lexicon service to newspaper information.

3.2 The versioned Structured Document Base

The versioned Structured Document Base (vSDB) [40] has applied concepts of the version model introduced in [41] to structured documents. In this model all objects are marked with a time and author stamp and can be equipped with application-defined attributes for version identification. Moreover, the model distinguishes single-state objects (snobs) representing non-versioned document objects from multi-state objects (mobs) representing

[3] To enable the exchange of documents between different installations of the SDB in a world-wide network, we decided to use this identification of users for the RACE TELEPUBLISHING project. Any other unique identification of the user would also fulfill this purpose. If the SDB is used without the intention to use several installations of the SDB in parallel, the identification can even be simpler.

versioned document objects and introduces the descendancy relationship to model the reuse of information across document boundaries. These concepts have been successfully implemented on top of the SDB. Since the model distinguishes snobs and mobs and defines a smooth transition between those concepts, the vSDB can transparently be used for non-versioned applications. In addition, the vSDB comprises an import and export facility for versioned structured documents on the basis of a specially designed DTD. As for the IEN, the vSDB has been used to implement the Editor's Local Versioned Document Base containing images and structured documents the editor is working on (Req. 6).

3.3 The Dream Parser

The content acquisition implies the use and reuse of external material. IEN editors have access to a number of external sources that deliver electronic contents like external news services, image and advertising agencies, or information systems containing data related to the subject a journalists wants to report on. Information not conforming to the format of the publisher's editorial system has to be converted and restructured according to the requirements of its publishing environment.

In the IEN prototype, GMD-IPSI implemented the content acquisition of net news information. The Dream (document structure recognition and markup) Parser, a tool to automatically mark up documents that contain only either implicit or incomplete coding by executing specified, rule-based transformations, has been used to restructure net news articles into SGML documents conforming to the IEN newspaper article DTD. Using the import facility of the SDB, these articles can directly be moved into the Editor's Local Versioned Document Base and are thus available for editing and value adding.

3.4 The Editor's Workbench

Content authoring and adding value to electronic publications demands qualitatively enhanced support during editing and value adding. The Editor's Workbench, developed by other partners of the TELEPUBLISHING project [19], supports the different tasks to be performed for the creation of the IEN constituents (Req. 5) including text and image editing, translation for a multilingual edition, and managing versions of articles. It also supports linking parts of the newspaper by hypertext links, classifying articles by specifying article descriptions for retrieval and presentation purposes, and previewing the visual effects of the specifications and decisions.

The Editor's Workbench is linked to the Editor's Local Versioned Document Base containing images and structured documents the editor is working on, to external information systems for content acquisition, and to the Pool of IEN Constituents maintaining published material.

The central component of the actual Editor's Workbench being implemented for the IEN prototype is the versioned Structure Editor that supports the creation of SGML-conformant hyperdocuments by controlling the correctness of the edited document with respect to its DTD. Each document and document element can be maintained in several versions. Using the versioned Structure Editor link, classification, and presentation information can be edited by manipulating the corresponding elements and attributes. For linking or the integration of complex hypermedia documents into an electronic publication, it would be desirable to integrate functionality provided by hypermedia authoring and editing systems

such as the SEPIA hypermedia authoring system [35] or the TEDI hypermedia editor [29] into an editor's workplace.

A subcomponent of the Editor's Workbench is an Editor Browser which allows one to navigate and check the content of all data repositories, i.e., the Editor's Local Versioned Document Base and the Pool of IEN Constituents. The Article Previewer shows the IEN constituents as they would appear in a newspaper issue and thus visualizes the effects of the performed edits.

All these components of the Editor's Workbench make use of the retrieval and navigation facilities offered by the SDB. The versioned Structure Editor in addition profits from the whole functionality offered by the vSDB.

The material ready for publication is stored in the Pool of IEN Constituents. This information is used by the production tools during production and delivery. The Content Composer uses the Pool to select the content of the individual newspaper issue according to the reader's profile. Depending on the choice of media either a Print Layout Composer delivers a printable newspaper issue following the publisher's layout style, or a Hypermedia Presentation Composer generates a Hypermedia IEN issue for electronic end-systems. In the following, we discuss the tools relevant for the electronic IEN, the Content Composer and the Hypermedia Presentation Composer.

3.5 Content Composer

The individualization of the content of active publications is a central question for the IEN. In the profile, the reader may specify the degree of his/her interest in certain sections of the newspaper. The Content Composer (Req. 1), implemented by another partner of the TELEPUBLISHING project [19], maps this information against the section structure, keywords and ratings attached to elements of newspaper articles stored in the SDB. The result of this mapping will be assembled to an individualized newspaper issue, which is stored as another structured document in the SDB.

The implementation of the Content Composer is technically based on the functionality provided by the SDB. The keywords are part of the DTDs for IEN constituents and can therefore directly be attached to the concrete IEN constituents. So, the relevant newspaper content can be accessed directly via the query interface. The SDB function interface is used to build up the assembled newspaper issues according to the particular newspaper DTD.

3.6 The Hypermedia Presentation Composer

One key problem imposed by the IEN is the automatic presentation of documents. Since designing and previewing of hundreds of individual issues is not possible, the issues have to be produced without manual corrections.

As shown in the section "Exploring metaphors for active document interfaces", our approach to diminishing the problems of disorientation and getting lost in interactive documents is to apply document metaphors in a consistent manner. Nevertheless, these experimental designs are only a starting point and good design rules for electronic documents are still missing.

Moreover, it must be possible to present the hyperdocuments on various hypertext systems, depending on the reading software available at the reader's site. Therefore, the Hypermedia Presentation Composer is intended as a vehicle to experiment with the

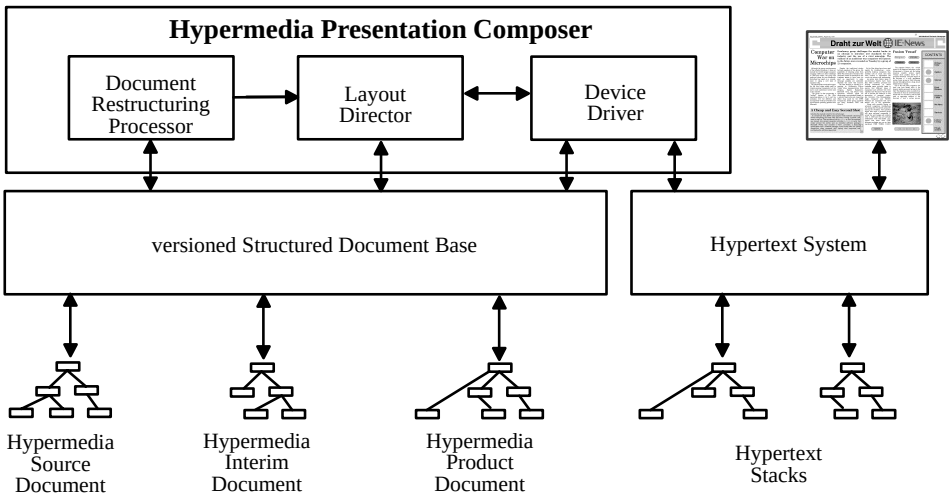

Figure 8. Overall architecture of the Hypermedia Presentation Composer

presentation of electronic documents. It must be possible to compose an issue for different hypermedia systems, or to experiment with different presentations of an issue in a single system.

In order to be an experimentation tool for both testing of different presentations of electronic documents and delivery to different end-systems, the Hypermedia Presentation Composer is designed as a knowledge-based system following the general publishing model. Figure 8 shows the overall architecture of the Hypermedia Presentation Composer. The overall composing process consists of three tasks: restructuring, publication assembling and interaction generation. These tasks are performed by three components that interact and communicate by sharing structured documents stored in the SDB as SGML-conformant documents. An overall coordinator process of the Hypermedia Presentation Composer controls and guides these three components.

According to a set of restructuring rules [22] the syntax of which adheres to DSSSL [2], first the Document Restructuring Processor transforms the set of documents to be delivered, the so-called Hypermedia Source Documents, into the so-called Hypermedia Interim Document, which is used for publication assembling and interaction generation. The Hypermedia Interim Document describes the arrangement of document content in a general document design by so-called controller elements. Not all of the content has to be used for a certain publication and the presentation of the content does not have to follow the structure of the information units. These decisions, which are independent of the chosen presentation style, are taken by the Document Restructuring Processor.

Then, following a selected presentation style, the Layout Director together with the Device Driver perform the publication assembly and interaction generation by mapping the Hypermedia Interim Document into the Hypermedia Product Document, which describes the final publication. In doing so, the Device Driver concurrently controls the generation of the concrete document to be delivered in the target hypertext system, for example as hypertext stacks in HyperNeWS or HyperCard.

During publication assembly, the Layout Director generates so-called layout controller objects for each layout controller element of the general design described in the Hypermedia Interim Document. According to the layout directives stated in the presentation style the layout controller objects driven by the Layout Director will generate abstract device objects describing the document to be rendered as a set of basic layout entities, such as laid out text strings, or buttons. These constituents are maintained in the Hypermedia Product Document. In order to define the actual size of the objects to be rendered, the Layout Director requests the Device Driver to compute the actual values. The Device Driver associates concrete layout objects of the target system, such as concrete HyperNeWS cards or buttons, to every abstract device object and computes the needed information by controlling and consulting the hypertext target system. So in parallel to the abstract layout generation performed by the Layout Director, the concrete device objects such as stacks, cards or buttons—and thus the final active document in the hypertext target system—are produced.

The interaction generation can be compared to generating a table of contents as performed by document formatting systems. If the publication assembly is finished and all device objects are generated and known, the Layout Director instructs the Device Driver to generate scripts for buttons that realize the switching of cards or call other publications. As for the IEN, the flipping of pages, the functionality of the contents bar, buttons referring to related articles, background information or extended services like the lexicon service will be generated by defining the corresponding HyperNeWS procedures for the respective concrete device objects. Note, general update functionality, such as the recoloring of headlines and the movement of articles in the mailtool-metaphor IEN, are activities already defined in the type of the concrete layout object associated with the corresponding abstract device object. A detailed description of the Hypermedia Presentation Composer can be found in [21].

4 DIFFERENCES FROM RELATED WORK

In this paper we have presented the IEN, an active publication that induces new requirements on publishing technology. The requirements are direct consequences of the active publication our project designed.

The approach to the preparation and production of an electronic document that is most similar to the IEN is the Electronic Magazine developed at Bellcore [11]. As with the IEN, the production of the Electronic Magazine is based on SGML. Whereas the IEN focuses on hypermedia publications, the Electronic Magazine aims at the integration of multi-media data into electronic documents. Other newspaper projects, such as NewsPeek [28] or Newspace [5] of the MIT and the Digital News System of the Apple/CNN project at EDUCOM [17] focus on delivering the issues to a single system. Thus, delivery on several systems as for the IEN is no requirement.

NewsPeek is not intended as an electronic publication. The focus rather is on an interactive database interface. The most recent project ‘Newspace’ in addition attempts to take advantage of advances in display technology [27]. NewsPeek scans several sources and editing, which is restricted to selecting the material, is done by software agents. Thus, NewsPeek focuses on information retrieval whereas the IEN is designed as a complete publishing product. The IEN has to provide tools for human editors. To build these tools database support including versioning concepts is required.

The individualization of content of publications is a central question in the IEN. We think that our first approach to explicitly classify information is not sufficient. In our opinion, it is necessary to extend information retrieval concepts towards information intake. Information intake refers to information that impinges on users, e.g., users intake electronic mail and news. In information retrieval the user is actively searching, whereas a user reacts to information intake.

The NewsPeek [28] system has incorporated several concepts for content individualization based on content analysis and complex keyword filters as well as concepts for changing the newspaper content because of reading decisions by the reader. Baclace [4] has proposed a personal information intake filtering system exploring a hybrid learning algorithm. But none of these approaches takes into account the explicit structure information of structured documents, as for example Frei and Stieger proposed in their work [14]. We think that new approaches to information intake combining all these concepts should be developed for the individualization of content of active publications. The data representation, access and manipulation facilities offered by the SDB, including querying of links, are a good starting point for integrating such approaches into environments for the production of active documents.

A key requirement induced by the need to use and reuse information in several contexts as is implied by active publications in general is to support meaningful access functionality on a data representation that is system-, device-, application- and product- independent. The Helsinki Structured Text (HST) Database System [24] has been proposed for maintaining structured documents. Compared to the SDB, the HST does not conform to SGML and does not support the generation of unique identifiers for document references. Moreover, the HST has no mechanisms for representing shared documents. In addition, hypermedia databases such as HAM [8] and HyperBase [34] manage the application-independent storage, management and retrieval of hyperdocuments. But besides simple data integrity—for example prohibition of dangling links—they provide no consistency checks over the document structure and no type checking facilities to ensure the DTD conformity of documents. An approach to extend the functionality of hypermedia databases is to implement them on object-oriented database management systems that provide schema definition and checking facilities [26]. We expect considerable improvements of the implementation of the SDB and the extensions of functionality towards HyTime (see the section, “The Structured Document Base”) by moving the SDB to the object-oriented database management system VODAK [30].

The versioning model implemented in the vSDB is a first approach to supporting versioning for hypertext. The fine-grained, heavily interdependent object structure of the hypertext network in addition implies cognitive overhead problems during version creation and aggravates disorientation problems during version selection. Extensions to the initial model proposing to keep contextual information with the versions that guide version creation and help for version identification have been published in [18].

Others [6,9] have proposed to learn from paper documents in order to alleviate disorientation problems in electronic documents, too. The consistent application of document metaphors to electronic document presentation has not been pursued by many approaches. The Digital News System [17] maps each newspaper article manually onto a separate HyperCard card. NewsPeek [28] uses the metaphor of a newspaper front page and Newspace [5] the metaphor of a set of newspaper article clippings for an news-oriented information retrieval system. Both interfaces are generated automatically, but they only consider

the incremental update and delivery to a single, tailored target system. DynaText [23] is an approach to delivering reference work described as SGML-conformant documents as electronic books to various window systems. It offers a simple default presentation for each document. More complex presentations have to be programmed explicitly using the DynaText System Integrator Toolkit.

The Hypermedia Presentation Composer differs in many ways from formatters for print products or conventional editor systems. Although the Hypermedia Presentation Composer uses the model of boxes and glues as described in [25], it does not perform typesetting on the character level. The focus of the Hypermedia Presentation Composer is rather the delivery of well-designed electronic documents to various systems. The typesetting functionality, for example determining the extensions of a textfield given a selected font and content, is performed by the hypertext target system, which is guided and controlled by the Device Driver. But the rules used to determine size and place may differ according to the specific requirements of a publication, such as the different presentations of the IEN, which are stated in the respective presentation style. Moreover, the modular architecture allows one to configure the Hypermedia Presentation Composer easily to new extensions and facilities of hypertext systems.

Compared to the work on editor systems, such as EmbeddedButtons [7] or active documents in Tioga [36] and Interleaf [12], HyperNeWS plays the role of an editor system for the active hypermedia IEN issues and the Hypermedia Presentation Composer is a tool to generate these active documents for different editing environments. Owing to the functionality offered by HyperNeWS, the IEN issues delivered to the user may contain all kinds of activities mentioned in [7], [36] or [12], allowing the content and presentation of the publication to change according to the reader's activities (see the section, "Exploring metaphors for active document interfaces").

However, the documents delivered to the reader are only available in read-only mode [36], i.e., the activities of the documents cannot be changed or edited by the user. From the point of view of a publication designed by a publisher for an intended audience this approach is meaningful. For example, the kind of activities provided by an active publication also depend on the metaphor chosen for user interaction: whereas the mailtool-metaphor IEN automatically includes the most recent articles into the publication, such an approach is impracticable for the newspaper-metaphor IEN supporting the identification of articles by their fixed location on a specific newspaper page in a certain newspaper issue.

In comparison with the work on Tioga [36] and Interleaf [12] to provide an editor environment to create and consume active documents, the aim of the IPE is to generate well-designed active documents for different editor environments based on abstract specifications of document content and document activities. The environment to create and maintain these specifications may be totally different from the environment for document delivery. Of course, the presentation and interaction specifications applied to the generation of an active document have to match the capabilities of the chosen delivery system. For example, the mapping of the Hypermedia Product Document to concrete hypertext objects in HyperNeWS requires the provision of the appropriate HyperNeWS object prototypes such as a prototype for a headline list capable of updating the color of headlines. Our approach to generate active documents for different environments is only a first attempt in this direction. A lot of work remains to be done on identifying and standardizing the specification of activities (see below).

5 DISCUSSION AND CONCLUSION

The main conclusion we can draw from the IEN experiment is that the application of document structuring concepts to the publishing of electronic documents offers potential enhancement to overall publishing environments in general. It also may assist in the production of other kinds of electronic publications, such as the ipsiLex hypermedia lexicon service [13].

The central implementation of structured document manipulation, retrieval and navigation in the SDB significantly speeds up the prototyping of task-specific publishing tools. Since the SDB supports an import and export facility for structured documents into and from shared data repositories controlled by a database management-application, existing tools can easily be integrated into the publishing environment.

Moreover, experience gained with the IEN suggest the extension of logical markup towards other aspects of electronic publishing, as they appear for printed and for electronically delivered documents. Note that SGML only defines the logical structure of documents. Any semantics needed for further document processing has to be applied by the applications and is usually stored in attributes. Nevertheless, there are efforts to standardize the semantics of document processing, for example DSSSL [2] or HyTime [31].

While using the IPE, we found two clear requirements for enhancements to the current architecture. First, computer resources have to be allocated for the composition of each individualized issue. While the generation of a newspaper section with about ten articles including background information takes one minute on a SUN Sparc 10 for the mailtool-metaphor IEN, the complex, rule-based layout of the newspaper-metaphor IEN sometimes even takes up to 20 minutes on the RACE TELEPUBLISHING development platform, a DECstation 3100. Clearly, performing these computations on the publisher's site would make the publisher's site a bottleneck in a real world application.

Second, the current architecture, in which only the compiled electronic representation is sent to the user (see Figure 7) hinders the use of private archives or other local data. For example, private archives obviously should be maintained at the reader's site. Since only the hypertext-target-system-specific information is sent to the user, information to be stored in the personal archive has to be derived from this information or to be retrieved from the publisher's site on demand.

All of this suggests a more replicated architecture in which content is composed at the publisher's site, but presentation is computed at the reader's site (see Figure 9).

ACKNOWLEDGEMENTS

This work was supported in part by the CEC RACE project 1075 (TELEPUBLISHING). We thank the partners of the TELEPUBLISHING project and our colleagues of the PaVE department at GMD-IPSI, who made the IEN experiment possible. We would like to thank our anonymous reviewers for helpful comments. Special thanks are given to Jörg Haake, Wiebke Möhr and Ute Sotnik for their comments on earlier versions of this paper.

REFERENCES

1. *Information Processing—Text and Office Systems—Standard Generalized Markup Language (SGML),* ISO 8879, International Standards Organization, (1986).

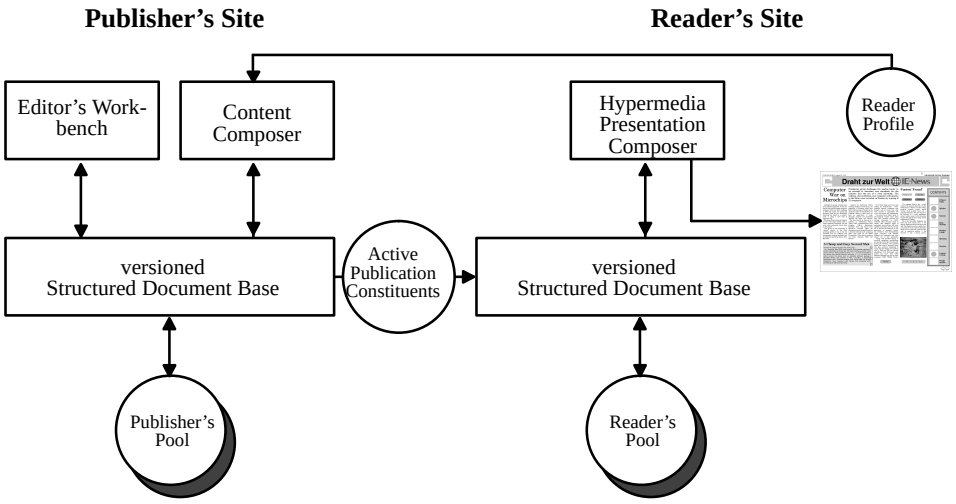

Figure 9. Distributed architecture of the integrated publication environment

2. *Information Processing—Text and Office Systems—Document Style Semantics and Specification Language (DSSSL),* ISO/IEC JTC 1/SC 18 (Draft Proposal), International Standards Organization, (1989).
3. *Guidelines for the Encoding and Interchange of Machine-Readable Texts,* Text Encoding Initiative, Document Number: TEI P1 (July 1990).
4. P. E. Baclace, 'Personal information intake filtering', in *Proc. of the Bellcore "High-Performance Information Filtering Foundations, Architectures and Applications Workshop",* November 1991.
5. W. Bender, H. Lie, J. Orwant, L. Teodosio and N. Abramson, 'Newspace: mass media and personal computing', in *Proceedings of the Summer 1991 USENIX Conference,* USENIX Association, Berkeley, CA, June 1991, pp. 329–348.
6. M. Bernstein, 'The bookmark and the compass: orientation tools for hypertext users', in *SIGOIS Bulletin,* **9**(4), 34–45, (1988).
7. E. A. Bier and A. Goodisman, 'Documents as user interfaces', in *Proceedings of the International Conference on Electronic Publishing, Document Manipulation and Typography (EP90),* ed. R. Furuta, Cambridge Series on Electronic Publishing, Cambridge University Press, Cambridge, September 1990, pp. 249–262.
8. B. Campbell and J. M. Goodman, 'HAM: a general purpose hypertext abstract machine', *Communications of the ACM,* 31(7), 856–861, (1988).
9. F. Cole and H. Brown, 'Standards: What can hypertext learn from paper documents?', in *Proc. of the Hypertext Standardization Workshop,* Jan. 16–18, National Institute of Standards and Technology (NIST), Gaithersburg, MD, PB 90-215864, 1990, pp. 59–70.
10. J. Conklin, 'Hypertext: an introduction and survey', *IEEE Computer Magazine,* **20** (9), 17–41, (1989).
11. G. C. Cruz and T. H. Judd, 'The role of a descriptive markup language in the creation of interactive multimedia documents for customized electronic delivery', in *Proceedings of the International Conference on Electronic Publishing, Document Manipulation and Typography (EP90),* ed. R. Furuta, Cambridge Series on Electronic Publishing, Cambridge University Press, Cambridge, September 1990, pp. 277–290.
12. P. M. English, E. S. Jacobson, R. A. Morris, K. B. Mundy, S. D. Pelletier, T. A. Polucci and H. D. Scarbo, 'An extensible, object-oriented system for active documents', in *Proceedings of the International Conference on Electronic Publishing, Document Manipulation and Typography (EP90),* ed. R. Furuta, Cambridge Series on Electronic Publishing, Cambridge University Press, Cambridge, September 1990, pp. 263–276.
13. D. Fischer and W. Möhr, 'Lexikon-Redaktion: eine Herausforderung für Computer-Assistenz

beim Publizieren' (in German), *GMD-Spiegel, Informationen aus der wissenschaftlichen Arbeit der Gesellschaft für Mathematik und Datenverarbeitung*, March 1991, pp. 40–45.
14. H. P. Frei and D. Stieger, 'The retrieval view, a component of the document architecture', in *Hypertext/Hypermedia '91 (Tagung der GI, SI und OCG)*, ed. H. Maurer, Springer Verlag, Heidelberg, 1991, pp. 99–108.
15. D. Goodman, *The complete HyperCard 2.0 handbook*, 3rd edn. Bantam Books, Toronto, 1990.
16. J. Gosling, D. Rosenthal, S. H. David and M. J. Arden, *The NeWS book—an introduction to the network/extensible window system*, Springer Verlag, New York, 1989.
17. E. M. Hoffert and G. Gretsch, 'The digital news system at EDUCOM: a convergence of interactive computing, newspapers, television and high-speed networks' *Communications of the ACM*, **34**(4), 113–116, (1991).
18. A. Haake, 'CoVer: a contextual version server for hypertext applications', in *Proceedings of the ACM Conference on Hypertext*, eds. D. Lucarella, J. Nanard, M. Nanard and P. Paolini, ACM Press, New York, December 1992, pp. 43–52.
19. W. Hudetz, U. Rachor, A. Haake, K. Reichenberger, I. Liermann, Y. Cotronis, H. Thomas, A. Nulas and V. Reible, 'Evaluation report for the IEN pilot', Technical Report 75/Fhg/ISI/DS/L/064/b0, TELEPUBLISHING Project, RACE-Programme, April 1993.
20. C. Hüser, 'Report on a prototypical interface for structured documents and its application to the IEN scenario', Technical Report 75/GMD/IPS/DS/L/047/b0, TELEPUBLISHING Project, RACE-Programme, August 1991.
21. C. Hüser and A. Haake, 'The hypermedia presentation composer: a tool for automatic hyperdocument delivery', in *Hypermedia, Proceedings der Internationalen Hypermedia '93 Konferenz*, eds. H. P. Frei and P. Schäuble, Springer Verlag, Heidelberg, March 1993, pp. 57–66.
22. G. Ihnofeld, Spezifikation eines Hypertext Composing Prozesses und Entwicklung der Regelsprache für die Transformation von strukturierten Dokumenten anhand einer beispielhaften Anwendung (in German), Diploma Thesis, TH Darmstadt, Fb. Informatik, December 1991.
23. C. Kelley, *DynaText (TM) Product Overview*, Electronic Book Technologies, One Richmond Square, Providence, RI 02906, April 1992.
24. P. Kilpeläinen, G. Lindèn, H. Mannila, and E. Nikunen, 'A structured document database system', in *Proceedings of the International Conference on Electronic Publishing, Document Manipulation and Typography (EP90)*, ed. R. Furuta, Cambridge Series on Electronic Publishing, Cambridge University Press, Cambridge, September 1990, pp. 139–152.
25. D. E. Knuth, *The TEXbook*, Addison-Wesley, Reading, MA, 1984.
26. D. B. Lange, 'Object-oriented hypermodeling of hypertext supported information systems', in *Proceedings of the 26th Hawaii International Conference on System Sciences*, Vol. 3, ed. T. H. Mudge, IEEE Computer Society Press, Washington, 1993, pp. 380–389.
27. H. Lie and W. Bender, 'The electronic broadsheet: all the news that fit to display', in *Proceedings of the International Conference on Electronic Publishing, Document Manipulation and Typography (EP 92)* , ed. C. Vanoirbeck, Cambridge Series on Electronic Publishing, Cambridge University Press, Cambridge, April 1992, pp. 199–209.
28. A. Lipman and W. Bender, 'News and movies in the 50 megabit living room'. Paper presented at *Globecom*, IEEE Computer Society Press, Tokyo, 1987.
29. W. Möhr, L. Rostek and D. H. Fischer, 'TEDI: an object-oriented terminology editor' in *TKE' 93: Terminology and Knowledge Engineering, Proceedings of TKE '93—Third International Congress on Terminology and Knowledge Engineering*, ed. Klaus-Dirk Schmitz, Indeks Verlag, Frankfurt, August 1993, pp. 363–374.
30. E. J. Neuhold and V. Turau, 'Database Research at IPSI', *SIGMOD RECORD*, **21**(1), 133–138, (1992).
31. 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, (1991).
32. V. Quint and I. Vatton, 'Combining hypertext and structured documents in Grif', in *Proceedings of the ACM Conference on Hypertext*, eds. D. Lucarella, J. Nanard M. Nanard, and P. Paolini, ACM Press, New York, December 1992, pp. 23–32.
33. V. Reible and R. Minio, 'Publishing as a broadband application', in *Proceedings of the International Conference on Integrated Broadband Services and Networks*, The Institution of Electrical Engineers (IEE), London, October 1990, pp. 153–159.

34. H. Schütt and N. Streitz, 'Hyperbase: a hypermedia engine based on a relational database management system', in *Hypertext: Concepts, Systems and Applications (Proceedings of the European Conference on Hypertext (ECHT 90))*, eds. A. Rizk, N. Streitz and J. André, Cambridge Series on Electronic Publishing, Cambridge University Press, Cambridge, November 1990.
35. N. Streitz, J. Haake, J. Hannemann, A. Lemke, W. Schuler, H. Schütt and M. Thüring, 'SEPIA: a cooperative hypermedia authoring environment', in *Proceedings of the ACM Conference on Hypertext*, eds. D. Lucarella, J. Nanard, M. Nanard and P. Paolini, ACM Press, New York, December 1992, pp. 11–22.
36. D. B. Terry and D. G. Baker, 'Active Tioga documents: an exploration of two paradigms', *Electronic Publishing*, **3**(2), 105–122, (1990).
37. Verlag Frankfurter Allgemeine Zeitung, *Alles über die Frankfurter Allgemeine Zeitung*, Verlag Frankfurter Allgemeine Zeitung, Frankfurt, 1989.
38. A. van Hoff, *HyperNeWS 1.4 User's Guide*, The Turing Institute, Glasgow, 1990.
39. J. Warmer and S. van Egmond, 'The implementation of the Amsterdam SGML Parser', *Electronic Publishing*, **2**(2), 65–90, (1989). Also issued as a technical report by the Faculteit Wiskunde en Informatica, Department of Mathematics and Computer Science, Vrije Universiteit Amsterdam, 1987.
40. A. Weber and V. Schoepf, 'Prototypical application of a version model', Technical Report 75/GMD/IPS/DS/L/051/b0, TELEPUBLISHING Project, RACE-Programme, January 1991.
41. A. Weber and V. Schoepf, 'Konzepte zur Versionenverwaltung für die Hyperdokumenterstellung in einer hypertextbasierten Publikationsumgebung' (in German), in *Hypertext/Hypermedia '91 (Tagung der GI, SI und OCG)*, ed. H.Maurer, Springer Verlag, Heidelberg, 1991, pp. 274–285.