Toward Active, Extensible, Networked Documents: Multivalent Architecture and Applications

Thomas A. Phelps and Robert Wilensky
University of California, Berkeley
{phelps, wilensky}@CS.Berkeley.EDU

Abstract

Rich varieties of online digital documents are possible, documents which do not merely imitate the capabilities of other media. A true digital document provides an interface to potentially complex content. Since this content is infinitely varied and specialized, we must provide means to interact with it in arbitrarily specialized ways. Furthermore, since relevant content may be found in distinct documents, we must draw from multiple sources, yet provide a coherent presentation to the user. Finally, it is essential to be able to conveniently author new content, define new means of manipulation, and seamlessly mesh both with existing materials.

We present a new general paradigm that regards documents with complex content as "multivalent documents", comprising multiple "layers" of distinct but intimately related content. Small, dynamically-loaded program objects, or "behaviors", activate the content and work in concert with each other and layers of content to support arbitrarily specialized document types. Behaviors bind together the disparate pieces of a multivalent document to present the user with a single unified conceptual document. As implemented in Java in the context of the World Wide Web, multivalent documents in effect create a customizable virtual Web, drawing together diverse content and functionality into coherent document-based interfaces to content.

Examples of the diverse functionality in multivalent documents include: "OCR select and paste", where the user describes a geometric region on the scanned image of a printed page and the corresponding text characters are copied out; video subtitling, which aligns a video clip with the script and language translations so that, e.g., the playing video can be presented simultaneously in multiple languages, and the video can be searched with text-based techniques; geographic information system (GIS) visualizations that compose several types of data from multiple datasets; and distributed user annotations that augment and may transform the content of other documents.

In general, a document management infrastructure built around a multivalent perspective can provide an extensible, networked system that supports incremental addition of content, incremental addition of interaction with the user and with other components, reuse of content across behaviors, reuse of behaviors across types of documents, and efficient use of network bandwidth. Multivalent documents exploit digital technology to enable new, more sophisticated document interaction.

The work reported here was supported in part by National Science Foundation grant IRI-9411334 as part of the NSF/NASA/ARPA Digital Library Initiative.

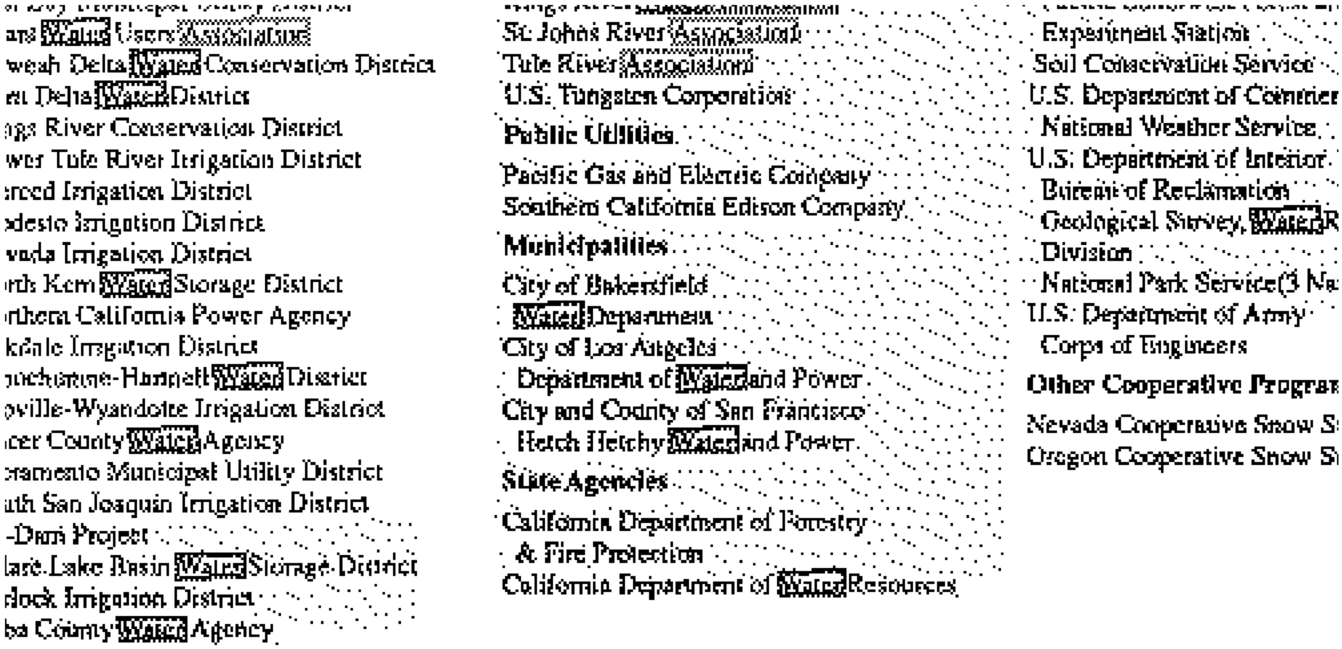
1. Introduction

"Multivalent Documents" is a new paradigm for the organization of complex digital document content and functionality. The multivalent view of a document contrasts with monolithic data formats that attempt to encompass all possible content types of the document in a single, complicated specification, and provide for interaction with them with concomitant complicated editors, often of limited extensibility; this means that, especially for specialized documents, the system can be simultaneously overwhelming and lacking. In contrast, the multivalent approach "slices" a document into layers of more uniform content, to which additional layers may be added at a later time at equal status. Functionality is provided by relatively lightweight, dynamically-loaded program-objects, called "behaviors", that manipulate the content. Behaviors may communicate with multiple layers and other behaviors, and new behaviors can be added subsequently at equal status. Layers and behaviors are described in a typing system like that found in many programming languages, and all such pieces relevant to a particular document are dynamically composed via a "type graph".

In conceiving of documents as interacting layers of content and functional behaviors that unite to present a single conceptual document, our paradigm introduces new capabilities and benefits. By representing a document as multiple components, others besides the original document's author can add additional content and behavior at a later time, even introducing technological innovations that were unknown during the preparation of the initial materials. These additions can be made locally, without special cooperation of the server of the base document. By keeping the pieces of content simple, the behaviors that manipulate them are relatively straightforward to define; the components are also more easily reusable. By storing the content in distinct pieces, only those necessary to support the requested interaction need be shipped over the network, thus conserving bandwidth.

The general paradigm of multivalent documents is described at length elsewhere [Phel96]. In this paper we describe a series of applications of the multivalent framework in order to illuminate its breadth of application and richness of functionality. We then provide a description of the multivalent architecture. This architecture is a networked model of digital documents that operates on top of a World Wide Web (WWW) [Bern92] augmented with protocols that allow more sophisticated structuring. The loosely coupled nature of the architecture introduces issues which are considered thereafter. We conclude with a look at related work and a statement of current status and future plans.

Published in *Proceedings of ACM Digital Libraries '96* / Bethesda, Maryland / March 20-23, 1996.



ans Water Users Association
weah Delta Water Conservation District
st Delta Water District
ngs River Conservation District
wer Tule River Irrigation District
erced Irrigation District
odesto Irrigation District
vada Irrigation District
rth Kern Water Storage District
orthern California Power Agency
kdale Irrigation District
uolumne-Hatnell Water District
oville-Wyandotte Irrigation District
acer County Water Agency
cramento Municipal Utility District
uth San Joaquin Irrigation District
-Dam Project
lare Lake Basin Water Storage District
rlock Irrigation District
ba County Water Agency

St. Johns River Association
Tule River Association
U.S. Tungsten Corporation
Public Utilities
Pacific Gas and Electric Company
Southern California Edison Company
Municipalities
City of Bakersfield
Water Department
City of Los Angeles
Department of Water and Power
City and County of San Francisco
Hetch Hetchy Water and Power
State Agencies
California Department of Forestry
& Fire Protection
California Department of Water Resources

Experiment Station
Soil Conservation Service
U.S. Department of Commer
National Weather Service
U.S. Department of Interior
Bureau of Reclamation
Geological Survey, Water R
Division
National Park Service(3 Na
U.S. Department of Army
Corps of Engineers
Other Cooperative Prograr
Nevada Cooperative Snow S
Oregon Cooperative Snow S

Figure 1: This is a scanned image of a printed page, which is the initial digital capture format for most documents in Berkeley's Digital Library Project. In many ways, the user can work with the image of the page as if it were a word processing document. The prototype, which is implemented in Java, refers to a layer of geometrically-positioned characters to provide "select and paste" functionality (of the corresponding OCR or alternative text), hyperlinks, definitions dynamically served over the network, and searching in a WWW client. In the figure above, the shaded region has been selected, and a search has yielded the boxed words.

2. Applications of Multivalent Documents

2.1 Prototypes on Scanned Page Images

As a very simple example of a multivalent document, consider documents that began their lives as scanned page images. In the UC Berkeley Digital Library Project [Wile96, Wile95] (http://elib.cs.berkeley.edu), we have 100,000 pages of such documents from the California Department of Water Resources. It is straightforward to run optical character recognition (OCR) on such a collection and hence make the documents available for access by full-text search, and we have done so. The page images, though, remain inert pictures of text, with little functionality; one must move back and forth between images and OCR to make selections or perform searches.

Yet the scholar may prefer to work with the page images because, in general, it is the only digital representation that maintains the full information content of the original, including illustrations, layout, and uncommon markings (as in musical or mathematical notation). This is vitally important for historically significant documents such as hand-written manuscripts, illuminated books, and fine press materials, where the typography materially contributes to the value of the work. Even were the OCR to be perfect, translations into other digital representations inevitably lose information. And it is difficult to predict what needs preserving; at the extreme, it is possible that in the course of time that even the worm holes become significant, as a variant interpolation of the text eaten out may yield a different translation and, perhaps, a new interpretation.

In many other ways, however, scanned page images can be seen as the worst case of digital document formats. Starting with array of pixels, we must construct the content of the document, information that is directly available in other formats. However, the general multivalent document model has nothing to do with images. Indeed, the multivalent framework can be usefully applied to other document formats. In case of SGML, for instance, the user will still want the higher-level functionality—adding annotations, sorting tables—while the implementor gains the significant advantages of extensive markup and control of text presentation. From the multivalent perspective, scanned page images, like PostScript or TeX's formatted DVI files, simply fix the presentation or appearance. With suitable interfaces to manipulate the presentation, higher-order functionality can be reused across source formats. (Application of the model to SGML or HTML is an area of current research.)

Consider now our prototype multivalent document implementation. Figure 1 displays the image of a scanned page (with telltale page skew). Underlying this image layer are two more layers: the corresponding ASCII characters and their geometric positions on the image (both generated automatically). A program behavior allows the user to select regions of the image and subsequently paste the corresponding OCR into another document. In addition, users can search in the image as if it were a document in a word processor. In this case, the user's query is matched against the underlying OCR; hits are then mapped back into the corresponding geometric regions in the image, where they are given a yellow background. Although this is a relatively simple image, the same techniques would work for text labels embedded in maps and for handwritten text.

Additional information layers yield more sophisticated interactions for variations of select and paste. In the case of mathematical equations layers, one could manually prepare various encodings of the notation—TeX markup, PostScript, Mathematica code—and make these available to the user, who could choose to extract the representation most appropriate to the task at hand. Bibliography entries could be stored similarly, but in an abstract

Facility	Gross Capacity (Acre-feet)	Surface Area (Acres)	Shoreline (Miles)
Antelope Lake	22,600	930	15
Frenchman Lake	55,500	1,580	21
Lake Davis	84,400	4,030	32
Lake Oroville	3,537,600	15,800	167
Thermalito Forebay	11,700	630	10

Figure 2a: This is a table image from a scanned document. A multivalent framework may associate structural information such as a description of the rows and columns of the table, as well as the ASCII text in each cell.

Facility	Gross Capacity (Acre-feet)	Surface Area (Acres)	Shoreline (Miles)
Lake Oroville	3,537,600	15,800	167
Lake Davis	84,400	4,030	32
Frenchman Lake	55,500	1,580	21
Antelope Lake	22,600	930	15
Thermalito Forebay	11,700	630	10

Figure 2b: With the content available in a multivalent document decomposition, a table behavior can respond to, say, a mouse click on the "Surface Area" heading by sorting the table by that column and displaying the results by rearranging the pixels of the image.

form amenable to mechanical parsing (perhaps taken directly from a BibTeX database); in this case, a behavior could automatically translate the information into other formats such as Refer or EndNote.

Figure 2 shows another multivalent prototype implementation, as before and after images of a table manipulation. Of course, an author must choose a particular ordering before committing a table to print. In an online multivalent framework, the description of the table's rows, columns, and cell contents can be supplied as semantic layers. In this way, we can manipulate the table interactively. The prototype uses advanced document analysis techniques, developed in the project, that derive the supporting layer information. Now, in response to, say, a mouse click on a given column heading, the prototype sorts the table image by that column and displays the results by rearranging pixels in the page image; columns are typed so that numbers are sorted numerically, everything else alphabetically. One could consider adding other functionality, such as summing rows or columns, or even placing the table in a general spreadsheet behavior. Fortunately, the multivalent document model allows incremental addition of such behaviors even after initial authoring.

In the multivalent model, sources of document functionality need not be limited to the browser and the server from which it came. We envision legions of networked document services, and have incorporated two in our prototype. The first is a converter located at Xerox PARC that takes the output of Xerox's ScanWorX OCR software (XDOC format) and returns word bounding boxes, thus establishing the correspondence between image and text. The prototype calls this service on the fly for each page.

The other distributed document service takes a chosen word, sends it to Encyclopedia Britannica's Web site to be defined, then filters the returned HTML and presents it to the user. On the following page, Figure 3 depicts this for the word "precipitation".

Figure 3 also illustrates hyperlinks, with links sources in both text flow and map.

2.2 More Complex Instances

Although a multivalent approach can be taken for simple documents, it is most valuable when applied to complex digital documents. By "complex document" we mean documents that contain a large number of varied types of information and therefore could benefit from sophisticated interactions with users. An example of a complex multivalent document that incorporates a dynamic medium is subtitled video, which aligns a video clip with the script and language translations. The playing video can be presented simultaneously in multiple languages without a heavyweight copy of the video stream for each language, and also the video can be searched with text-based techniques.

2.2.1 Geographic Information Systems

Given the Digital Library Project's focus on environmental information, an area of particular interest to us is geographic information systems (GIS's). GIS's already regard their content as layers, in which some geographic coordinate framework is overlaid with different kinds of information, such as representations of the shape of land masses and water bodies, political borders, highways, the placement of important structures, and measurement data of various kinds.

Current practice in the GIS world is to construct distinct systems for GIS content, e.g., ArcView. Indeed, our project has used such a stand-alone client, called Napa [Brow95], which visualizes different types of geographic coverages stored in a data base. Like many GIS systems, Napa visually positions users at an altitude and provides a view of the earth from that latitude. The user can then pan and zoom. The system can be easily configured so that each of the various data sets are displayed only at certain altitudes. Thus, from 100 miles up, there is no point in visualizing bridges and roads, but one would want to see these as one gets just a short distance above the ground. In addition, the user can easily turn off and on the display of individual data sets. As the user pans and zooms, the Napa client issues queries to the database to retrieve any portions of the visualized data sets that are needed to fill out the visible geographic region of the display. Some of the data sets are visualized directly, e.g., a county boundary is usually a polygon drawn on the screen. Others, such as aerial or terrestrial photographs, are displayed as icons; clicking upon them causes the designated object to be displayed.

From a multivalent perspective, it is possible to provide similar functionality without a special purpose previewer. For example, consider a (traditional) document in which a map presents a static view of some set of geographic information. In this instance, various sources of data are leveled into a bit-mapped representation. However, in the multivalent approach, we would retain the individual layered structure that compose into the visual depiction. The particular presentation simply requires previewer capabilities for the various levels of structure, plus rules of composition—exactly what the multivalent document infrastructure provides. Given these capabilities, it would be possible to attach a map to a document with no information content and introduce new means of manipulating it, or override content or behavior with improved or customized versions.

2.2.2 User Annotations and Collaboration

User annotations is the other main area of present interest in the project. User annotations are theoretically no different than adding more layers to the existing model. However, some elaborations of a document are so common as to call for special authoring support. In the multivalent model it is straightforward to add functionality commonly found in annotation packages, such as the ability to draw text and geometric shapes on the document image and to attach scrolling text boxes to points in other semantic layers. But working in a multivalent framework offers other benefits. Consider the example of the Talmud. The text corresponding to the Jewish law occupies only a small portion in the center of the image, where it is surrounded by various Rabbinical commentary. A semantic structural layer could describe the location of these various commentaries. The scholar is instructed to read each commentary as if each were the only one on the page [Holt84]. This can be facilitated in the multivalent architecture with a general behavior that examines the structural layer and erases all commentaries but the selected one. This contrasts with an annotation system that simply attaches annotations to a point in the document; in these systems one would need to annotate each region separately for each distinct use of that region.

The multivalent specialization described above can be summarized as an annotations system in a networked environment with an active client. One could leverage this work in the development of a live computer-mediated collaboration system. The input and output channels of content would need to be made live and aware of participants in a meeting session, but there already exists a basic network awareness and document manipulation capability for more than a shared whiteboard facility.

3. High-level Multivalent Architecture

3.1 Composable Behaviors

The behavior corresponding to a semantic layer has three kinds of components: the data or information content of the layer, functional interfaces for its own use and the use of other behaviors, and one or more user-level interfaces. This is diagrammed in Figure 4 below.

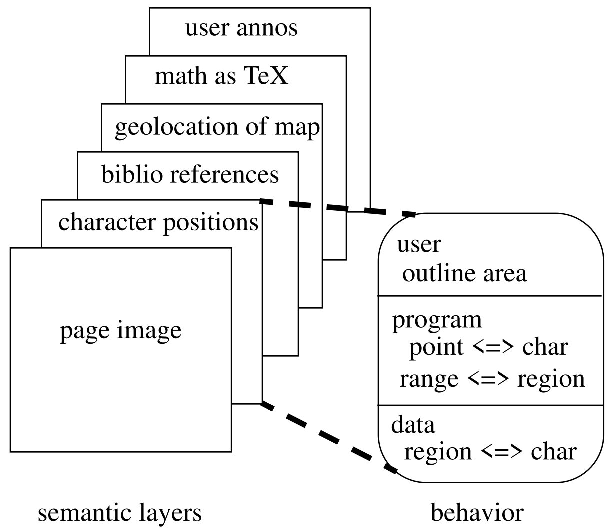


Figure 4: At left, the content of a scanned document conceptualized as semantic layers of information.
At right, the programmatic or behavioral implementation of a layer (highly abbreviated).

Behaviors resemble classes in object-oriented programming languages, encapsulating each semantic layer as a set of methods

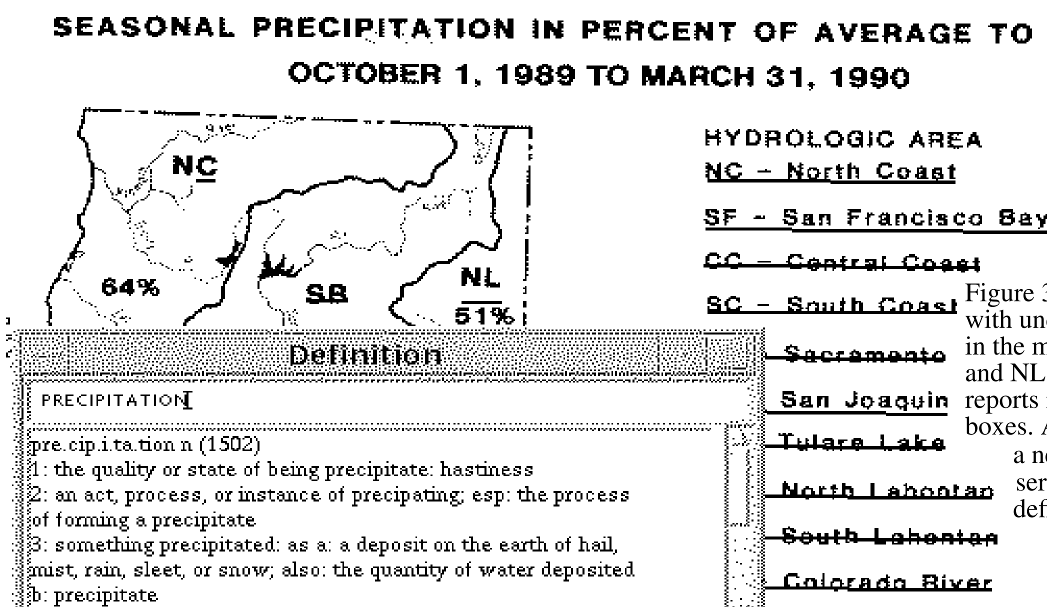


Figure 3: Hyperlinks are shown with underlines. Note the links in the map region, for NC, SB and NL. Unfortunately, the OCR reports imprecise word bounding boxes. Also note the use of a networked document service in retrieving the definition of "precipitation".

that operate on private data. (Indeed, they are implemented as such a set of distinguished classes.) That is, data in a behavior is not directly accessible by other behaviors, which are therefore forced to operate at a higher level than bit-tweaking a byte stream, which is a volatile format. Instead, behaviors communicate through program-level interfaces.

Consider the multivalent document depicted in the diagram above as a concrete example. This document consists, in part, of a scanned page image with geometrically positioned OCR. The character layer stores the mapping between character positions in the text stream and their location on the image. These locations may be stored directly as bounding boxes, or they may be given as character origins along with a single global pointer to the corresponding font metrics from which the bounding boxes may be calculated. The exact method is hidden to other behaviors, since all access is provided through higher-level function interfaces. The select and paste behavior needs to map a mouse click to a character position, and take it and subsequent mouse drags and highlight the intervening characters. It does this by taking the two positions given by the initial mouse click and the current mouse position and mapping these to character positions in the text stream; these text positions are then transformed into the region which is drawn on the image. Notice that this region is not usually the rectangular region whose corners are the two points given by the mouse, as it obeys line boundaries; therefore, it is useful to utilize a behavior that is intimately familiar with the data.

As described later, a variety of behaviors may be active at any point in time, on various regions of the document. To aid the user in determining what is active where, most behaviors can outline its area of control as one, and perhaps its only, user-level interface.

Not all behaviors have all three components. Some behaviors are data-centric, serving primarily as information repositories, with enough program-level interfaces to provide access to the data. Other behaviors are program or functionality-centric. For instance, a general searching behavior may not store any data in itself, except perhaps a list of “stop words” that should be ignored during the search. The searching function calls upon other behaviors to provide the text to search. At the user-level, it may have the ability to present a simple type-in box for the search term, or it may rely solely on other behaviors to invoke it. Still other behaviors primarily provide a user interface to functionality and content available elsewhere. Most customization by the average user will take sophisticated functionality developed by experts and mold the interaction with them to personal taste.

3.2 Behaviors in Operation

Often the semantic layers of a multivalent document are geographically dispersed on various repositories. For example, perhaps the page images of the U.S. Constitution may be stored in the Library of Congress server, the OCR derived locally, the Japanese language translation maintained in Kyoto, and a line-by-line analysis pointing out the influence of the French Revolution exhibited as a technology demonstration at the new library complex in Paris.

The more cooperative servers are fronted by a database that can respond to queries by “name” for a specific piece of information, as opposed to a URI that maps more or less directly into a file system, and for entities matching a description of its attributes, like “semantic layers associated with document 28329” or “behaviors that can search Unicode”. As described later, though, no particular cooperation from a server is required beyond delivering the raw data.

The user controls a client that can communicate with various servers. In the scenario diagrammed on the previous page, the user

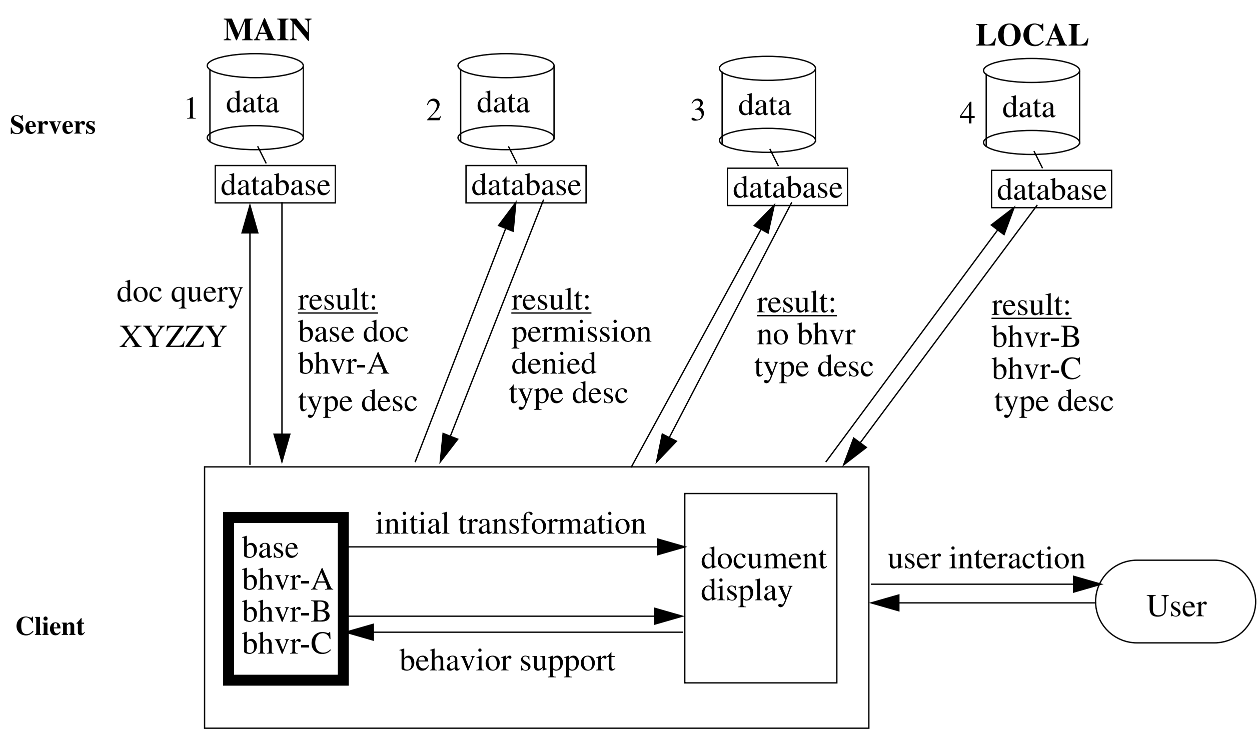


Figure 5: In response to view a given conceptual document, the client queries relevant servers, loading content and functional behavior essential to operation as well as concise descriptions of the information available at that server.

has requested the conceptual document named “XYZZY”. The query first goes to a “handle server” (similar to that described in [Kahn94]) that takes the name and returns a list of servers that have relevant information. The client queries this list of servers for the essential behaviors for minimal interaction with the document. Other behaviors are loaded as needed; this on demand loading conserves network bandwidth, which is important considering that a multivalent decomposition is most appropriate for complex documents with a great deal of content. Typically the main components of the document are fetched from a remote source. In Figure 6, server number one (‘MAIN’) holds the page image of the document and ‘bhvr-A’, which has been deemed essential. The user does not have permission to use server two. This may be a commercial supplier of commentaries, with whom the user does not have an account. Server three returns nothing immediately. In this case the handle server may have out-of-date information but more typically the server does not have any essential behaviors. The last server (“LOCAL”) returns two behaviors. They may be the user’s personal annotations on the page, or they may be wrappers that automatically invoke other behaviors, in effect locally declaring those remote behaviors as essential.

In all cases the server returns a series of “type descriptors”, that is, concise characterizations of the behaviors at that server. This information is used in constructing the type graph.

3.3 The Type Graph

All servers for which the client has access permission return a “type descriptor”. Each layer and behavior is typed, and the *type graph* is a construction of the relationships among pieces. When a behavior needs a particular information layer or action, it consults the type graph for an object that can satisfy it. If the behavior is not available locally (and not cached), it is fetched at this time over the network. The type graph is key to managing interactions among behaviors. Because the type graph is locally managed, it can be massaged and rearranged. One such use of the type graph is to override a behavior or piece of a behavior, say from the official repository for a particular document, with a local customization.

Another use of the type graph is to introduce new technology in a first class way into documents that were initially prepared before its development or in ignorance of it, but at any rate without special accommodations made for it. How the type graph facilitates seamless integration of unanticipated behaviors is illustrated by the TileBars [Hear94] searching visualization method. Briefly, in a TileBars interface, the user enters search words for multiple term sets. For instance Figure 7 illustrates the result for one document of a search for two term sets, with term set one comprising “flood fire drought” and term set two, “damage insurance”. A separate search is performed for each term set. The documents to be searched must be segmented, or *tiled;* a simple tiling may take each page as a tile, while a more sophisticated one would segment along lexically coherent paragraph units. A TileBar is proportional in length to the length of a document with one row per term set, and greater concentrations of hits within a tile square are represented by darker colors. As diagrammed in Figure 7, this representation displays the

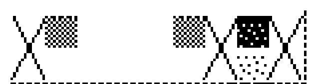


Figure 7: A TileBar. The length of the bar is proportional to the length of searched document, and the darkness of tile squares represents the concentration of hits in that area of the document. X’s indicates elided pages; the rightmost two tiles, with heavy mention of term set one and passing reference to term set two, is page 147.

pattern of hits within a document. Moreover, one can compare the patterns for different term sets to quickly see where each set is strongly mentioned or perhaps where one set is strongly mentioned but where there is only passing reference to the other. (A TileBars interface to the Berkeley’s document collection is available at http://elib.cs.berkeley.edu/tilebars.html.)

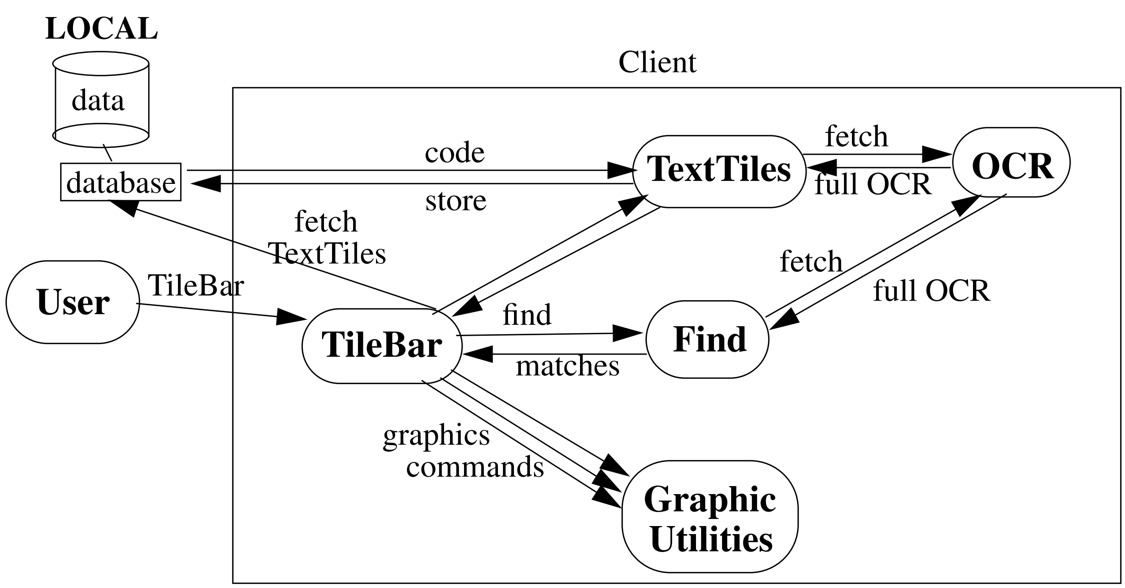


Figure 6: To create a TileBar visualization for a search on a document created before the invention of this technology, the type graph first calculates a TextTiling segmentation of the document on the fly.

Initially in Figure 6 on the previous page, the user is studying a page from the document in his client, the client has a reference to "TileBars" in its type graph, and the code for TileBars resides on the local server. Now the user requests a TileBars search. To construct this search visualization, the TileBars code needs to learn the locations of the matches in the document and the document tile to which that position corresponds. The maintainers of the authorized digital document may not have provided a text tiling, but the type graph reports that one can be calculated on the fly with TextTile code on the local server. After it is dynamically loaded, the code calls upon a behavior that can supply the characters of the document, which are segmented into tiles; these results are reported to the TileBars and may be cached locally. Likewise a standard search utility requests the characters and reports its results to TileBars. Armed with the information it needs, TileBars user-level code calls upon standard page drawing utilities to build the visualization. The TileBars user-level code actively manages interaction with the bars so that it can respond to a mouse click in a tile by directly displaying the corresponding page.

Given the preceding exposition of the Multivalent Documents framework, we can appreciate the name. The Merriam-Webster Collegiate Dictionary defines "valence" as the "relative capacity to unite, react, or interact". This new model digital documents is called "multivalent" in recognition of the fact that layers and behaviors, united through the type graph, interact with the user as a single coherent document similarly to the way atoms bond to form compounds with new properties. But whereas atoms share only their outermost electrons, in the multivalent document model every piece has the potential to react with every other.

4. Issues

4.1 Misalignment of Layers

The intimately related but decoupled nature of multivalent documents introduces a concern: a change to document could misalign other layers. Although difficult to fully combat in the most general case, an array of strategies can mitigate the practical impact of this possibility. In some cases, a layer of content does not change: chances are that the PostScript of a Berkeley technical report from 1988 will never be updated. In the case of a scanned page of text, its structure and the textual transcription, changes in the scanned page can be fully propagated through to the others by completely recomputing them. For other cases, the layer can be versioned at the server. Attachments to that layer also record its version number, and clients can then either request the specific version, or the new version along with a list of changes from the requested version so that the layers can be realigned. An ad hoc association between layers on volatile and uncooperative non-versioning servers can record context in order to increase chances reorienting itself after a mutation. Finally, some layers are unaffected by changes to an associated layer. Anything referring to a structural unit is related on an abstract level, and when the geometric location of another unit changes, the relationship on the symbolic level is undisturbed—in fact it has the desirable property that it is transparently updated to operate properly with the new information.

4.2 User Interface Coherence

Another problem is how to present a comprehensible user interface in the face of competing behaviors from varied sources. In a sense, learning the space of interaction with a new document specialization is much the same as learning a new application program, and the same techniques used in graphical user interfaces could be applied. That is, functionality common to most multivalent documents, searching perhaps, would be standardized, and other functions could be placed into menus of commands. Behaviors that introduce interaction modes would first be required to register the user interaction gestures in which they are interested, and only modes with disjoint sets of gestures would be allowed to coexist; enabling one automatically disables conflicting behaviors. Alternatively, a Toolglass and Magic Lens [Bier93] style of interaction would seem to complement a multivalent document approach perfectly. Toolglasses carry collections of Magic Lens tools, and the user invokes a tool on a document by placing a lens over the affected area and operating through the lens. Multivalent documents would provide the content and a set of tools; Toolglasses could organize them and a Magic Lenses would provide an excellent interaction paradigm.

5. Related Work

5.1 Compound Documents

A comparison with compound document systems like OpenDoc [Nels95] and Microsoft's OLE [Broc95] highlights the distinctive decomposition of the multivalent approach. In analogy to operating systems, the relationship between compound documents and multivalent documents parallels that between processes and threads. That is to say, compound documents organize multiple editors in the same document but at a much coarser level of granularity. In the compound approach, a geometric region of the document can be devoted to a particular, usually heavyweight, editor, which can introduce new content and functionality into the container document. Although OpenDoc allows an editor to communicate with other editors to access their data and request actions, both OpenDoc and OLE fundamentally view the document as a set of very loosely coupled data types spatially arranged. Content is coupled with functionality and screen real estate, and each functional unit operates primarily with a single, complex data type. In the multivalent approach, complexity is built up from simple content and most interesting behaviors operate on multiple layers of content and with other behaviors.

5.2 Active Documents

Many systems activate the digital page with program script control, including Active Tioga Documents [Terr90], Embedded Buttons [Bier91] and Computational E-mail [Bore92] to list but a few. In general these systems connect a script to a specific region of the document; in these regions the document is active, and elsewhere the document retains its ordinary properties. In the multivalent approach, active regions vary according to what behaviors are active, and the content of that region varies according to what semantic layers are available.

Others systems, such as Henry [Silv94] and Firefly [Buch92], coordinate external viewers or editors through narrow interapplication interfaces to induce on the heterogeneous content composite hyperlinks and temporal behavior. In the multivalent approach, document behavior is programmed in the native object system, but access to advanced tools is maintained: Dedicated, sophisticated analyzers and editors can be used at document preparation time to compute or compose information, and the results are stored for manipulation at runtime by less heavyweight code.

5.3 Multiple Representations

Many multivalent documents use multiple representations in the service of one or a few presentations. This inverts the Smalltalk model-view-controller paradigm [Kras88] which maintains a single shared model in support of multiple views.

RightPages [Stor92] models scanned journal article pages "as three planes of information: the image, OCR text, and page layout". As well, it maintains the representations distinctly (on disk) and as a (C++) object internally. RightPages is an example of a multivalent perspective in the form of a custom system for a fixed and small number of layers.

Starting with paper documents and operating on scanned images, HyperFacs [Myka95] automatically analyzes document and generates hyperlinks, which can traversed with the accompanying browser. HyperFacs considers the problem of scanned document analysis, which the multivalent work does not, and models a document as a set of layers, but limits the application of the model to a fixed and small number of layers, for scanned pages only.

In order to edit scanned images of text, Image Emacs [Bagl94] constructs on the fly a map of connected components that roughly corresponds to characters—sidestepping the time and complexity of optical character recognition—and operates through this layer in reformatting "characters" as requested by editing commands. Multivalent documents share with Image Emacs the characteristic that both manipulate the document through layers of partial information, without demanding the "native" or source representation. Although a multivalent behavior could also identify the connected components on the fly also, its bias is toward caching computationally intensive analysis for use by portable lightweight code.

5.4 Annotations

Annotation systems maintain a distinction between a "base" document and the additional annotation material, a distinction that is generalized in a multivalent document. It is useful to be able to store annotations separately from the base document for several reasons: the source of the document may not be mutable (e.g., it comes from a read-only server or a CD-ROM), the data format may not be amenable to storing annotation information (a digitized sound clip meant to be streamed as raw data through a speaker driver may not have accommodations for annotations), and it should be possible to annotate shared documents without viewing annotations made by others. TkMan [Phel94], a graphical, hypertext browser for UNIX manual pages, allows the user to mark regions as if with a yellow highlighter marker. Highlighting information is stored in a user's personal database and merged with manual pages as they are shown; when a manual page changes, as determined from its file modification date, the user is asked whether or not to throw out the highlights wholesale, but no attempt is made to adjust the highlights to the modified document. Adobe's Acrobat [Walt94] supports textual attachments tied to a geometric position on the page, and annotations from several users can be combined into a composite page, but no attempt is made to adjust to a modified base document. More advanced annotation systems like CoNote [Davi] and ComMentor [Rosc95] tie textual annotations to particular locations in the text stream, and compose annotations from multiple users; the latter system is robust against changes to the original document. Because of the historic lack of programmability of WWW clients, these last two systems compose the text and image streams at the server.

6. Implementation

Although Mosaic [Andr93] and Netscape [Nets94] are sufficient for simple viewing of simple media like images and text, full multivalent interaction requires a browser that can execute the program-objects. We have chosen Sun's C++-like Java language [Gosl95]. Additionally, Java has the important properties of being platform independent, secure, and dynamically loadable. The basic document model of Java's related HotJava WWW browser mimics a compound document in that rectangular regions of the document are controlled by dynamically-loaded program code. We expand this region to control the entire document (interaction with HTML is an area for future research). All document content and program code is stored in the object-relational database Illustra, the commercial version of Postgres [Ston91].

We are applying the multivalent document paradigm to source materials important to the University of California, Berkeley's Digital Library Project, initially to California Department of Water Resources technical reports. These reports consist of scanned page images that are analyzed for structure, characters transcription, and tables. As more content, such as maps and mathematics can be extracted intelligently, it will be added as additional layers.

With the goal of making the digitized collection widely available, we deliver materials over the World Wide Web as much as possible. Section 2.1, which includes Figures 1, 2, and 3, describes functionality currently available in a prototype implementation. This image-centric functionality includes: OCR select and paste, select and paste of alternative text, searching, hyperlinks, networked definitions, and table sorting. We have also implemented TileBars, although not yet in multivalent form.

In the future, the Project's rich collection of geographic materials will take advantage of the multivalent framework articulated above in the flexible delivery over the network of functionality found in dedicated geographic information systems (GIS).

The other current focus of research is a distributed annotations system. Although no different from other behaviors at an abstract level, an annotations system needs a friendly user interface for such common annotation styles as adding a textual comment or hyperlink, graphically embellishing an image, and eliding a region of content. One possibility for an advanced annotation type is a rules system that the user could program to invoke a program behavior (as a black box) in response to patterns identified in document content.

7. Summary

We have described a new model for digital documents called multivalent documents. It takes a fine-grained, object-oriented perspective on digital documents that is well suited to a networked world of rapidly evolving document technology. This paradigm views digital documents, especially complex ones with varied content and styles of interaction, as comprising homogeneous layers of semantic content. Content is activated by program behaviors that are dynamically loaded from geographically distributed locations. Behaviors specialize interaction for classes of documents, and unify semantic layers to present the user with a single conceptual document. Both content and functionality can be augmented at a later time in a way that places new material at equal status with existing material. Although a custom system could be written for any given document niche, a multivalent framework leverages work done for other multivalent documents. A multivalent perspective is useful for a wide variety of complex documents. We believe that a multivalent document approach to complex digital documents enables new, sophisticated functionality and content for complex digital documents.

Acknowledgments

Gary Kopec contributed to the development of many of these ideas. He is also responsible for the networked XDOC converter. Ginger Ogle implemented the server side of the TileBars search.

8. References

[Andr93] Marc Andreessen. NCSA Mosaic technical summary, May 1993.

[Bagl94] Steven C. Bagley and Gary E. Kopec. Editing images of text. *Communications of the Association for Computing Machinery*, pages 63–72, December 1994.

[Bern92] T.J. Berners-Lee, R. Cailliau, J-F Groff, and B. Pollermann. World-Wide Web: The information universe. In *Electronic Networking: Research, Applications and Policy*, volume 2, pages 52–58. Meckler Publishing, Westport, CT, USA, 1992.

[Bier91] Eric A. Bier. EmbeddedButtons: Documents as user interfaces. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, pages 45–53, November 1991.

[Bier93] Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton, and Tony D. DeRose. Toolglass and Magic Lenses: The see-through interface. In *Proceedings of SIGGRAPH '93*, pages 73–80, August 1993.

[Bore92] Nathaniel S. Borenstein. Computational mail as network infrastructure for computer-supported cooperative work. In *Proceedings of Computer Supported Cooperative Work (CSCW)*, pages 67–74, November 1992.

[Broc95] Kraig Brockschmidt. *Inside OLE 2*. Microsoft Press, Redmond, WA, 1995.

[Brow95] P. Brown and M. Stonebraker. Big sur: A system for the management of earth science data. In *Proceedings of the 20th International Conference on Very Large Databases*, September 1995.

[Buch92] M. Cecelia Buchanan and Polle T. Zellweger. Specifying temporal behavior in hypermedia documents. In *Proceedings of the ACM Conference on Hypertext (ECHT '92)*, pages 262–271, November 30–December 4 1992.

[Davi] Jim Davis. CoNote. http://dri.cornell.edu/pub/davis/annotation.html.

[Gosl95] James Gosling and Henry McGilton. The Java language environment: A white paper, 1995.

[Hear94] Marti A. Hearst. Context and structure in automated full-text information access. Technical Report UCB/CSD 94-836, University of California, Berkeley, 1994.

[Holt84] Barry W. Holtz, Ed. *Back to the Sources: Reading the Classic Jewish Texts*. Summit Books, New York, 1984.

[Kahn94] Robert Kahn and Robert Wilensky. Locating electronic library services and objects: A frame of reference for the cs-tr project. http://WWW.CNRI.Reston.VA.US/home/cstr/handle-intro.html, February 1994.

[Kras88] Glenn E. Krasnet and Stephen T. Pope. A cookbook for using the model-view-controller user interface paradigm in Smalltalk-80. *Journal of Object-Oriented Programming*, pages 26–49, August/September 1988.

[Myka95] Andreas Myka and Ulrich Guntzer. HyperFacs - building and using a digitized paper library. *SIGLINK*, (2), September 1995.

[Nels95] Chris Nelson. OpenDoc and its architecture. *The X Resource*, 1(13):107–126, 1995.

[Nets94] Netscape Communications Corporation. Netscape. Commercial Software, 1994.

[Phel94] Thomas A. Phelps. TkMan: A man born again. *The X Resource*, 1(10):33–46, 1994.

[Rosc95] Martin Roscheisen, Christian Mogensen, and Terry Winograd. Interaction design for shared World-Wide Web annotations. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI' 95)*. The Association for Computing Machinery, May 1995.

[Silv94] Mario J. Silva. Active documentation for VLSI design. Technical Report UCB/CSD 94-843, University of California, Berkeley, 1994.

[Ston91] M. Stonebraker and G. Kemnitz. The postgres next-generation data base system. *Communications of the Association for Computing Machinery*, October 1991.

[Stor92] Guy A. Story, Lawrence O'Gorman, David Fox, Louise Levy Schaper, and H.V. Jagadish. The RightPages image-based electronic library for alerting and browsing. *IEEE Computer*, pages 17–26, September 1992.

[Terr90] Douglas B. Terry and Donald G. Baker. Active Tioga documents. Technical Report CSL-90-6, Xerox Corporation, Palo Alto Research Center, June 1990.

[Walt94] Mark Walter. Acrobat 2.0: Adobe moves up market, beyond ad hoc document delivery. *Seybold Report on Desktop Publishing*, 9(1):3–10, 1994.

[Wile95] Robert Wilensky. UC Berkeley's digital library project. *Communications of the Association for Computing Machinery*, 38(4):60, April 1995.

[Wile96] Robert Wilensky. Toward work-centered digital information services. *IEEE Computer Special Issue on Digital Libraries*, May 1996.