Ready for Prime Time - Pre-Generation of Web Pages in TIScover

Birgit Pröll, Heinrich Starck
Institute for Applied Knowledge Processing (FAW), University of Linz
{bproell | hstarck}@faw.uni-linz.ac.at

Werner Retschitzegger
Department of Information Systems (IFS), University of Linz
werner@ifs.uni-linz.ac.at

Harald Sighart
ORACLE Consulting Frankfurt, Germany
hsighart@de.oracle.com

ABSTRACT

In large data- and access-intensive web sites, efficient and reliable access is hard to achieve. This situation gets even worse for web sites providing precise structured query facilities and requiring topicality of the presented information even in face of a highly dynamic content. The achievement of these partly conflicting goals is strongly influenced by the approach chosen for page generation, ranging from composing a web page upon a user's request to its generation in advance. The official Austrian web-based tourism information and booking system TIScover tries to reconcile these goals by employing a hybrid approach of page generation. In TIScover, web pages are not only generated on request in order to support precise structured queries on the content managed by a database system. Rather, the whole web site is also pre-generated out of the extremely dynamic content and synchronized with the database on the basis of metadata. Thus, topicality of information is guaranteed, while ensuring efficient and reliable access. This paper discusses the hybrid approach as realized in TIScover, focussing in particular on the concepts used for pre-generation.[1]

Keywords

WWW, tourism information system, page generation, optimization, reliability

1. INTRODUCTION

The basic architectural principle of the WWW is a simple client-server communication: Whenever a WWW-client requests a web page identified by an URL, the corresponding page is returned from the server to the client. However, the architecture behind is recently getting more and more sophisticated. At the beginning, the web pages of a web site were simply stored in the web server's directory and edited directly in case of any changes to *content* (i.e., the data presented at a site), *structure* (i.e., the logical composition of a site and each of its pages) or *layout* (i.e., the graphical representation of each page) [6]. Nowadays, not least due to new requirements emerging from several application areas for the web such as electronic commerce, the employment of database systems (DBS) to store at least the content of a web site turned out to be worthwhile. With this, it is possible to *easily* handle large amounts of data in a *consistent way* on a *large distributed scale*.

One key decision that has to be made when integrating DBS with web sites is *when* and *how* to generate the web pages out of the content stored within the DBS. This process which is called *page generation* can be performed basically in two alternative ways [3]. First, it can be done *on request* which means that not before a user requests a certain page the corresponding data is retrieved out of the database, the page is composed and delivered to the client. Second, the pages or some part thereof can be *pre-generated*, i.e., in advance of a user's request. This can be done *periodically* or *immediately* when a modification is done which affects content, structure or layout. Finally, these two basic alternatives can be combined in that upon modification, the affected web pages are invalidated only, whereas pre-generation is performed not before these pages are requested. This solution is especially effective for those web pages where modification frequency exceeds request frequency, since in that case unnecessary pre-generation is avoided. For all of these alternatives, the generated pages can be additionally cached on the client, on the server or on a proxy. There are good reasons for each of these approaches (cf. Section 3) and finding the optimal tradeoff represents a major challenge in run-time management of web sites [6]. The official Austrian tourism information and booking system TIScover has tried to cope with this need by employing a *hybrid approach* in that generation on request and generation on change are combined into one system [9]. This was done in order to ensure high performance and failure liability while preserving topicality and providing exact structured search capabilities at the same time.

[1] This paper has been also accepted for presentation at the ACM SIGMOD Workshop on the Web and Databases (WebDB'99), Philadelphia, Pennsylvania, 3-4 June, 1999.

This paper discusses the hybrid approach and particularly addresses the pre-generation of web pages. After a short overview of TIScover in Section 2, Section 3, introduces the hybrid approach. In Section 4, the design goals and prerequisites underlying the process of pre-generation are identified and the concepts used for pre-generating web pages are illustrated by means of a running example. Section 5 is dedicated to lessons learned in the course of the realization and utilization of pre-generation in TIScover. Finally, Section 6 compares our approach to related work and gives an outlook to future work.

2. AN OVERVIEW OF TIScover

The development of TIScover has been started in 1996 based on the experiences made with the pioneering system TIS@WEB [2]. The aim of TIScover is twofold [9], [10], [11]: first, tourists should be *supplied with comprehensive, accurate and up-to-date tourism information* on countries, regions, villages and all destination facilities they offer like hotels, museums or other places worth seeing. Second, it aims to *attract the tourist to buy certain tourism products* either offline or even more important to allow the tourist to buy them *online*. Originally, TIScover was realized to market the facilities of a certain region of Austria, namely Tyrol, only. Meanwhile, five other Austrian regions have joined TIScover [14]. Besides that, TIScover has been employed in Asia, presenting tourism information about Thailand [15][2], it is currently used by the German company START Media Plus, a major player in the area of online reservation systems, to present tourism information about Germany [16] and it has just gone online in Switzerland as KISSswiss, employed by the companies Kümmerly+Frey and Basler Versicherungen [17].

Currently the database of TIScover Austria comprises about two gigabyte of data, distributed over approximately 400 tables. There exist more than 400.000 web pages stored in about one million files. The content covers among others 1.500 towns and villages and 20.000 accommodations originating from more than 4.500 tourism information providers, ranging from hotels to local tourism offices. Per month, the system has to handle up to 5 million pageviews, 1.2 million visits as well as up to 20.000 requests for information and online bookings.

As illustrated in Figure 1, the functionality provided by TIScover can roughly be categorized into three different components, the *public Internet* component, the *Extranet* and the *Intranet* [10]. The *public Internet* component comprises that functionality of the system which is accessible to the public, whereby the most important modules are *Atlas* and *Booking*. The module Atlas allows the customer to browse through all kinds of tourism information by navigating through a geographical hierarchy and to use a *full text search*. The module Booking allows for a *precise structured search* based on a subset of the tourism information presented by Atlas, like villages, hotels, available rooms, events and camping sites along the geographical hierarchy as well as *online booking* of these tourism products. Furthermore, TIScover provides an *Extranet* allowing authorized tourism information providers, no matter being a small guesthouse or a large local tourist office to update and extend their tourism information and products directly. Finally, the *Intranet* component of TIScover

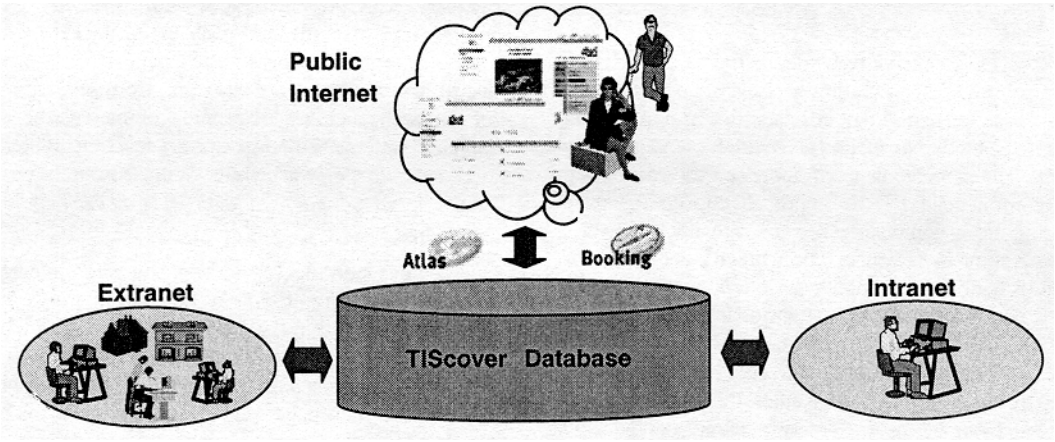

Figure 1: Functional Components of TIScover

which is accessible at the system provider's side only allows configuring the whole system in various ways. It is, for example, possible to extend the geographical hierarchy, to specify expiration dates for reports and to define the default language for all system components.

3. A HYBRID APPROACH FOR PAGE GENERATION

When deciding about which approach of page generation should be employed in TIScover, several pros and cons of each alternative were considered. Generation on request is most commonly used today, not least since it provides both, *topicality of information* and the possibility for the user to retrieve data based on *precise structured database queries*. However, especially when considering data- and access-intensive web sites, this approach has strong influence on *runtime performance* and *failure liability*. This is mainly due to the rather complex operations that have to be performed when a user requests a certain web page [4]. Another disadvantage that should not be underestimated especially concerning commercial web sites is that many of those web sites generating pages on request are part of the so-called *hidden web* [6]. This means that these pages are often not accessed by web crawlers for indexing and are consequently not found by search engines.

On the contrary, pre-generation circumvents these drawbacks by performing the task of retrieving data, composing web pages and

[2] Note, that due to the economic crisis in Asia, the asian version of TIScover is currently not operating.

storing them at the server in advance of a users request, thus allowing the web server to *reliably deliver several hundred pre-generated pages per second* [4]. Following a rule of thumb, pre-generation is especially useful for those cases where the number of requests for a certain page is higher than the number of updates affecting content, structure or layout of this page. However, one problem with this approach is that precise structured queries are hardly possible[3]. In order to provide the user with search capabilities anyhow, *traditional information retrieval techniques* have to be employed [12]. Another problem is that pre-generation very much hardens the *personalization* of a web site. This is since personalization would require that the web site adapts at runtime to the special needs of different classes of users by providing, e.g., appropriate navigation patterns and presentation styles [3]. In any case, the most severe drawback of pre-generation, especially in view of a high update frequency, is that the pre-generated web pages have to be synchronized with the database content (cf. Section 4).

To take advantage of the benefits of each of these approaches and circumventing their drawbacks, TIScover employs a *hybrid approach* for page generation. For this, the module Booking directly accesses the database and generates pages on request, thus coping with the demand for precise structured search facilities. On the contrary, the module Atlas which allows for navigation and full text search works on pre-generated pages, thus satisfying the need for reliable and efficient access. These pages represent the whole content as managed by the TIScover database. In the following, the pre-generation part of this hybrid approach is described in more detail.

4. PRE-GENERATION OF WEB PAGES IN TIScover

4.1 Prerequisites and Design Goals

The pre-generation process employed in TIScover is based upon some prerequisites and design goals, driving its realization.

4.1.1 Separating Content, Structure and Layout

In TIScover, *content level*, *structure level* and *presentation level* are separated from each other as far as possible in order to ease the tasks of updating and restructuring the web site. First, the content level comprises the database tables and attributes holding different kinds of *tourism objects* such as hotels and rooms. Second, the structure level comprises *page types* and *entities*. A page type represents a number of pages at the type level, having the same logical structure such as a homepage or a page for room descriptions. The purpose of an entity is to group all those page types that logically make up an information unit. For example, the entity hotel contains a homepage, an extra page with pictures, pages for directions and pages for room descriptions. In this context, TIScover requires that on the one hand the *same content* (e.g. the telephone number of a hotel) can be *represented on multiple page types* possibly associated with different entities and on the other hand to be able to base any *given page type* on *multiple database tables*. Finally, the presentation level covers layout information for the different page types. Summarizing, pre-generation has to take this separation into account by managing the necessary mapping data in a flexible way.

4.1.2 Individuality of Layout

Due to the requirements of tourism information providers, whose main concern is to differentiate from their competitors, TIScover has to support *individual layouts*. This means that the tourism information providers should be able to choose an appropriate layout for each of their pages on the basis of the Extranet. For this, the system provider constructs different *layout template sets* on the basis of the Intranet. A layout template set groups a number of templates representing a homogenous design for different page types, e.g., in terms of background color, positioning of logos and textual descriptions. This means that every time a certain page is pre-generated, the associated layout template has to be determined by the system.

4.1.3 Granularity of Generation

Last but not least, an important goal concerns the granularity of generation. A trivial approach would be to *pre-generate the whole set of web pages* periodically or even immediately after a change occurred. However, it is obvious that this is not applicable in face of a data- and access-intensive web site like TIScover, neither in terms of time restrictions nor in terms of processing overhead. Besides that, one could also imagine a finer granularity such as the *pre-generation of all those pages whose page type is affected* by the change. However, in order to avoid unnecessary generations, the goal followed in TIScover is to pre-generate only *those web pages* that are *actually affected* by a change.

4.2 Pre-Generation by Example

Whenever *content* or *layout* of the TIScover web site is maintained either by tourism information providers using the *Extranet* or by the system provider on the basis of the *Intranet*, the affected web pages have to be determined and pre-generated. Note, that automatic pre-generation in case of changes to the *structure* is at this time not supported by TIScover (cf. Section 6). In the following, each step necessary for pre-generation is described in more detail by means of a running example. In order to reduce complexity, two simplifications have been made. First, only one type of event, namely a modification of the content by means of the Extranet is considered. Second, only a subset of the tables and attributes containing the metadata necessary for pre-generation are actually shown.

Let's assume that the hotel "Gasthof Post" changes its telephone number by means of the Extranet. At the content level, the telephone number is represented by an attribute of the table `Hotel`. At the structure level, the telephone number is assigned to two different pages, namely the homepage of the hotel and the list of accommodations, maintained by the village, where the hotel is located. In the following, the steps are described, which are necessary to pre-generate these two affected pages (cf. Figure 2).

4.2.1 Registering of Events

First of all, the event, i.e., the update of the telephone number by the "Gasthof Post" (cf. □ in Figure 2) has to be registered. For this, information about the event is automatically inserted into the so-called *EventQueue* by the Extranet (cf. ②). This information comprises the *ID(s)* of the modified *tourism object(s)* (`4711`), the name(s) of the updated table(s) called *triggering table(s)* (`Hotel`)

[3] Note, that there are already several approaches, allowing to apply complex predicates on the content and the structure of a page [6]. However, these approaches are not further considered.

and the name(s) of the affected attribute(s) called *triggering attribute(s)* (TelNo).

4.2.2 Determining all Affected Page Types

The *StructureManager*, which is realized as an independent OS process, continuously monitors the EventQueue by means of polling[4] (cf. □) whereas the polling interval can be configured within the Intranet. As soon as a new entry is detected, the StructureManager queries the *Content_To_Structure_Mapping* table using the triggering table (Hotel) and the triggering attribute (TelNo) as key (cf. □). As the name indicates, this table contains metadata about the mapping between content information and structure information. In particular it contains all *page types*, a certain attribute is placed on. Referring to our example, the telephone number is placed on both, on a homepage (Homepage) and on a list of accommodations (ListOfAccom). These page types are further categorized by depicting the *entity* the page type is associated with. This categorization is necessary, since a certain page type can be used by different entities. For example, the page type homepage can not only be used by the entity hotel, but also by the entity village. Returning to our running example, the homepage (Homepage) is categorized by means of the entity hotel (Hotel), whereas the entity village (Village) categorizes the list of accommodations (ListOfAccom).

4.2.3 Identifying the Relevant Tourism Objects

The next step is to find the content, respectively the tourism objects, which are, besides those affected by the maintenance process (4711) necessary for pre-generation. For this, the StructureManager uses the table *Structure_To_Content_Mapping* containing metadata in terms of appropriate SELECT-statements, expressing how to retrieve the tourism object(s) associated with a certain page at a certain geographical level expressed by the two attributes Entity1 and Entity2 (cf. □). In our example, the query searches for the village, the accommodation "Gasthof Post" is located in and returns the ID 8181. Now, the StructureManager completes its task by inserting the tourism objects necessary to pre-generate the affected pages into the *TaskQueue* (cf. □).

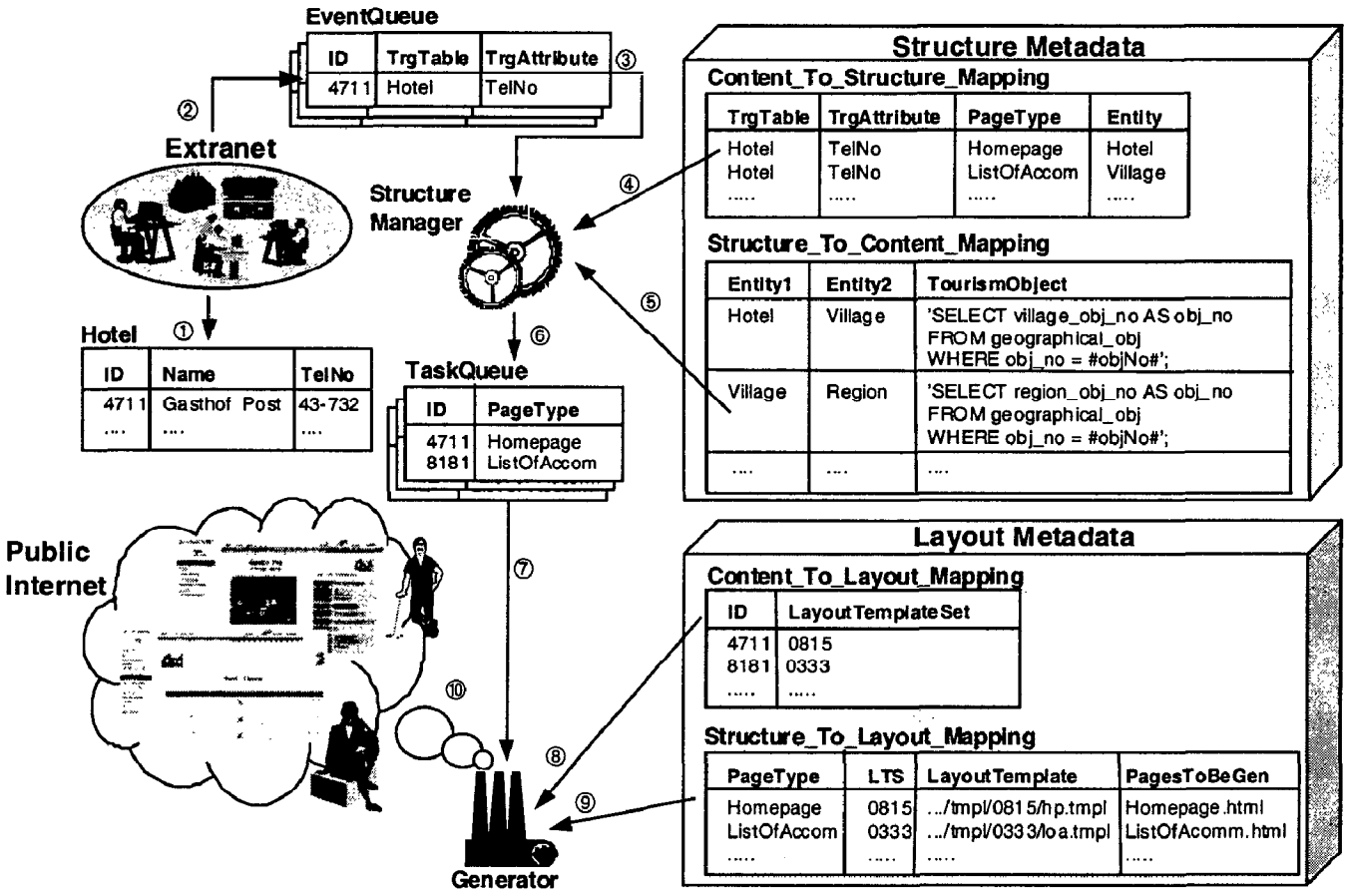

Figure 2: The Process of Pre-Generation by Example

4.2.4 Determining Layout Templates and Affected HTML Pages

Finally, it has to be determined which *concrete HTML page files* have to be pre-generated and which *layout templates* are necessary in order to ensure that the newly generated pages get the same layout as before. For this, the *Generator* appears on stage. Analogous to the StructureManager, the Generator is realized as an independent OS process, which continuously monitors the TaskQueue (cf. □). After recognizing a new task, the *layout*

[4] Instead of polling, it would also be possible to activate the Structure Manager by means of database triggers [7]. However, this is part of future work.

template set, which has been assigned to each tourism object by the tourism information provider is retrieved on the basis of the metadata stored within the table *Content_To_Layout_Mapping* (cf. □). This is `0815` concerning the tourism object `4711` and `0333` with respect to the tourism object 8181. Next, the Generator has to determine out of this template set the *concrete template file* for the respective page type and the *concrete HTML page file* which has to be re-generated (cf. □). This is done on the basis of metadata, stored within the table *Structure_To_Layout_Mapping*. The Generator is finally able to generate only those web pages that were affected by the maintenance process (cf. □).

5. LESSONS LEARNED

During the development and especially in the course of the real utilization of pre-generation in TIScover, several issues have become apparent which should be discussed in more detail in the following.

5.1.1 Propagation and Update Delays cannot be Avoided

TIScover has to generate in average 24.000 web pages per day, representing more than 5 percent of the whole TIScover web site. This high workload results first from the *large number of tourism information providers* which are able to maintain their data by means of the Extranet. Second, it is a consequence of the *huge amount of data* which is already managed by TIScover and third and probably most important it is due to the *dynamic nature of some tourism information*. For example, rates, schedules, events, opening hours and reports on weather, snow conditions and avalanche conditions change with the days, weeks, months and seasons. In such a scenario, the pre-generation of web pages automatically leads to *propagation and update delays* which cannot be completely avoided. Nevertheless, to reduce these delays, in TIScover not only a single Generator can be employed, but rather an arbitrary number of Generators can be configured to *process the TaskQueue in parallel*, thus improving the throughput of the system. To reduce the number of pages which have to be pre-generated even further, larger fragments of a web page that appear on several other web pages too (e.g. a logo together with some textual descriptions) are defined separately by using the Server Side Include mechanism. As a consequence, the assembly of the whole web pages takes place not before a user's request.

5.1.2 Hot Spots Exist Within the Content

It has also been shown, that there are several *hot spots* within the system, posing particularly high requirements on topicality. For these hot spots, pre-generation time must be reduced to a minimum, whereas for other parts (such as the new background color of the homepage of a village), it is acceptable to be slightly out of date. Some hot spots can be found within those content, being subject to a particularly *high update frequency possibly at the same time* (e.g., snow reports are actualized by local tourism offices at least every morning). Another area in TIScover, where pre-generation time must be minimized is those part of the content which is not only presented by the Atlas but also searchable by means of the module Booking. Since it directly accesses the possibly updated content, there would be *inconsistencies* between the information presented by the Booking module and the same one presented by the Atlas. That is, it should be ensured that the Booking module makes - within a certain tolerance - the same information available as the Atlas at any time. One mechanism which is used by TIScover to deal with hot spots are *priorities*. The system provider is able to specify by means of the Intranet priorities on different levels of granularity, ranging from entities via page types to tourism objects. As a consequence, elements with high priority are precedented by the Generator. In addition, it is possible to *explicitly manipulate* the TaskQueue by deleting or changing the order of already scheduled tasks or by scheduling new high priority tasks at the pole position.

5.1.3 Complex Navigation Structures Must be Considered

TIScover is a *frame-based system*, in that besides the content frame, two additional context-dependent navigational frames are supported. This leads to a complex navigation structure which has to be pre-generated in case of any changes, a fact that is not explicitly considered by many research prototypes [3]. In TIScover, navigation frames are treated analogous to "normal" page types. That means, they are also part of the structure and layout metadata and are pre-generated in the same way as content frames.

5.1.4 Multilinguality and Multi-Currency Support Increases Pre-Generation Overhead

Since TIScover acts in an international context, multilinguality in terms of an arbitrary number of languages and multi-currency is supported by all of its functional components. In order to achieve multilinguality, every page has to be pre-generated in each language supported by the system. For pages containing price information not only pure HTML code has to be generated but rather the system is required to generate client-side JavaScript code. On time of a user's request, this code is responsible for calculating the price information in the chosen currency.

5.1.5 Mapping Content, Structure and Layout by Means of Metadata Enhances Flexibility

As described in Section 4.2, knowledge about the mapping between content, structure and layout is to a large extent not hard-coded but rather reified in form of metadata and stored within the database. This allows to flexibly change the mapping without affecting the StructureManager or the Generator application.

6. RELATED WORK AND OUTLOOK

There already exist some advanced research prototypes dealing with the issue of page generation. The prototype system *Araneus*, for example, employs a hybrid approach similar to TIScover and allows pre-generating only those pages that are affected by changes [1], [13]. The detection of affected pages is also done on the basis of metadata stored within the database system. Unlike TIScover, re-generation of affected pages is done by means of database triggers. The prototype system *AutoWeb*, in general, generates pages on request [8]. Additionally it is possible to selectively pre-generate portions of the site based on semantic criteria. It is not clear, however, if pages, affected by changes can be detected automatically by the system, as it is the case in TIScover. Again metadata stored within the database is used to represent the site's structure, navigation and presentation. Finally, the prototype system *Strudel* generates pages on the fly [5]. Mechanisms for propagating changes to affected pages are not mentioned. It is planed to extend the system in order to give the designer the choice between pre-generation and composition of

pages on request. Unlike the other approaches, metadata is stored within an ad hoc repository for semistructured data.

In general, one main difference to all of these prototypes is that the generation approach taken in TIScover has to consider many issues which are required because of the employment of TIScover as a commercial system such as multilinguality, content dependent navigation structures and hot spots within the content to mention just a few. Work to integrate additional existing research ideas in TIScover is in progress. Currently, two major concerns are first, to extend the metadata schema in order to incorporate also changes on the structure level without enforcing a manual change of the layout templates and the structure metadata, and second, to increase the separation between layout and structure by using XML and XSL instead of HTML.

7. REFERENCES

[1] Atzeni, P., Mecca, G., Merialdo, P. Design and Maintenance of Data-Intensive Web Sites. In Proc. of the Conf. On Extending Database Technology (EDBT'98), Valencia, Spain, 1998.

[2] Burger, F., Kroiß, P., Pröll, B., Richtsfeld, R., Sighart, H., Starck, H. TIS@WEB - Database Supported Tourist Information on the Web. In Proc. of the Int. Conf. on Information and Communication Technologies in Tourism (ENTER'97), A Min Tjoa (ed.), Springer, Edinburgh, 1997.

[3] Ceri, S., Fraternali, P., Paraboschi, S. Design Principles for Data-Intensive Web Sites. In SIGMOD Record, Vol.24, No. 1, March 1999.

[4] Challenger, J., Iyengar, A., Dantzig, P. A Scalable System for Consistently Caching Dynamic Web Data. In Proc. of the IEEE Infocom '99 Conf., New York, March 1999.

[5] Fernandez, M., Florescu, D., Kang, J., Levy, A., Suciu, D. STRUDEL: A Web Site Management System. In Proc. of the ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'97), Tucson, Arizona, June 1997.

[6] Florescu, D., Levy, A., Mendelzon, A. Database Techniques for the World Wide Web: A Survey. In ACM SIGMOD Record, Vol. 27, No. 3, Sept. 1998.

[7] Kappel, G., Retschitzegger, W. The TriGS Active Object-Oriented Database System - An Overview. In ACM SIGMOD Record, Vol. 27, No. 3, Sept. 1998.

[8] Paolini, P., Fraternali, P. A Conceptual Model and a Tool Environment for Developing More Scalable, Dynamic, and Customizable Web Applications. In Proc. of the Conf. On Extending Database Technology (EDBT'98), Valencia, Spain, 1998.

[9] Pröll, B., Retschitzegger, W., Wagner, R.R., Ebner, A. Beyond Traditional Tourism Information Systems – TIScover. In Journal of Information Technology in Tourism, Vol.1, Inaugural Volume, Cognizant Corp., USA, 1998.

[10] Pröll, B., Retschitzegger, W., Wagner, R.R. TIScover - A Tourism Information System Based on Extranet and Intranet Technology. In Proc. of the 4th Americas Conf. on Information Systems (AIS'98), Baltimore, Maryland, 1998.

[11] Pröll, B., Retschitzegger, W., Wagner, R.R. Holiday Packages on the Web. In Proc. of the Int. Conf. on Information and Communication Technologies in Tourism (ENTER'98), D. Buhalis et al. (eds.), Springer, Innsbruck, 1998.

[12] Salton G., and McGill M. Introduction to Modern Information Retrieval, McGraw-Hill, New York, NY, 1983.

[13] Sindoni, G. Incremental Maintenance of Hypertext Views. In Proc. of the Workshop on the Web and Databases (WebDB'98), (in conjunction with EDBT'98), 1998.

[14] TIScover Austria, http://www.tiscover.com, TIS GmbH Innsbruck, FAW Hagenberg, 1999.

[15] TIScover ASIA, http://www.tiscoverasia.com, GoThailand, 1998.

[16] TIScover Germany, http://www.deutschlandreise.de, START Media Plus, 1999.

[17] TIScover Switzerland, http://www.kissswiss.com, Kümmerly+Frey, Basler Versicherungen, 1999.