Methods and systems for producing and updating views (405) of information stored in a database (409) are provided. Schemas (413) are utilized to format a view of information in the database. When the database is modified, the schemas (413) are utilized to produce new views of the information in the database (409) and the new views are published for users to access (417). With the invention, the views (405) of information in the database (409) remain current and yet users can access the views (405) without requiring the use of the database (409), thereby allowing it to perform other tasks.
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>Albania</td>
<td>ES</td>
<td>Spain</td>
<td>LS</td>
<td>Lesotho</td>
<td>SI</td>
<td>Slovenia</td>
</tr>
<tr>
<td>AM</td>
<td>Armenia</td>
<td>FI</td>
<td>Finland</td>
<td>LT</td>
<td>Lithuania</td>
<td>SK</td>
<td>Slovakia</td>
</tr>
<tr>
<td>AT</td>
<td>Austria</td>
<td>FR</td>
<td>France</td>
<td>LU</td>
<td>Luxembourg</td>
<td>SN</td>
<td>Senegal</td>
</tr>
<tr>
<td>AU</td>
<td>Australia</td>
<td>GA</td>
<td>Gabon</td>
<td>LV</td>
<td>Latvia</td>
<td>SZ</td>
<td>Swaziland</td>
</tr>
<tr>
<td>AZ</td>
<td>Azerbaijan</td>
<td>GB</td>
<td>United Kingdom</td>
<td>MC</td>
<td>Monaco</td>
<td>TD</td>
<td>Chad</td>
</tr>
<tr>
<td>BA</td>
<td>Bosnia and Herzegovina</td>
<td>GE</td>
<td>Georgia</td>
<td>MD</td>
<td>Republic of Moldova</td>
<td>TG</td>
<td>Togo</td>
</tr>
<tr>
<td>BB</td>
<td>Barbados</td>
<td>GH</td>
<td>Ghana</td>
<td>MG</td>
<td>Madagascar</td>
<td>TJ</td>
<td>Tajikistan</td>
</tr>
<tr>
<td>BE</td>
<td>Belgium</td>
<td>GN</td>
<td>Guinea</td>
<td>MK</td>
<td>The former Yugoslav</td>
<td>TM</td>
<td>Turkmenistan</td>
</tr>
<tr>
<td>BF</td>
<td>Burkina Faso</td>
<td>GR</td>
<td>Greece</td>
<td>ML</td>
<td>Mali</td>
<td>TR</td>
<td>Turkey</td>
</tr>
<tr>
<td>BG</td>
<td>Bulgaria</td>
<td>HU</td>
<td>Hungary</td>
<td>MN</td>
<td>Mongolia</td>
<td>TT</td>
<td>Trinidad and Tobago</td>
</tr>
<tr>
<td>BI</td>
<td>Benin</td>
<td>IE</td>
<td>Ireland</td>
<td>MR</td>
<td>Mauritania</td>
<td>UA</td>
<td>Ukraine</td>
</tr>
<tr>
<td>BR</td>
<td>Brazil</td>
<td>IL</td>
<td>Israel</td>
<td>MW</td>
<td>Malawi</td>
<td>UG</td>
<td>Uganda</td>
</tr>
<tr>
<td>BY</td>
<td>Belarus</td>
<td>IS</td>
<td>Iceland</td>
<td>NA</td>
<td>Netherlands</td>
<td>US</td>
<td>United States of America</td>
</tr>
<tr>
<td>CA</td>
<td>Canada</td>
<td>IT</td>
<td>Italy</td>
<td>NE</td>
<td>Niger</td>
<td>VN</td>
<td>Viet Nam</td>
</tr>
<tr>
<td>CF</td>
<td>Central African Republic</td>
<td>JP</td>
<td>Japan</td>
<td>NL</td>
<td>Netherlands</td>
<td>YU</td>
<td>Yugoslavia</td>
</tr>
<tr>
<td>CG</td>
<td>Congo</td>
<td>KE</td>
<td>Kenya</td>
<td>NO</td>
<td>Norway</td>
<td>ZW</td>
<td>Zimbabwe</td>
</tr>
<tr>
<td>CH</td>
<td>Switzerland</td>
<td>KG</td>
<td>Kyrgyzstan</td>
<td>NZ</td>
<td>New Zealand</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CI</td>
<td>Côte d'Ivoire</td>
<td>KP</td>
<td>Democratic People’s</td>
<td>PL</td>
<td>Poland</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CM</td>
<td>Cameroon</td>
<td>KR</td>
<td>Republic of Korea</td>
<td>PT</td>
<td>Portugal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CN</td>
<td>China</td>
<td>KZ</td>
<td>Kazakhstan</td>
<td>RO</td>
<td>Romania</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CU</td>
<td>Cuba</td>
<td>LC</td>
<td>Saint Lucia</td>
<td>RU</td>
<td>Russian Federation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CZ</td>
<td>Czech Republic</td>
<td>LI</td>
<td>Liechtenstein</td>
<td>SD</td>
<td>Sudan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DE</td>
<td>Germany</td>
<td>LK</td>
<td>Sri Lanka</td>
<td>SE</td>
<td>Sweden</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DK</td>
<td>Denmark</td>
<td>LR</td>
<td>Liberia</td>
<td>SG</td>
<td>Singapore</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EE</td>
<td>Estonia</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.
STATIC VIEWS OF DATA BASES

BACKGROUND OF THE INVENTION

The present invention relates to information storage systems. More specifically, the invention relates to producing and updating hyper-linked views of information stored in a database.

Databases are used to store information (or data) in an organized way so that the information can be retrieved and viewed very quickly and reliably. Accordingly, databases are very efficient at storing and retrieving data. To achieve this efficiency in handling data, databases keep the information structured in a way that is efficient for the hardware characteristics of the systems on which they run. Relational database management systems (RDBMSs) keep the data generally organized in tables and indexes. When users query the database, the data is retrieved from tables, formatted and presented to the user in suitable formats for display or printing.

Databases contains two basic kinds of information: content information (e.g., tables) and structural information (e.g., indexes). Content information is data such as names, addresses, documents, descriptions, attributes, and the like. Structural information is the information needed by the database to keep the content information organized in order to optimize the combination of storage space used and performance obtained. The way tables and indexes are maintained in the database is often very different from how users view the information. For example, a database can store in a table a list of names and addresses and in another table a list of products with their description and price. A user working in the sales department can query the database to get the list of people who bought a certain product. In this case, the customer would view on his terminal the data associated to the products and the data associated to individuals. The software that formats the data in views and presents them to the user is the database application.
Every time users want to query the database to obtain information, they need to execute a database application that queries the database and formats the information using certain rules called schemas. Schemas (or templates) can be embedded in the application or can be stored in the database itself and represent the empty (data-less) structure of the view that is presented to the users.

The preparation of views by the application software is performed every time users query the database to obtain views. Even when many users request the same view at the same time, the application software queries the database and prepares the same view as many times as the number of users. In this case, the application executes the same operation producing the same result many times simultaneously, unnecessarily using up the resources of the system.

FIG. 1 shows a process of creating a view of information in a database utilizing a schema. A database 101 includes a customer table 103 and a table product 105. The database record in customer table 103 includes an index to the first database record in product table 105. A sales schema 107 specifies the format of a view of information that a user may request.

Dummy values are shown in sales schema 107 to indicate that these values will be filled when the view is created.

A view 109 shows the view generated utilizing sales schema 107. As shown, the user is able to see what products a specific customer has purchased.

A common solution to the problem of repeating the same operations of querying the database and producing the same view is partially solved by the use of caching mechanisms. A cache is a temporary replication of part of the data contained in the primary storage (databases in our case) in a faster secondary storage area (generally random access memory or a hard disk). Cache systems are used in database applications to store in memory or in the file system, the most recent table accessed in the database and views produced by the application in order to avoid the redundant execution of operations needed to create a view already prepared for subsequent requests.

FIG. 2 shows the use of a cache to store views of information in a database. A user terminal 201 issues a request for a view. A database application 203 checks to see if the
requested view is already in a cache 211. The cache stores multiple recently requested views 213. If the requested view is in cache 211, the database application then checks to see if the cached view is invalid or out of date. For example, the database application can check timestamps to determine if the tables utilized to make this view have changed since the view was generated.

If the requested view is not in cache 211 or it is out of date, database application 203 retrieves data from a database 205. Database 205 stores multiple tables 207. The database application retrieves a schema 209 and prepares the view according to the data and schema. The view is then sent to the user and stored in cache 211 for subsequent access.

The use of a cache can save the application the time for executing several steps if the view was already prepared and is still available in the cache. Even though the use of a cache can certainly improve the performance (throughput) of an information system, caching has some limitations. An application benefits from a cache only if the view requested by the user was recently prepared for this or another user. Also, since caches are expensive, they are usually limited in capacity and can hold only a fixed number of views. When the cache is full, the oldest views in the cache are removed and replaced with new ones. Therefore, the probability of finding a view in the cache is limited by the size of the cache in relation with the size of the database.

Additionally, when data changes in the database, the affected views become invalid or out of date. Although software may be utilized to continually check for invalid views and refreshing them when required, such software represents significant overhead that reduces the performance of the system.

Accordingly, there is a need for innovative methods and systems for producing and updating views of information stored in a database. It would be desirable to have a system where views are accessible to users without adversely affecting the performance of the database system.
SUMMARY OF THE INVENTION

Embodiments of the present invention provide methods and systems for producing and updating views of information stored in a database. Schemas (or templates) are utilized to format a view of information in the database. When the database is modified, the schemas are utilized to produce new views of the information in the database and the new views are published for users to access. With the invention, the views of information in the database remain current and yet users can access the views without requiring the use of the database and, possibly more important, without requiring the database application to check if a view is both in a cache and still valid, which can be a very time consuming process. Accordingly, the database application can have more processing time for other tasks. Several embodiments of the invention are described below.

In one embodiment, the invention provides a method of accessing information in a database. A request to modify a record in the database is received and a view is identified that includes information in the record. A new view is generated that includes information in the modified record, where the new view is specified by a schema that includes at least one tag corresponding to information in the database. In a preferred embodiment, the record stores information about a document and the view is a hyper-text representation of the document.

In another embodiment, the invention provides a method of accessing information in a database. A request to modify a record in the database is received and it is determined if the request changes a view, where the view is specified by a schema that includes at least one tag corresponding to information in the database. A new view is generated that includes information in the modified record. It can be determined the request changes a view by analyzing the request to identify a tag that corresponds to new information in the request and checking the schema for the identified tag.

In another embodiment, the invention provides a system for viewing information stored in a database. The system includes a view of a document and a schema that specifies the view including at least one tag corresponding to information about the document. A database stores
records, where a record includes the information about the document, a pointer to the view and a pointer to the schema. An information manager receives requests to modify the record and generates a new view utilizing the schema so that the new view includes information in the modified record.

Other features and advantages of the invention will become readily apparent upon review of the following description in association with the accompanying drawings.
BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 shows a process of creating a view of information in a database utilizing a schema.

FIG. 2 shows the use of a cache to store views of information in a database.

FIG. 3 illustrates an example of a computer system that can be utilized to execute the software of an embodiment of the invention.

FIG. 4 illustrates a system block diagram of the computer system of FIG. 3.

FIG. 5 shows a block diagram of a system that publishes static views of documents or folders for user access.

FIG. 6 shows a relationship of a document, database record, views, and schemas.

FIG. 7 shows a flow chart of a process of adding a new document (or folder) to a database.

FIG. 8 shows a flow chart of a process of modifying the database including the generation of new views if the views should reflect a change in the database.

FIG. 9 shows a flow chart of a process of determining if views change as a result of the modification of the database.

FIG. 10 shows a flow chart of a process of generating a new view of information in the database.

FIG. 11 shows a flow chart of another process of modifying the database including the generation of new views.

FIG. 12 shows an example of a database record.

FIG. 13 shows a portion of a schema in hyper-text markup language (HTML) that specifies a view of a folder.

FIG. 14 shows a static view of the folder.

FIG. 15 shows a page that may be utilized to modify the information about the folder.

FIG. 16 shows how the page may be utilized to modify the title and abstract of the folder.
FIG. 17 shows the database record of FIG. 12 that has been modified as specified in FIG. 16.

FIG. 18 shows a new static view of the folder including the modifications.
DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

In the description that follows, the present invention will be described in reference to embodiments that produce and update views of documents and folders stored in a database. The views are produced according to schemas, where both the view and schema are in the hyper-text markup language (HTML). However, the invention is not limited to any particular language, computer architecture or specific implementation. Therefore, the description of the embodiments that follows is for purposes of illustration and not limitation.

FIG. 3 illustrates an example of a computer system that can be used to execute the software of an embodiment of the invention. FIG. 3 shows a computer system 301 that includes a display 303, screen 305, cabinet 307, keyboard 309, and mouse 311. Mouse 311 can have one or more buttons for interacting with a graphical user interface. Cabinet 307 houses a CD-ROM drive 313, system memory and a hard drive (see FIG. 4) which can be utilized to store and retrieve software programs incorporating computer code that implements the invention, data for use with the invention, and the like. Although CD-ROM 315 is shown as an exemplary computer readable storage medium, other computer readable storage media including floppy disk, tape, flash memory, system memory, and hard drive can be utilized. Additionally, a data signal embodied in a carrier wave (e.g., in a network including the Internet) can be the computer readable storage medium.

FIG. 4 shows a system block diagram of computer system 301 used to execute the software of an embodiment of the invention. As in FIG. 3, computer system 301 includes monitor 303 and keyboard 309, and mouse 311. Computer system 301 further includes subsystems such as a central processor 351, system memory 353, fixed storage 355 (e.g., hard drive), removable storage 57 (e.g., CD-ROM drive), display adapter 359, sound card 361, speakers 363, and network interface 365. Other computer systems suitable for use with the invention can include additional or fewer subsystems. For example, another computer system could include more than one processor 351 (i.e., a multi-processor system) or a cache memory.
The system bus architecture of computer system 301 is represented by arrows 367. However, these arrows are illustrative of any interconnection scheme serving to link the subsystems. For example, a local bus could be utilized to connect the central processor to the system memory and display adapter. Computer system 301 shown in FIG. 4 is but an example of a computer system suitable for use with the invention. Other computer architectures having different configurations of subsystems can also be utilized.

Embodiments of the invention provide systems and methods for producing and updating views of stored information. In preferred embodiments, the information is in the form of documents or folders stored in a file system. However, the invention may be advantageously applied to any stored information so it is not limited by the specific embodiments described herein. A "view" of a document is the image of the document the user sees. The invention provides what is called "static views" because the views do not change when accessed by the user (as in caching), but only upon modification of the database.

FIG. 5 shows a block diagram of a system that publishes static views of documents or folders for user access. A user can operate a browser 401 on a computer system that is in communication with a Web server 403. The browser typically receives HTML files via the hyper-text transport protocol (HTTP) and displays the documents on the screen.

Web server 403 can access multiple views 405. The views are hyper-text documents that include information about (e.g., meta-information) or contained in documents or folders that are stored in the database. Views 405 are in HTML and the user specifies a desired view via a uniform resource locator (URL). Although views 405 are shown with a couple of hyper-links 406 for illustration purposes, any number of hyper-links can be used. When a user requests a view, Web server 403 accesses the appropriate view 405 and sends the HTML file to the user for display by browser 401. The browser and Web server may operate on the same computer system, be on the same intranet or operate on a wide area network (WAN) such as the Internet.

Documents and folders that users may desire to access are stored in a file system 407. A folder is simply a container of documents and the file system can store the documents and
folders in any manner that is known in the art. The description that follows will concentrate the
discussion on stored documents. However, folders can and are managed in a similar manner.
A database 409 stores information about the documents in file system 407. As will be
described in more detail in reference to FIG. 6, database 409 stores a record that includes
information about each document in file system 407 that can be requested by a user. A
“database” is a repository of information and in FIG. 5, both file system 407 and database 409
provide a repository of information so they can be thought of as a single “database.” Although
the file system is utilized for storing documents in preferred embodiments, the contents of the
documents can also be stored in database 409.

At the center of the system shown in FIG. 5 is an information manager 411. The
information manager acts as the intermediary between Web server 403 that is responsible for
providing views of documents and database 409 that is responsible for storing information
about documents. In order to produce views of documents, information manager 411 has access
to one or more schemas 413. A schema is a template of how the information about a document
should be displayed. For example, a schema could specify that the information about a
document should be shown in condensed form or a schema could specify a more comprehensive
form. Schemas can also be directed to different artistic styles of presenting the information.

A document can get stored in the database through Web server 403. Assume a user has
requested to store a document. Information manager 411 receives the document and can ask the
user specific questions about the document in order to determine the meta-information for the
document. For example, the user can be asked to enter the author, title, category, keywords,
creation date, links to other documents, and the like about the document, which will be
generally referred to as “meta-information.” Additionally, the user can specify the schema that
will be utilized to view the document. If no schema is specified, a default schema can be
utilized.

Information manager 411 stores the document in file system 407 and the information
about the document in a record in database 409. The information manager then generates a
view of the document utilizing the specified schema. Once the view is produced, the
information manager publishes the view as indicated by an arrow 415. Once a view is published, the view can be accessed by a user or users through browser 401 via Web server 403 as described above. The information manager is typically on the same computer system as file system 407, database 409 and schemas 413. However, there is no requirement that any of the components of FIG. 5 reside on the same computer system so the components can be distributed between and among computer systems in different ways depending on the application.

As mentioned earlier, a user can add a document to the database through browser 401. The user can also modify the document or information about the document through the browser. In some embodiments, a direct user access 417 is provided that allows users to accomplish the same or similar tasks directly. For example, in an environment where users interact with information manager 411 over a network via browser 401, a system administrator can have the ability to interact directly with the information manager via direct user interface 417 such as a custom Windows application.

Now that one embodiment of an overall system has been described, details of the underlying data structures will be described. FIG. 6 shows details of a document, database record, views, and schemas. A document 451 typically includes text and hyper-links, but it can also include graphics and sounds depending on the application. For example, document 451 can be generated by a conventional word processor. In reference to FIG. 5, document 451 can be stored in file system 407.

A database record 453 includes information about document 451. Record 453 includes meta-information 455 about the document. As shown, the meta-information may include fields indicating the author, title, category, creation date, and the like of the document. Record 453 includes a field that is a link or pointer to document 451. The pointer is typically implemented as a path and file name of the document in the file system, but this is not required.

Record 453 also includes zero one or more field pairs of schema/view pointers. Pointers 459 reference views 461 and pointers 463 reference schemas 465. In each schema/view pair, the schema is the template for producing the view. Pointers 459 and 463 can be implement as
URLs or any other mechanism. In reference to FIG. 5, record 453 can be stored in database 409.

In preferred embodiments, both views 461 and schemas 465 are HTML files. The schemas include tags (e.g., an author tag) that correspond to information about the document. When the view is produced, the information corresponding to the tags in the schema are retrieved from the database and inserted into the schema to produce the desired view. Both the views and schemas can have hyper-links to other documents.

FIG. 7 shows a flow chart of a process of adding a new document to the database. At a step 501, the information manager receives a new document. The new document may be received by the information manager from a user that wishes to add the document to the database. The document is stored in the file system at a step 503. The path and file name of the document is stored as a pointer to the stored document.

At a step 505, meta-information about the document is received. The meta-information can be obtained by providing the user with a form to fill out that includes the desired meta-information. The meta-information will be used in the views of the document to enhance a user's understanding of the document. In other embodiments, some or all of the meta-information is obtained automatically by analyzing the document.

A user selects a schema that specifies the desired view for the document at a step 507. The schemas can present a variety of different formats in which the document can be viewed.

The user may select a schema from a list and in other embodiments, a view utilizing a selected schema is generated for the user to see and approve. The user may select one or more schemas in which the document may be viewed. Additionally, there may be a default schema so if the user does not select a schema, the default will be utilized.

At a step 509, the information manager generates the view or views of the document and its meta-information utilizing the selected schema or schemas. The view is published at a step 511, which allows users access to the view. At a step 513, the information manager saves the meta-information in a record in the database. The record can also include a pointer to the
document in the file system and pointers to the views and schemas for this document. The related indexes in the database may also need to be updated to reflect the new record.

Once a view of a document has been published, users can access the view. More importantly, the users can access the view without requiring access to the database. When many users are sending requests to a conventional database, the performance of the database application can degrade quickly. With the invention, the users are able to view documents in the database without taxing the resources of the database application. Although the views can be called “static views” because they are not updated like prior art caches, the views are nonetheless updated when the database is modified.

FIG. 8 shows a flow chart of a process of modifying the database including the generation of new views if the views should reflect a change in the database. At a step 501, the information manager receives a request to modify a record in the database. The request can be to modify the meta-information, provide a revised version of the document or to change a schema that produces a view. The request to modify the record in the database is fulfilled at a step 553. Preferably, the request to modify the record is fulfilled first (i.e., before any views are modified). This is because once the database is updated, the views and other information can be updated at a later date if necessary. For example, after the computer system has recovered from a crash.

At a step 555, the information manager determines if the request changes a view specified by schema. In general, the information manager analyzes the schemas to determine if the schemas have a tag that corresponds to information that was modified by the request. The process of determining if views changed will be described in more detail in reference to FIG. 9.

If it is determined that one or more views changed at a step 557, the information manager generates a new view for each view that changes at a step 559. The process of generating new views will be described in more detail in reference to FIG. 10.

Once the one or more views are generated, the new view is published at a step 561. The new view can be posted by uploading it to the Web server that manages access to the views.
FIG. 9 shows a flow chart of a process of determining if views change as a result of the modification of the database. At a step 601, the information manager analyzes the request to modify a record in the database to identify tags corresponding to new information in the request. For example, if the request modifies the title of a document, the title tag would be identified.

For each schema that is utilized by the document, the information manager checks the schema for the identified tag or tags at a step 603. The schemas may be identified by the pointers in the database record associated with the document (see FIG. 6). Continuing with the simple example where the title of the document was changed, the information manager analyzes the schemas utilized to view the document and finds each schema that includes the title tag.

If an identified tag is found in a schema, the information manager determines that the request changes the view specified by the schema at a step 605. The new view is generated at step 559 of FIG. 8 and can be generated according to a process shown in FIG. 10.

FIG. 10 shows a flow chart of a process of generating a new view of information in the database. At a step 651, the information manager retrieves a schema. For each tag in the schema, the information manager retrieves the information corresponding to the tag from the database at a step 653. The tags are not required to be identical to field names in the records of the database. For example, a tag in HTML may need to be in a certain format, which is very different than the format utilized as fields in the database. Any number of ways known in the art may be utilized to match up tags and fields in the database records. In preferred embodiments, a table is produced that provides the translation between tags and fields in the database records.

At a step 655, the information manager inserts the information in the schema in place of the one or more tags to produce the new view. The tags serve as placeholders for information about the document. Thus, when a view is created, the information specified by the tags is substituted for the tags.

The above has described a process where the information manager identifies views that change when the database is modified. Although this has the advantage that only the views that need to be changed are changed, depending on the application, it may be too performance
intensive to identify the views that change so it may be beneficial to just generate all new views for the document.

FIG. 11 shows a flow chart of another process of modifying the database including the generation of all new views for the document. At a step 701, the information manager receives a request to modify a record in the database. The information manager fulfills the request at a step 703. Steps 701 and 703 are the same steps as steps 551 and 553 in FIG. 8.

At a step 705, the information manager generates all new views for the document. The information manager utilizes each schema specified in the database record for the document and generates a new view for each schema. The new views may be produced according to the process in FIG. 10. Once the new views are generated, the new views are published at a step 707. Publishing the views can entail uploading the new views to URLs specified in the database record.

The above has described how the invention provides and updates users’ views of documents and meta-information stored in a database. However, the invention relates to visualizing documents so it may be beneficial to see a simple example of the invention. FIGS. 12-18 will be utilized to show an example of an embodiment of the invention.

FIG. 12 shows an example of a database record. A database record 801 is shown that includes multiple fields. A title field 803 includes the title of a folder, which is shown as “Title of this Page.” An abstract 805 includes the abstract of the folder, which is shown as “This is the Abstract of this Page.” A schema field 807 indicates the schema to produce the view of the folder, which is shown here a number that indexes the schema. Lastly, a mapping field 809 and a directory field 811 are related to the location of the view produced utilizing the schema in field 807. The mapping and directory fields are concatenated together to yield part of the URL of the view. For simplicity, the database is not shown with a pointer to the folder on the file system.

FIG. 13 shows a portion of a schema in HTML that specifies a view of a folder. A schema 851 is an HTML file that specifies the view of the folder. Tags 853 will be utilized within the schema to specify locations where information from the database should be inserted
to generate a view. As mentioned earlier, the tags may not be identical to the fields of the database record so a table may be utilized to match up corresponding tags and fields. For example, a table may be as follows:

<table>
<thead>
<tr>
<th>Field</th>
<th>Tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>szTitle</td>
<td>##WF_FolderTitle##</td>
</tr>
<tr>
<td>szAbstract</td>
<td>##WF_FolderAbstract##</td>
</tr>
</tbody>
</table>

A table is not required and many other mechanisms can be utilized.

FIG. 14 shows a static view of the folder. A view 901 was produced utilizing database record 801 and schema 851. The title from title field 803 was inserted at location 903 in the view and the abstract from abstract field 805 was inserted at location 905. Mapping field 809 and directory field 811 were utilized to produce URL 907 that identifies view 901. The view may be produced by the process described in reference to FIG. 10.

View 901 also includes hyper-links 909, 911, 913, 915, and 917. Some hyper-links in views come directly from the document or folder. These hyper-links were present or "hard-coded" into schema 851. Edit hyper-link 919 may be utilized to edit the database as will be describe in more detail in reference to the next figure.

FIG. 15 shows a page that may be utilized to modify the information about the folder. If the user activates edit hyper-link 919, a page 951 is displayed. Page 951 is an HTML file that includes information in database record 801. Most notably, title field 803 and abstract field 805 are shown at locations 953 and 955, respectively. The user may then edit these fields by changing the information at these locations.

FIG. 16 shows how the page may be utilized to modify the title and abstract of the folder. A page 1001 is the same as page 951 except that the user has changed the title to "NEW TITLE of this Page" and the abstract to "This is the NEW Abstract of this Page." as indicated at locations 1003 and 1005, respectively.

FIG. 17 shows the database record of FIG. 12 that has been modified as specified in FIG. 16. A database record 1051 has a new title in a title field 1053 and a new abstract in an
abstract field 1055. FIG. 18 shows a new static view of the folder including the modifications. When a new view of the folder is generated, a view 1101 is produced. View 1101 includes the new title and abstract at locations 1103 and 1005, respectively. The remainder of the view remains unchanged.

Although the example is very simple for illustration purposes, it illustrates that the invention utilizes schemas to produce views of documents or folders stored in a database. As information in the database changes, new views are generated. Users may access the views without requiring the database application to search for and provide the desired document or folder, which allows the database resources to be utilized on other tasks. Additionally, the database application does not need to check the status of the cache continually as in prior art methods.

The hyper-links in the views allow users to access other information in the database without actually accessing the database. This can be especially desirable when the number of accesses to the information in the database is far greater than the number of updates to the information. With prior art techniques such as caching, the system continually checks to see if the cache is up to date. The views in preferred embodiments are formatted and presented using a hyper-text language such as HTML, portable document format (PDF), FrameMaker, and the like.

While the above is a complete description of preferred embodiments of the invention, various alternatives, modifications, and equivalents can be used. It should be evident that the invention is equally applicable by making appropriate modifications to the embodiments described above. For example, the invention is not limited to documents and folders but may be applied to any type of information including images, sales data, financial information, employee data, and the like. Therefore, the above description should not be taken as limiting the scope of the invention that is defined by the metes and bounds of the appended claims along with their full scope of equivalents.
CLAIMS

1. A method of viewing information stored in a database, comprising:
   receiving a request to modify a record in the database;
   identifying a view that includes information in the record; and
   generating a new view that includes information in the modified record, the new view
   being specified by a schema that includes at least one tag corresponding to information in the
   database.

2. The method of claim 1, wherein the generating a new view comprises:
   for each tag in the schema, retrieving the information corresponding to the tag from the
   database; and
   inserting the information in the schema in place of the at least one tag to produce the
   new view.

3. The method of claim 1, further comprising fulfilling the request to modify the
   record in the database.

4. The method of claim 1, wherein the information in the record includes
   information regarding a document or folder.

5. The method of claim 1, wherein the identifying a view compromises utilizing a
   pointer to the view in the record.

6. The method of claim 1, wherein the schema that specifies the new view is
   identified by utilizing a pointer to the schema in the record.

7. The method of claim 1, wherein there are multiple views that include information
   in the record, each view being specified by a schema.

8. The method of claim 1, further comprising publishing the new view.

9. The method of claim 1, wherein the new view is in hyper-text markup language
   (HTML).

10. The method of claim 1, wherein the at least one schema is in hyper-text markup
    language (HTML).
11. A computer program product for viewing information stored in a database, comprising:
   computer code that receives a request to modify a record in the database;
   computer code that identifies a view that includes information in the record;
   computer code that generates a new view that includes information in the modified
   record, the new view being specified by a schema that includes at least one tag corresponding to
   information in the database; and
   a computer readable medium that stores the computer codes.

12. The computer program product of claim 11, wherein the computer readable
    medium is selected from the group consisting of CD-ROM, floppy disk, tape, flash memory,
    system memory, hard drive, and data signal embodied in a carrier wave.

13. A method of viewing information stored in a database, comprising:
    receiving a request to modify a record in the database;
    determining if the request changes a view, the view being specified by a schema that
    includes at least one tag corresponding to information in the database; and
    generating a new view specified by the schema that includes information in the modified
    record.

14. The method of claim 13, wherein the generating a new view comprises:
    for each tag in the schema, retrieving the information corresponding to the tag from the
    database; and
    inserting the information in the schema in place of the at least one tag to produce the
    new view.

15. The method of claim 13, wherein the determining if the request changes a view
    comprises:
    analyzing the request to modify a record in the database to identify a tag that
    corresponds to new information in the request;
    checking the schema for the identified tag; and
    if the identified tag is found in a schema, the request is determined to change the view
    specified by the schema.

16. The method of claim 13, further comprising fulfilling the request to modify the
    record in the database.
17. The method of claim 13, wherein the information in the record includes information regarding a document or folder.

18. The method of claim 13, wherein the record includes a pointer to the view.

19. The method of claim 13, wherein the record includes a pointer to the schema.

20. The method of claim 13, wherein there are multiple views that include information in the record, each view being specified by a schema.

21. The method of claim 13, further comprising publishing the new view.

22. A computer program product for viewing information stored in a database, comprising:

- computer code that receives a request to modify a record in the database;
- computer code that determines if the request changes a view, the view being specified by a schema that includes at least one tag corresponding to information in the database;
- computer code that generates a new view specified by the schema that includes information in the modified record; and
- a computer readable medium that stores the computer codes.

23. The computer program product of claim 22, wherein the computer readable medium is selected from the group consisting of CD-ROM, floppy disk, tape, flash memory, system memory, hard drive, and data signal embodied in a carrier wave.

24. A system for viewing information stored in a database, comprising:

- a view of a document;
- a schema that specifies the view and includes at least one tag corresponding to information about the document;
- a database that stores records, wherein a record includes the information about the document, a pointer to the view and a pointer to the schema; and
- an information manager that receives requests to modify the record and generates a new view utilizing the schema so that the new view includes information in the modified record.

25. The system of claim 24, further comprising a plurality of views.
26. The system of claim 25, wherein the plurality of views are in hyper-text markup language (HTML)

27. The system of claim 26, further comprising a Web server that manages access to the view plurality of views.

28. The system of claim 24, further comprising a file system that stores the document.

29. The system of claim 24, further including a view of a folder that is stored in the file system.
Creation process of a view

FIG. 1
FIG. 5
ADDING A NEW DOCUMENT (OR FOLDER)

RECEIVE NEW DOCUMENT

STORE THE DOCUMENT IN THE FILE SYSTEM

RECEIVE META-INFORMATION ABOUT THE DOCUMENT

SELECT A SCHEMA THAT SPECIFIES THE DESIRED VIEW

GENERATE THE VIEW UTILIZING THE SCHEMA

PUBLISH THE VIEW

SAVE THE META-INFORMATION IN A RECORD IN THE DATABASE

DONE

FIG. 7
MODIFYING THE DATABASE

RECEIVE A REQUEST TO MODIFY A RECORD IN THE DATABASE

FULFILL THE REQUEST TO MODIFY THE RECORD IN THE DATABASE

DETERMINE IF THE REQUEST CHANGES A VIEW SPECIFIED BY A SCHEMA

VIEW(S) CHANGE?

YES

FOR EACH VIEW THAT CHANGES, GENERATE A NEW VIEW SPECIFIED BY THE SCHEMA

NO

PUBLISH THE NEW VIEW(S)

DONE

FIG. 8
Determining if views changed:

1. Analyze the request to modify a record in the database to identify tag(s) corresponding to new information in the request.

2. For each schema, check the schema for the identified tag(s).

3. If an identified tag is found in a schema, the request changes the view produced by the schema.

Done.
GENERATING A NEW VIEW

651

RETRIEVE A SCHEMA

FOR EACH TAG IN THE SCHEMA, RETRIEVE INFORMATION CORRESPONDING TO THE TAG FROM THE DATABASE

653

655

INSERT THE INFORMATION IN THE SCHEMA IN PLACE OF THE TAG(S) TO PRODUCE THE NEW VIEW

DONE

FIG. 10
MODIFYING THE DATABASE

701

RECEIVE A REQUEST TO MODIFY A RECORD IN THE DATABASE

703

FULFILL THE REQUEST TO MODIFY THE RECORD IN THE DATABASE

705

GENERATE ALL NEW VIEWS

707

PUBLISH THE NEW VIEWS

DONE

FIG. 11
FIG. 12

FIG. 17
FIG. 13
Title of this Page

This is the Abstract of this Page.

This is the Content of this Page.

This is the Content of this Page.

This is the Content of this Page.

This is the Content of this Page.

.....TELEKOM 8.900 -0.300 2506 .....S DARBY 3.200 -0.100 2420 .....PROMET 0

Keywords: Keyword1 Keyword2 Keyword3 Keyword4

Powered by Infosquare ContentServer (C)
Last Update Wed May 27 09:24:01 1998

FIG. 14
FIG. 15
FIG. 16
NEW TITLE of this Page

This is the NEW Abstract of this Page.

This is the Content of this Page.
This is the Content of this Page.
This is the Content of this Page.
This is the Content of this Page.

NK 11.000 -0.100 9553 .... REKAPAC 0.385 -0.005 3937 .....AMSTEEL 0.420 -0. 

Keywords: Keyword1 Keyword2 Keyword3 Keyword4

FIG. 18
INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/14581

A. CLASSIFICATION OF SUBJECT MATTER
IPC(6): G06F 17/30, 15/00, 19/00,
US CL: 707/1; 395/762; 345/335
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED
Minimum documentation searched (classification system followed by classification symbols)
U.S.: 707/1; 395/762; 345/335

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
NONE

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
APS, IEEE, Dialog

C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y, P</td>
<td>US 5,737,591 A (KAPLAN ET AL) 07 APRIL 1998, COL. 2, LINE 60-COL. 3, LINE 40.</td>
<td>1-29</td>
</tr>
<tr>
<td>Y</td>
<td>US 5,276,870 A (SHAN ET AL) 04 JANUARY 1994, COL. 3, LINES 25-82.</td>
<td>1-29</td>
</tr>
<tr>
<td>Y</td>
<td>US 5,544,358 A (CAPPS ET AL) 06 AUGUST 1996, COL. 2, LINES 7-55.</td>
<td>1-29</td>
</tr>
<tr>
<td>Y</td>
<td>US 5,446,882 A (CAPPS ET AL) 29 AUGUST 1995, COL. 2, LINES 5-51.</td>
<td>1-29</td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of Box C. See patent family annex.

Date of the actual completion of the international search
21 SEPTEMBER 1998

Date of mailing of the international search report
09 NOV 1998

Name and mailing address of the ISA/US Commissioner of Patents and Trademarks
Box PCT
Washington, D.C. 20231
Facsimile No. (703) 305-3230

Authorized officer
FRANTZ COBY

Telephone No. (703) 305-9707

Form PCT/ISA/210 (second sheet) (July 1992)
<table>
<thead>
<tr>
<th>Category*</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
</table>