Chapter XII

Modeling Data-Intensive Web Sites for Personalization, Integrity and Performance

Christos Bouras
University of Patras, Greece
Computer Technology Institute-CTI, Greece

Agisilaos Konidaris
University of Patras, Greece
Computer Technology Institute-CTI, Greece

ABSTRACT

This chapter presents a step-by-step approach to the design, implementation and management of a Data-Intensive Web Site (DIWS). The approach introduces five data formulation and manipulation graphs that are presented analytically. The core concept behind the modeling approach is that of "Web fragments," that is an information decomposition technique that aids design, implementation and management of DIWS. We then present the steps that must be followed in order to "build" a DIWS based on Web fragments. Finally, we show how our approach can be used to ensure the basic DIWS user requirements of personalization, integrity and performance.

INTRODUCTION

The Web is made up of millions of Web sites, which in turn are made up of millions of interconnected Web pages. A Web site can be seen as a set of Web pages (files in general) that share a common domain name (assigned to a Web server), are linked to each other (and to external Web pages) and can usually be accessed through a gateway Web page called the "home page." Clients request information from Web sites (through HTTP requests) and Web servers respond, by sending this information over the Internet (through HTTP responses). Some of the issues that play a central role in the formulation of the request pattern of a Web site are information popularity and scope, the language used, response times, appealing design and easy navigation. It is clear that certain Web sites are accessed more (or much more) than others, continuously or over a certain period of time. Large volumes of data are requested of these Web sites during the peak periods of user demand. These Web sites are generally called Data-Intensive Web Sites (DIWS henceforward). In this chapter, we give the following definition to DIWS:

A DIWS is a Web site that uses a database as a back-end for the storage of large volumes of frequently updated data. DIWS show high demand for data accessed through their Web pages, continuously or in specific peak periods.

One must always keep in mind that a Web site can be characterized as "Data intensive" only after its usage. The metric of intensiveness is always vague when designing a Web site. That is why, even when one can predict that a Web site will be data-intensive, the "intensiveness" metric cannot be absolutely defined. Thus, a lot of attention must be given to request prediction and scalability, in order to answer to potentially higher demand than predicted.

This chapter will take an in-depth look at DIWS and provide a uniform solution to the following scenario:

Given a large volume of available and frequently updated data, how can a content provider utilize the Web/database paradigm to model, manipulate, present, provide, manage and maintain the data, in a way that answers to the basic user requirements of personalization, integrity and performance.

RELATED WORK

A lot of work has been done in research fields related to data-intensive Web sites. In this section, we will review work that focuses on three basic DIWS fields. These fields are:

- DIWS design, engineering and modeling
- Performance issues in DIWS
- DIWS case studies

There are three outstanding methodologies proposed in the bibliography for designing a DIWS.

STRUDEL, presented in Fernandez (1998), is a Web site management tool that enables site builders to construct and manage a site declaratively. The basic idea is the separation of the Web site's data, the site's structure and the site's graphical representation.

AutoWeb, presented in Fraternali (2000), is a methodology and a development environment for WWW applications. The first step in designing Web applications is their conceptual design. The conceptual schema is then translated to the database schema.

Araneus, presented in Mecca (1998), is a management system for Web-bases. The system proposes a specific organization of Web documents and Web data inside a Web base. This structure ensures efficient management and performance.

Some important work has been presented in Isakowitz (1997). In this work, a fragment-like design approach is presented and called the m-slice approach. This work is closely related, because it can be used (with some extensions and additions) to formally define Web fragments. In Liu (2000), the issue of scheduling frequent executions of queries in order to keep a very-large Web site up-to-date. The interesting feature of this paper is that the frequency of query execution is based on two types of constraints: loose and tight. Cavalcanti (2001), Christodoulou (2001) and Kerer (2001) present methodologies for designing and maintaining Web sites. Brewer (2001) presents several issues that play a role in the design of giant-scale services. Many of these issues can be utilized in the design of DIWS. Florescu (1999) presents a materialization approach for keeping dynamic Web sites up-to-date.

Several papers and articles are concerned with the basic issue of DIWS, which is performance. Most of the papers approach the performance problem through the caching of dynamic data as presented in Challenger (2001, 2000) and Candan (2001). The papers provide different methodologies for caching dynamic Web data on Web servers. They also provide performance models and results on the effectiveness of their approaches.

The work found in Labrinidis (2001) presents efficient materialization methodologies for treating dynamic pages on the Web.

A lot of important work has also been done in the evaluation of the performance and other metrics of real-world Web sites. The first case study

presented in Challenger (2001, 2000) is related to Web sites of Olympic games. The IBM team that was responsible for the design and maintenance of consecutive Olympic games Web sites, has reported on many important issues related to DIWS. The basic features of their approach are load balancing and dependency materialization. Important DIWS-related findings are also presented in Padmanabhan (2000). These findings stem from the study of several performance and access parameters at the MSNBC Web site.

DIWS MODELING SCHEMA

Before presenting our DIWS modeling approach, we present the client/server model on which the approach is based. The client/server model that we will work on in this chapter is the clearly defined three-tier model shown in Figure 1. We consider the database repository and the Web server as autonomous entities that can physically reside on the same hardware or on different machines. They are both referred to as the "Server." We will assume that the Web server is the application part of the model where the DIWS pages reside and the database is the data part of the model where all (or almost all) the information contained in Web pages reside.

The model presented in Figure 1 represents the most popular client/server model on the Web today. The integration of the Web server and the database is continuously growing, and they are sometimes referred to as a Web-Base (Mecca, 1998) in the context of DIWS.

A DIWS must follow specific modeling steps in order to be efficient. The general schema for modeling a DIWS must consider the basic requirements of such a Web site. As already mentioned in Section 1, the basic user requirements are: personalization, integrity and performance. In order to answer to these requirements, we present a general modeling schema for DIWS in Figure 2.

Figure 1: The client/server model of our approach.

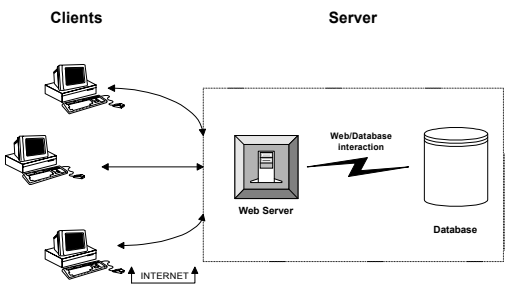
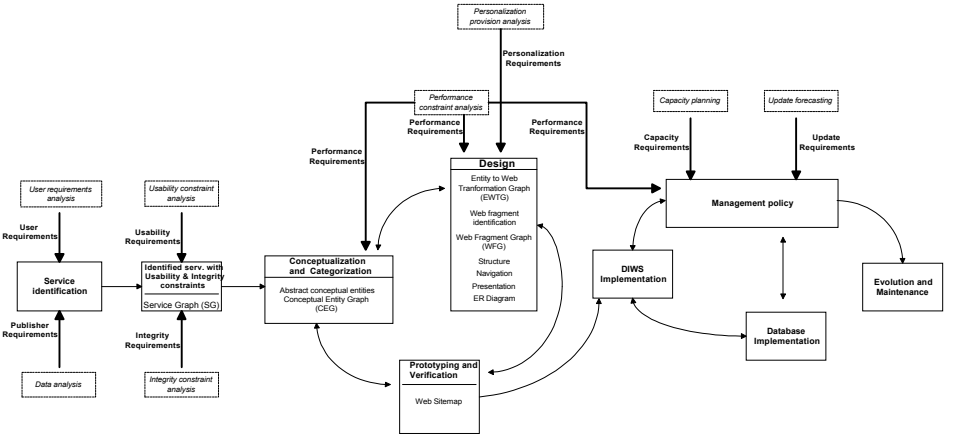


Figure 2: The DIWS modeling schema.



The modeling schema of Figure 2 is based on the schema presented in Fraternali (1999), where it is referred to as the life cycle of a Web application. We have enhanced and altered this schema, in order for it to meet the basic user requirements of DIWS users. The schema must be viewed from left to right. The notation used is presented in the following section together with the analysis of the schema modules.

DIWS MODELING SCHEMA ANALYSIS

An efficient modeling approach involves many key players. The key players are the people involved in the design, implementation and management of a DIWS. In our modeling schema, they are called **publishers, implementors** and **users**. Publishers are the people that have some kind of information that they want to model into a DIWS and provide it through the Web. Implementors are the people that undertake the task of turning the available data (and the publishers' requirements) into a DIWS. Users are the people that show an interest in the publishers' information and intend to access it through the Web. Our modeling approach involves all the key players and requires substantial interaction between them.

In the modeling schema of Figure 2, we first make a distinction between external modeling processes and modeling steps. Modeling processes are intermediate actions that are required for the successful completion of a specific modeling step. The modeling schema of Figure 2 illustrates the necessary steps and processes for creating a DIWS. Before we explain the modeling schema, we present the notation used in Figure 2:

- The small dotted boxes containing italic text are considered external modeling processes
- The normal boxes containing bold text are considered modeling steps
- The bold text beside highlighted arrows are considered external modeling step results and must be viewed as inputs of the modeling steps to which the related highlighted arrows are pointing to
- The normal arrows illustrate the step by step modeling process (from left to right) and also denote the interaction between the modeling steps
- The normal text (under the separating line) inside the modeling steps are internal modeling functions that we wanted to point out in specific modeling steps

External Modeling Processes

These processes are required for the successful implementation of the modeling steps. They result in the creation of several requirements that are considered inputs to the modeling steps to which they apply.

Data analysis

This step involves service providers and implementors. The publishers present the content that they would like to publish on the Web. They also present specific characteristics of the content (e.g., predicted availability). The implementors then analyze the content and perform an initial categorization into topics. An ideal data presentation statement from the part of the publishers during this step would be: "We will be able to provide stock quote information for 10 stock markets world-wide updated every 30 seconds."

User requirements analysis

This process involves users and implementors. As in all software modeling processes, user requirements are a valuable asset in the modeling of a DIWS. User requirements must not be confused with usability requirements. User requirements are usually derived from questionnaires or user interviews. They can also be general requirements related to Web browsing habits, preferred information and personalization issues. This external modeling process usually follows the data analysis process. The implementors have a clear view of the data that must be modeled and can interact with potential users in order to acquire information on how users would want the specific data provided. This modeling process provides the DIWS user requirements. A typical question in a user requirement questionnaire during this step would be: "Which stock market quotes would you like to be included in the home page of the site?"

Usability constraint analysis

This process involves implementors and users. Usability constraints are related to the user target groups of the DIWS. In this step, implementors ask users how they want the information to be accessible. Of course, all proven and documented Web site usability requirements are also taken into consideration. If groups of people with special usability requirements (e.g., special needs groups, handicapped, etc.) are among the target groups of a DIWS, their usability requirements will be taken under consideration during this process.

Integrity constraint analysis

This process involves publishers and implementors. Integrity constraints include linkage, consistency and publisher integrity concerns. During this step, the publishers present any of their specific integrity requirements. Some integrity requirements for the business section of a news portal would be: "The business related links on the home page should all point to URLs that will not be changed even if the information contained in pages that they point to actually changes" or "All business-related pages should include a footer holding current market information."

Personalization provision analysis

This process involves publishers, implementors and users. The publishers and the users provide their personalization requirements, and the implementors model them into Web methods. Personalization is usually related to the economic goals of a site, especially in the field of e-commerce. A publisher may need to enforce personalization constraints that match specific users to specific products or information. On the other hand, users look at personalization as a process enabling them to have easy and instant access to information they actually need. The implementors gather all the constraints and decide on the appropriate (active, passive or both) personalization policy of the DIWS.

Performance constraint analysis, capacity planning and update forecasting

These processes involve implementors. They will be considered the most important external processes of DIWS modeling in this chapter. The performance constraints are related to general performance characteristics of a DIWS. The performance requirements consider the following: simultaneous request servicing, response times, mean values (in Kbytes) of data contained in a Web page, response redirection, load balancing thresholds and many more. We will specifically analyze these processes in respect to the management policy, in the following sections.

Modeling Steps

The modeling steps are the basis of our approach. They may or may not be influenced by specific modeling processes.

Service identification

The implementors utilize user requirements and publisher requirements in order to identify the services that will be provided during this phase. Service identification results in the creation of a service list containing information like: "we will provide a stock quote," "we will provide frequently updated world business news" and so on.

Identified services with usability and integrity constraints

The goal of this step is the transformation of the service list provided by the previous step into a Service Graph (SG). The service graph will be described in more detail later in this chapter, but in brief, it consists of nodes representing services and edges, representing service relations. The service graph illustrates usability and integrity constraints through the text describing each service node and the direction of the edges. A Service Graph is shown in Figure 4.

Conceptualization and categorization

The identified services in the previous step are broken down into abstract conceptual entities in this step. Every service is represented by one or more conceptual entities. The entities are then transformed into Conceptual Entity Graphs (CEG). Every CEG is a series of service steps called conceptual entities. An example of a CEG is shown in Figure 5.

Design

This is the most important step in our modeling approach. The design phase receives the CEGs of the previous step and initially transforms them into Entity to Web Transformation Graphs (EWTG) and then into Web Fragment Graphs (WFG). The core idea of our approach is put to work during this modeling step. The ultimate goal is to develop Web fragments and be able to manipulate them to ensure performance, integrity and personalization. This step also consists of some other basic functions that follow the derivation of Web fragments. These are: Web site structure, navigation, presentation and of course the ER diagram of the back-end database. All of these functions utilize the Web fragment concept as we will show.

Prototyping and verification

This is a step that follows the initial design step. A prototype of the DIWS

is implemented in order to verify that all constraints have been taken into consideration and all design goals have been achieved.

DIWS implementation and database implementation

These are two closely related steps. The Web site implementation also relies on the Web fragment approach and is the extension of the prototyping and verification modeling step. The database implementation relies of the ER design function that was executed during the design phase.

Management policy

The DIWS management policy is at the heart of its performance. The management policy must ensure efficient response speed and Web site integrity. It is influenced by all previous steps, but also relies on two basic external processes: capacity planning and update forecasting. The management policy is the basic feature of the performance analysis that we will present in following sections.

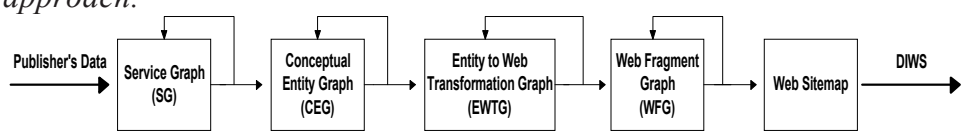
Evolution and maintenance

This is the final step of the DIWS modeling approach as in all software applications. The evolution of a DIWS must be viewed as content increase, presentation alteration and demand alteration. The Web fragment approach that we propose guarantees evolution. The maintenance of a frequently updated site is also a very important issue. The Web site should require minimum professional knowledge in order to be maintained. This is also guaranteed through our Web fragment approach.

WEB FRAGMENT MODELING APPROACH

In this work, we base our modeling approach on Web fragments. We initially introduced the concept of Web fragments as Web components in Bouras (2001a), and we quantified their performance benefits in the context of a DIWS in Bouras (2001b). In this section, we will briefly present the concept of Web fragments and show how they can easily be identified and manipulated through the modeling process presented in Figure 2. In Figure 3, we present the

Figure 3: The data transformation steps that occur in our modeling approach.



data transformation steps that occur in the modeling schema of Figure 2. We introduce the five graphs shown in Figure 3 for modeling a DIWS.

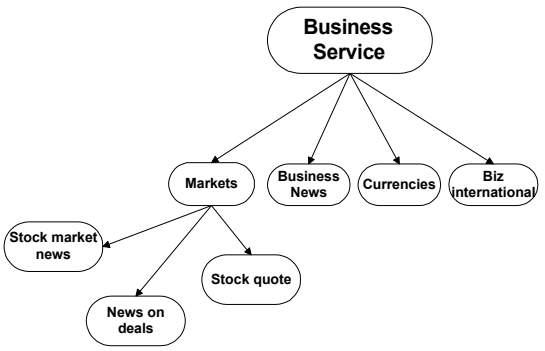
The data transformation steps are serial and each represent a directed graph. A recursive procedure is used to plot the graphs. Conceptually, this procedure stands for the verification process followed during the creation of every graph. We will briefly present each graph and show its significance in the data transformation chain by using the portal business section example.

Service Graph (SG)

The service graph is the first conceptual graph that we introduce in this chapter. This graph is constructed during the service identification phase. It is a directed graph with nodes representing services and directed edges representing service dependencies. For example, a service graph is shown in Figure 4.

The figure illustrates the result of the first manipulation step of the publisher's data. The graph shows the services that were identified and their dependencies. Note that the creation of the service graph is a recursive procedure and that the entities of the graph can be considered standalone services.

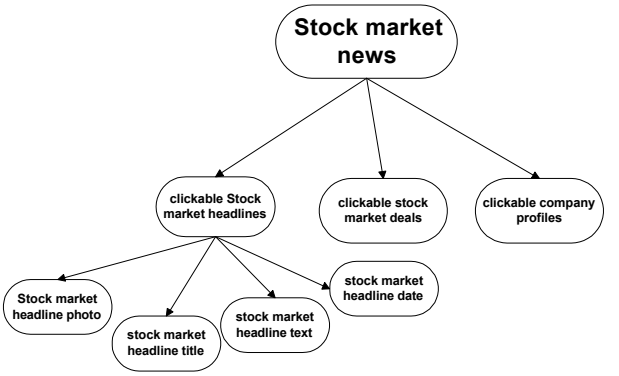
Figure 4: A potential Service Graph (SG) for the business section of a portal.



Conceptual Entity Graph (CEG)

The CEG is a transformation of the SG. This graph is created during the Conceptualization and Categorization step of the modeling process. Every node of the SG is analyzed into conceptual entities. A conceptual entity must be seen as the "means of provision" for every service depicted in the SG. The CEG is also a directed graph where the large oval nodes represent services, the

Figure 5: A potential Conceptual Entity Graph (CEG) for the stock market news section.



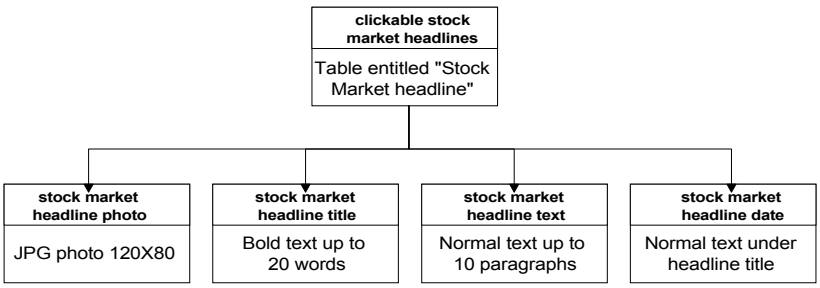
small oval nodes represent conceptual entities and the directed edges represent conceptual entity dependencies: A CEG example is shown in Figure 5. The figure shows that the "Stock market news" service will be provided through: A clickable stock quote (a stock market headline photo, a headline title, headline text and headline date), Clickable stock market deals, Clickable company profiles.

Note that the CEG is an extension of the SG. It further analyses the SG into conceptual entities that actually represent Web (HTML) elements such as images, links and text. These Web elements will be clearly shown after the following transformation.

Entity to Web Transformation Graph

The EWTG is a transformation of the CEG. This graph is created as the first phase of the design modeling step. The graph turns every node (conceptual entity) of the CEG into a real Web entity. Every conceptual entity has its correspondent Web entity that is actually its Web representation.

Figure 6: A potential Entity to Web Transformation Graph (EWTG) for the stock market headlines.



In other words, the EWTG provides Web-related characteristics to every conceptual entity of the CEG. Web-related characteristics can be linkage characteristics, format characteristics (table, bullets or other) and presentation characteristics. The EWTG for the clickable stock market news headlines service are shown in Figure 6.

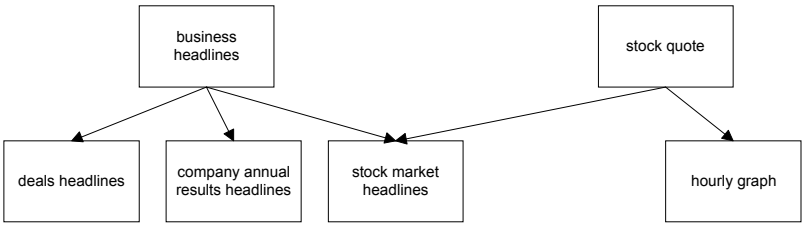
Web Fragment Graph

The WFG is a transformation of the EWTG. Before creating the WFG graph, Web fragments must be identified. The selection of Web fragments from the EWTG is closely related to the database design process that is executed in parallel during the design modeling step. A Web fragment is an entity that contains data of the same semantic content and can be the result of one simple or complex query to the database. The selection of Web fragments is not a difficult procedure when consulting the EWTG. For example, the clickable stock market headline table of Figure 6, together with all of its components (photo, text, etc.) can be considered a Web fragment, because:

- The semantic content of all components are the same
- One can easily conclude that all news headlines can be extracted from the database with the use of one query

A potential WFG is shown in Figure 7. The boxes denote identified Web fragments, and the arrows denote Web fragment relations.

Figure 7: A potential Web Fragment Graph (WFG).



Web sitemap

The Web sitemap is the final phase before the implementation step begins. The Web sitemap is similar to the contemporary meaning of a Web sitemap, but it links Web fragments instead of Web pages. During this phase, the actual Web pages are designed, with every page consisting of several Web fragments. The links between Web pages are actually links between Web fragments contained inside them.

PERFORMANCE ISSUES

In this section, we will look at performance through the Web fragment modeling approach. The ultimate performance goal of a DIWS is the ability to serve as many users as possible, as fast as possible. Performance is closely related to forecasting. When creating a DIWS, implementors must "speculate" on the number of users and the data-intensity of their DIWS. The performance solution that will be adopted before the DIWS goes to work must be able to:

- Efficiently handle the forecasted user requests and data-intensity (availability)
- Adapt to increasing future demand (expandability)

In this section, we propose an efficient approach to performance modeling of DIWS. Our approach is driven by the two basic performance characteristics of a DIWS (availability and expandability).

Adaptive Materialization

Many caching strategies have been proposed in order to efficiently handle the highly dynamic nature of a DIWS. In this work, we will look at caching as the result of materialization. Dynamic pages must be materialized and then cached on the server in order to avoid future, unnecessary, Web server/Database interactions. The issues related to the materialization and caching of dynamic data are:

- If the materialized (static) data is up-to-date
- If the overall system performance is substantially affected by the materialization procedure

We address both of these problems through an approach called adaptive Web fragment materialization. The idea behind this approach is closely related to the modeling methodology presented in the previous sections. As stated earlier, the modeling approach results in the creation of Web pages that include a certain number of Web fragments. Adaptive materialization is the process where materialization is carried out on a "per fragment" rather than a "per page" basis and is able to adapt to current server conditions. There are specific steps to adaptive materialization. These are:

The ranking of Web fragments

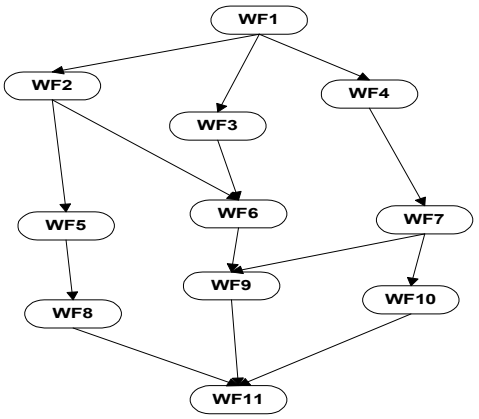
This is the process of ranking all the Web fragments that are the outcome of the modeling process with the use of a Web fragment ranking algorithm. The algorithm considers the number of pages in which each fragment is included and

their popularity. Then it ranks the Web fragments according to their "materialization significance" which is the metric of the positive impact (in terms of resource saving) that the materialization of every fragment will have on the system.

The Web fragment dependency graph

After ranking the Web fragments, the Web fragment dependency graph is created. This dependency graph explicitly shows the data dependencies between Web fragments. A fragment (F2) is dependent on another fragment (F1) when a change to F1 causes a change to F2. Conceptually, this means that when a News Headline Web fragment changes due to breaking news, the Main article Web fragment should also change in order for it to contain the breaking news article. Thus, the Main article Web fragment is actually dependent on the News headlines Web fragment and thus, a materialization to the latter should be followed by a materialization to the former. These dependencies, and thus the materialization order of Web fragments, are shown in a dependency graph such as the one shown in Figure 8.

Figure 8: A Web fragment dependency graph.



Adaptive materialization algorithm

This algorithm receives the Web fragment ranking and the Web dependency graph and then decides on the materialization policy that should be applied.

The algorithm also receives online input from resource utilization variables on the server. In adaptive materialization, if WF2 is materialized, all dependent Web fragments are materialized, meaning that WF6, WF9, WF11, WF5 and WF8 are also materialized. This is a similar approach to the one introduced in Challenger (2000).

Invalidation algorithm

The invalidation algorithm invalidates Web fragments from the Web server cache. The invalidation algorithm can use a series of approaches already suggested in the bibliography such as class-based invalidation presented in Zhu (2001) and up-date propagation invalidation presented in Labrinidis (2001).

INTEGRITY MAINTENANCE-EXPANDABILITY

The Web fragment approach can significantly contribute to the integrity goals set in previous sections.

Content integrity is related to Web page content and database content consistency. The adaptive materialization scheme that we described in the previous section can easily guarantee content integrity at any required level. An important issue related to content integrity and adaptive materialization is usable content integrity. The basic idea is that data needs to be updated only if viewed. Conceptually, this means that a Web fragment should not be materialized if not viewed. In other words, content integrity must be applied proportionally to user requests for specific pages. Why materialize a Web fragment that is rarely viewed with a frequency larger than its viewing frequency? The adaptive materialization algorithm and the invalidation algorithm guarantee this type of integrity.

Linkage integrity is also guaranteed through content integrity. The Web fragment dependency graph guarantees full linkage integrity, because when a link in a fragment is updated due to materialization, all the child fragments are updated. This means that all the fragments that are linked to the fragment being materialized are also materialized.

PERSONALIZATION

The Web fragments modeling approach provides freedom of choice to an implementor in choosing a suitable personalization policy. Metadescriptions can be provided on a Web fragment level and also on a Web page level. This means that under a specific personalization logic, users can be allowed to

manipulate Web fragments and easily categorize the content they are viewing. By consulting the dependency graph, one can easily identify that by removing a parent fragment through an active personalization process, users can exclude whole categories of information. This means that DIWS can be personalized in an extremely high degree and meet specific user needs.

In terms of passive personalization, the DIWS manager can apply specific policies with the use of specific exclusion or inclusion statements at the server. For example, the manager can issue statements such as: "Show the local news headline fragment on the home page for users accessing the site in each country"

These passive personalization statements can be issued by hard-coding them into the Web pages (*if...then....else* statements) or by implementing a high-level passive personalization tool that will be able to transparently handle such directives.

FUTURE TRENDS

In the future, the meaning of a DIWS will surely change. The foreseen increase of Web users will surely add to the request demand of many sites around the world. The increasing demand will bring forward the performance issue for a variety of sites (portals, news sites and others). The increasing amount of data that will be available on the Web will also bring forward the issue of personalization. The most important future enhancement to DIWS is here today. It is the eXtensible Markup Language (XML). The new Web standard can aid Web developers in all aspects of designing and implementing a DIWS. It solves all the issues of defining and separating data, layout and structure in Web pages. It provides efficient methods of personalization and presentation. The most important features though, are the possible improvements that XML can bring to DIWS performance.

CONCLUSIONS

In this chapter, we presented a new modeling approach to DIWS. The core component of our approach is the introduction of a unique Web conceptual entity called the Web fragment. We base all our modeling and data transformation efforts on the formulation of Web fragments. After the formulation of Web fragments, we have shown that the design, the implementation and the management of DIWS can become easier. The Web fragment must be seen as the middle-step to achieving the basic goals of a DIWS. Web fragments can be used in different ways in order to achieve different goals. They can also be seen as a step forward in structuring semistructured data on the Web. It is

a completely different approach to modeling DIWS on the classic Web page model. We have shown how to create Web fragments and how to use them in the context of a DIWS. We are convinced that the concept of Web fragments can further be extended with the use of XML concepts.

REFERENCES

Bouras, C. & Konidaris, A. (2001a). Web Components: A Concept for Improving Personalization and Reducing User Perceived Latency on the World Wide Web. *Proceedings of the 2nd International Conference on Internet Computing (IC2001)*, (238–244), Las Vegas, NV, June.

Bouras, C. & Konidaris, A. (2001b). Run-time Management Policies for Data Intensive Web sites. Proceedings of the International Workshop on Web Dynamics, (1–10), London, UK, January.

Brewer, E.A. (2001). Lessons from Giant-Scale services. *IEEE Internet Computing*, *5*(4), 46–55.

Candan, K.S., Li, W.S., Luo, Q., Hsiung, W.P., & Agrawal, D. (2001). Enabling Dynamic Content Caching for Database-Driven Web Sites. Proceedings of the ACM SIGMOD 2001 Conference. Santa Barbara, CA, May.

Cavalcanti, J.M.B. & Robertson, D.S. (2001). Synthesis of Web Sites from High Level Descriptions. *Lecture Notes in Computer Science, Volume 2016*, 190–203.

Challenger, J., Iyengar, A., Dantzig, P., Dias, D.M., & Mills, N. (2001). Engineering Highly Accessed Web Sites for Performance. *Lecture Notes in Computer Science*, *Volume 2016*, 247–265.

Challenger, J., Iyengar, A., Witting, K., Ferstat, C., & Reed, P. (2000). A Publishing System for Efficiently Creating Dynamic Web Content. Proceedings of the IEEE INFOCOM 2000 Conference, (844–853), Tel Aviv, Israel, March.

Christodoulou, S.P., Zafiris, P.A., & Papatheodorou, T.S. (2001). Web Engineering: The Developers' View and a Practitioner's Approach. *Lecture Notes in Computer Science, Volume 2016*, 170–187.

Fernandez, M., Florescu, D., Kang, J., Levy, A., & Suciu, D. (1998). Catching the Boat with Strudel: Experiences with a Web-Site Management System. Proceedings of the ACM SIGMOD Conference 1998, (414–425). Seattle, WA, June.

Florescu, D., Levy, A.Y., Suciu, D., & Yagoub, K. (1999). Run-Time Management of Data Intensive Web Sites. Proceedings of WebDB 1999 Conference, (7–12), Philadelphia, PA, June.

Fraternali, P. & Paolini, P. (2000). Model-Driven Development of Web Applications: The AutoWeb System. *ACM Transactions on Information Systems*, *18*(4), 323–382.

Fraternali, P. (1999). Tools and approaches for developing data-intensive Web applications: a survey. *ACM Computing Surveys*, *31*(3), 227–263.

Isakowitz., T., Kamis., A., & Koufar, A. (1997). Extending RMM: Russian Dolls and Hypertext. Proceedings of the Hawaii Int. Conf. on System Sciences (HICSS-30), Hawaii, January.

Kerer, C. & Kirda, E. (2001). Layout, Content and Logic Separation in Web Engineering. *Lecture Notes in Computer Science, Volume 2016*, (135–147).

Labrinidis, A. & Roussopoulos, N. (2001). Update Propagation Strategies for Improving the Quality of Data on the Web. Proceedings of the VLDB 2001 Conference, (391–400). Roma, Italy, September.

Liu, H., Ng, W.K., & Lim, E.P. (2000). Keeping a very large Website up-to-date: Some feasibility results. Proceedings of the Electronic Commerce and Web Technologies, First International Conference (399–408).

Mecca, G., Atzeni, P., Masci, A., Merialdo, P., & Sindoni G. (1998). The ARANEUS Web-Base Management System. Proceedings of the ACM SIGMOD 1998 Conference, (544–546), Seattle, WA, June.

Padmanabhan, V. & Qiu, L. (2000). The content and access dynamics of a busy Web site: Findings and implications. Proceedings of the ACM SIGCOMM 2000, (111–123). Stockholm, Sweden, September.