Materialized Hypertext Views

Giuseppe Sindoni
ISTAT - National Institute of Statistics, Italy

INTRODUCTION

A materialized hypertext view can be defined as "a hypertext containing data coming from a database and whose pages are stored in files" (Sindoni, 1999). A Web site presenting data from a data warehouse is an example of such a view. Even if the most popular approach to the generation of such sites is based on dynamic Web pages, a rationale for the materialized approach has produced many research efforts. The topic will cover logical models to describe the structure of the hypertext.

BACKGROUND

Hypertext documents in the Web are in essence collections of HTML (HyperText Markup Language) or XML (the eXtensible Markup Language) files and are delivered to users by an HTTP (HyperText Transfer Protocol) server. Hypertexts are very often used to publish very large amounts of data on the Web, in what are known as *data intensive Web sites*. These sites are characterized by a large number of pages sharing the same structure, such as in a University Web site, where there are numerous pages containing staff information, that is, "Name," "Position," "Department," and so on. Each staff page is different, but they all share the types of information and their logical organization. A group of pages sharing the same structure is called *page class*. Similarly to databases, where it is possible to distinguish between the intensional (database structure) and extensional (the database records) levels, in data intensive Web sites it is possible to distinguish between the site structure (the structure of the different page classes and page links) and site pages (instances of page classes).

Pages of a data intensive site may be generated *dynamically*, that is, *on demand*, or be *materialized*, as will be clarified in the following. In both approaches, each page corresponds to an HTML file and the published data normally come from a database, where they can be updated more efficiently than in the hypertext files themselves. The database is queried to extract records relevant to the hypertext being generated and page instances are filled with values according to a suitable hypertext model describing page classes (Agosti et al., 1995; Aguilera et al., 2002; Beeri et al., 1998; Crestani & Melucci, 2003; Baresi et al., 2000; Balasubramanian et al., 2001; Merialdo et al., 2003; Rossi & Schwabe, 2002; Simeon & Cluet, 1998). The hypertext can then be accessed by any Internet-enabled machine running a Web browser.

In such a framework, hypertext can be regarded as database views, but in contrast with classic databases, such as relational databases, the model describing the view cannot be the same as the one describing the database storing the published data.

The most relevant hypertext logical models proposed so far can be classified into three major groups, according to the purpose of the hypertext being modeled. Some approaches are aimed at building hypertext as integration views of distributed data sources (Aguilera et al., 2002; Beeri et al., 1998; Simeon & Cluet, 1998), other as views of an underlying local database (Fernandez et al., 2000; Merialdo et al., 2003). There are also proposals for models and methods to build hypertext independently from the source of the data that they publish (Agosti et al., 1995; Baresi et al., 2000; Balasubramanian et al., 2001; Rossi & Schwabe, 2002; Simeon & Cluet, 1998).

Models can be based on graphs (Fernandez et al., 2000; Simeon & Cluet, 1998), on XML Data Type Definitions (Aguilera et al., 2002), extension of the Entity Relationship model (Balasubramanian et al., 2001), logic rules (Beeri et al., 1998) or object-like paradigms (Merialdo et al., 2003; Rossi & Schwabe, 2002).

MAIN THRUST

The most common way to automatically generate a derived hypertext is based principally on the dynamic construction of virtual pages following a client request. Usually, the request is managed by a specific program (for example a Common Gateway Interface – CGI – called as a link in HTML files) or described using a specific query language, whose statements are embedded into pages. These pages are often called "pull pages," because it is up to the client browser to pull out the interesting information. Unfortunately, this approach has some major drawbacks:

- it involves a degree of Data Base Management System overloading, because every time a page is

requested by a client browser, a query is issued to the database in order to extract the relevant data;
- it introduces some platform-dependence, because the embedded queries are usually written in a proprietary language and the CGIs must be compiled on the specific platform;
- it hampers site mirroring, because if the site needs to be moved to another server, either the database needs to be replicated, or some network overload is introduced due to remote queries;
- it doesn't allow the publication of some site metadata, more specifically information about the structure of the site, which may be very useful to querying applications.

An alternative approach is based on the concept of *materialized hypertext view*: a derived hypertext whose pages are actually stored by the system on a server or directly on the client machine, using a mark-up language like HTML. This approach overcomes the above disadvantages because: *(i)* pages are static, so the HTTP server can work on its own; *(ii)* there is no need to embed queries or script calls in the pages, as standard sites are generated; *(iii)* due to their standardization, sites can be mirrored more easily, as they are not tied to a specific technology; and finally, *(iv)* metadata can be published by either embedding them into HTML comments or directly generating XML files.

A data model, preferably object-oriented, is used to describe a Web hypertext. This allows the system to manage nested objects by decomposition. This means that each hypertext page is seen as an object with attributes, which can be atomic or complex, such as a list of values. Complex attributes are also modeled as nested objects into the page object. These objects can also have both atomic and complex attributes (objects) and the nesting mechanism is virtually unlimited.

Below we will describe the Araneus Data Model (ADM) (Merialdo et al., 2003), as an example of a hypertext data model. Different models can be found in (Fernandez et al., 2000; Fraternali & Paolini, 1998).

ADM is a page-oriented model, as page is the main concept. Each hypertext page is seen as an object having an identifier (its Uniform Resource Locator - URL) and a number of attributes. Its structure is abstracted by its *page scheme* and each page is an instance of a page scheme. The notion of page scheme may be compared to that of relation scheme, in the relational data model, or object class, in object oriented databases. The following example describes the page of an author in a bibliographic site, described by the AUTHORPAGE page scheme.

```
PAGE SCHEME AuthorPage
  Name : TEXT;
  WorkList: LIST OF
       (Authors:  TEXT;
       Title: TEXT;
       Reference: TEXT;
       Year : TEXT;
       ToRefPage: LINK TO ConferencePage
             UNION JournalPage;
       AuthorList:LIST OF
         (Name: TEXT;
         ToAuthorPage: LINK TO AuthorPage OPTIONAL;););
END PAGE SCHEME
```

Each AUTHORPAGE instance has a simple attribute (NAME). Pages can also have complex attributes: *lists*, possibly nested at an arbitrary level, and *links* to other pages. The example shows the page scheme with a list attribute (WORKLIST). Its elements are tuples, formed by three simple attributes (AUTHORS, TITLE and YEAR), a link to an instance of either a CONFERENCEPAGE or a JOURNALPAGE and the corresponding anchor (REFERENCE), and a nested list (AUTHORLIST) of other authors of the same work.

Once a description of the hypertext is available, its materialization is made possible using a mapping language, such as those described in Merialdo et al., 2003).

FUTURE TRENDS

One of the topics currently attracting the interest of many researchers and practitioners of the Web and databases fields is XML. Most efforts are aimed at modeling XML repositories and defining query languages for querying and transforming XML sources (World Wide Web Consortium, 2004).

One of the current research directions is to explore XML as both a syntax for metadata publishing and a document model to be queried and restructured. Mecca, Merialdo, & Atzeni (1999) show that XML modeling primitives may be considered as a subset of the Object Data Management Group standard enriched with union types and XML repositories may in principle be queried using a language like the Object Query Language.

CONCLUSION

Data-intensive hypertext can be published by assuming that the pages contain data coming from an underlying database and that their logical structure is described