WORLD WIDE NEWS GATHERING AUTOMATIC MANAGEMENT

Luís Veiga and Paulo Ferreira
{luis.veiga, paulo.ferreira } @ inesc.pt
INESC, Rua Alves Redol, 9 - Lisboa -1000 Lisboa - Portugal

Abstract
The world-wide-web does not support referential integrity, i.e. dangling references do exist. This can be very annoying; in particular, if a user pays for some service in the form of web pages, he requires such pages to be reachable all the time. Currently, ensuring such referential integrity is the responsibility of webmasters: while a page is referenced the corresponding file never gets deleted. However, it is well know that this manual memory management is extremely error-prone leading to dangling references and storage leaks. We propose a solution for this problem based on the use of a garbage collection algorithm applied to the web. Thus, the referential integrity is ensured at the system level. We developed an application of web news in which a dangling reference never occurs.

Keywords: World Wide Web, Referential Integrity, Replication, Garbage Collection.

1. Introduction

Today, many international news agencies (e.g. Reuters, France Press, etc.) have electronic news sites implemented and integrated with the World Wide Web. These sites contain electronic on-line data in the form of text, photographs, audio and video clips.
Most newspapers' reporters use such sites as their source of information. As international news is concerned, most are simply replicated, bought to and retrieved from such international news agencies (INA). As examples of news which would be locally or nationally gathered and divulged, practically unchanged, to just every other nation in the globe, we can state these: natural catastrophic events, big plane crashes, train accidents, major international sport competitions events, internationally acclaimed prizes ceremonies, etc.
Numerous examples of these situations can also be found at regional newspapers (RN) where almost every single news presented, apart from the really local ones, are simply transcribed from, inspired in or duplicated (as you may wish to consider it) from one or various larger, nation-wide newspaper (NN).
These hypermedia objects are transferred from the INAs to others WebNews publishers (NNs and RNs) by a common communication way (e-mail mainly or ftp for large packets of information) in a totally manual or very lighted automated fashion. Thus, the employees of INAs, NNs and RNs do the management of these objects manually.
This manual management is clearly inefficient and error-prone. For example, some files may be erroneously deleted from the web server of an INA when there are still URLs pointing to it from NNs or RNs. This may occur because the disk space in the INA´s web server has been exhausted and some files must be deleted. Thus, this leads to the most annoying situation of dangling references.
In addition, it may happen that files no longer referenced from any other site are never deleted. This obviously leads to storage being wasted.
These situations also affect single users as explained now. Currently, most users keep in their bookmarks URLs for those web pages

with news that interest them most. These pages are provided either by INAs, NNs, or RNs.
It is most annoying that, some time after being book-marked, such URLs no longer point to a valid page because the corresponding file has been deleted. As already mentioned, this results from erroneous manual memory management performed by webmasters.
In conclusion, the lack of referential integrity of the web, i.e. not ensuring that a URL always point to a valid document, leads to dangling references and storage leaks.
The contribution of this work is the design and implementation of a system that ensures referential integrity at the system level. We use this functionality to provide a service of WebNews where dangling references and storage leaks never occur. In addition, webmasters are released from the complex task of memory management. The system uses a distributed garbage collection algorithm to maintain the referential integrity [ferreira98].
The paper is organised as follows. In the next section we present the model in which our approach is based. Then we describe the system architecture. The fourth section presents the most important implementation aspects. Section five presents our conclusions and some future work.

2. Model

This section presents the **worldwide News** model**.** This model describes the structure of personal, regional, national and international World Wide Web publications and its interconnections.

2.1 Memory Organisation

This model consists, generically, in a WARM (Wide Area Replicated Memory). This model assumes data shipping as the only means to access data. This data, once obtained through replication, is accessed only locally and, expectably, maintained in a cache for future use.

The WARM Runtime System supports the replication of data and the communication among web sites. Basically, it implements a wide-area distributed memory system [perdis00].
In this model, the lifetime of data is determined by its accessibility. In other words, while a datum is reachable it must persist; otherwise, the storage it occupies can be freed.
In this model data exists in two forms:

- **HTML documents** containing URLs to other documents making an accessibility graph.
- **Non-HTML data objects** consisting mainly of files of different types integrated or embedded in the documents referred earlier. These objects are, with the exception of text embedded in HTML documents, the true bearers of information in its most diversified forms. Their lifetime is determined by the existence of links to them in some reachable HTML documents as they are, essentially, leaf nodes in the accessibility graph.
- **Replicas** of the two types of objects just described (modified or not). These replicas may exist in any personal, regional, national or international web news site.

For simplicity, these different types of data will be referred to as simply objects whenever no distinction between them should be necessary.
These objects and the references among them form a complex graph. The roots of this graph are the URLs stored in any bookmark of the sites belonging to the worldwide News system.
The reachability of an object graph is defined as follows: an object is said to be reachable if it is accessible, directly or indirectly, from a root by following references.
Only the objects enfolded in this accessibility graph are considered as reachable. Others are excludable, as they waste storage space. The reclamation of unreachable objects is done automatically with a garbage collection algorithm that also ensures the referential integrity.

2.2 Process Model

It is important for the widespread of the model that no constraints or restrictions are posed on the publication form. Thus, the processes involved are basically, web browsers and web servers.
There are two main functions that processes may perform: **importing** and **publishing** news. The importing of information is done using an enhanced web browser able to fulfil a protocol that maintains object references consistent, i.e. preserving the correctness of the objects' reachability graph.
With this web browser a user (working in a NN, in a RN, or any other person) can import into its computer a file corresponding to some specific news. This results in the creation of a new replica of that file.
News objects are published (i.e., made accessible to everyone) by putting the corresponding file in a web server).
Both the web browser and the web server being used are enhanced with garbage collection (GC) code that ensures the referential integrity of the system (this is explained with more detail afterwards).

2.3 Coherence Model

The coherence engine is the entity of the WARM that is responsible for ensuring and maintaining system-wide coherence of replicas.
In this model, coherence among different replicas may not always be maintained due to operations that modify objects content. These consist, essentially, in operations like translation of text and sound content. These are either mixed with or replace the original content. Note that, as previously mentioned, every time an import action is performed, a new replica of the object is created.

3. Architecture

In this section, we describe the architecture of the system.

3.1 Application Overview

This system is based on client-server architecture. The client side of the application comprises of a regular Web browsing tool. News gathering consists in Web browsing sessions interleaved with the creation of remote references to various news objects on any number of sites, i.e. news importing.
As far as the server side is concerned, the application is, once again, a common Web server. Server tasks include, mainly, receiving reference creation messages from clients and, based on them, manage persistence related features and deleting unreachable data.
The user (gathering news) browses the web and whenever he desires to use some news object, he has to trigger an **import** operation (see figure 1). An **import** message is sent to the web server at the originating site, consisting only of the object's URL. The server replies back with the objects content.
There, at the server, a GC service updates GC specific data structures related to the objects being imported and, in response, sends a set of pre-processed data to maintain reachability and WARM coherence.

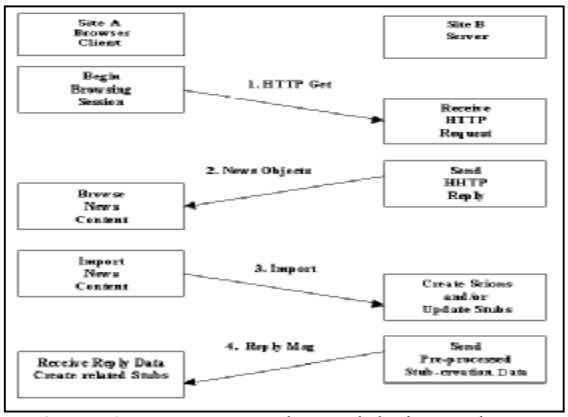

Figure 1: Messages exchanged during an import operation

3.2 News Composing

Once information objects are imported, they can be manipulated in any fashion by any tool (see figure 2). For example, a journalist can use any editing or composing software tool to modify his object replica, adding to the flexibility and ease of deployment of the model. Such a modification results in the deletion and/or creation of references to other objects. From the point if view of the graphs reachability, the only relevant operation is the **assignment** of references; this results either in the creation of new references or in the deletion of already existing ones.

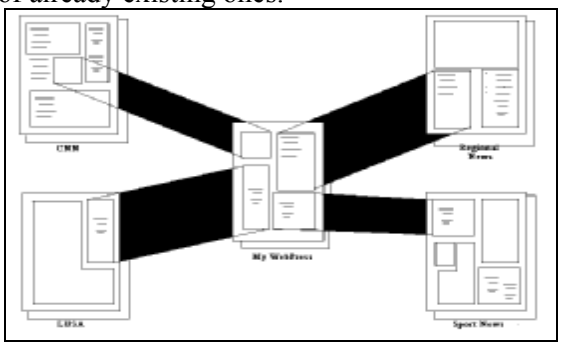

Figure 2: Example of new content creation based on imported content

3.3 Garbage Collection

Referential integrity is enforced through a distributed garbage collection algorithm [ferreira98]. We do not describe the algorithm in detail since it is out of the scope of this paper. Instead, we only present its main ideas.

As mentioned in the previous section, in the worldwide news gathering model, assignments are the operations that create references to information objects.

An important aspect for the efficiency, and indeed the very effectiveness of this model, is the preservation of referential integrity and the reclamation of storage allocated to objects that are no longer reachable. This model feature is based in two components, one local and one distributed.

The local component of the garbage collector is in charge of collecting the objects cached locally at each process. The distributed component is the one responsible for maintaining the inter-process links between different objects, preventing remotely reachable objects from being incorrectly reclaimed and its storage space reclaimed

4. Implementation

The system is implemented in the Java language mainly, for its intrinsic web-orientation and portability of code. In each site, a series of processes/servlets are running performing various application tasks (see figure 3).

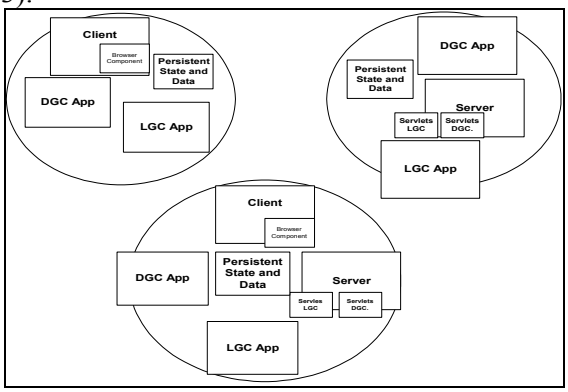

Figure 3: Configuration of the system: client-only, server only, and client-server in one site

4.1 Client

The client for this system makes use of a browsing component described later. It is capable of conducting regular web browsing sessions interleaved with news import operations. These operations are implemented by exchanging messages with the server through the use of a URL connection and saving reply data from the server.

4.1.1 Browsing Component

To speed up developing time for the system prototype, we used a browsing software component[1] (Microsoft's Internet Explorer [explorer]). Others like HotJava Web Browser

[1] Since it is not in the work goals to develop our own browsing tool from scratch.

component [hotjava] could have been used. Basically, it is used just for presentation purposes. Every information important to the system is included in the messages exchanged.

4.2 Garbage Collector

There is one local component of the GC algorithm in each process. It runs as a separate process in the system and can be continuously active, or run periodically (due to free space requirements) or by user demand. This component is responsible for effectively deleting objects when they are no longer reachable.
A distributed GC component runs on each process. These components communicate with each other in order to perform the GC specific distributed operations according to the algorithm [ferreira98].

4.3 Concurrency and Causality Issues

Application and GC specific data at a site is manipulated by the browser, and by the local and distributed component of the garbage collector[2]. To ensure data integrity, access to these structures must be synchronised. This is achieved by means of file locking.
In addition, to ensure algorithm correctness and to avoid race conditions, GC specific data structures must be accessed atomically, i.e. they cannot be modified in a record-by-record basis by different processes. Therefore, a generation mechanism must be used allowing different parts of the GC mechanism to manipulate different versions of this data. Due to the nature of the algorithm, this does not alter algorithm correctness nor it alters its liveliness. Storage reclaiming may, in spite, be delayed. This mechanism is based in vector clocks between sites with present and past[3] communication.

4.4 Persistency Issues

Every piece of algorithm information is maintained in persistent storage. Due to the concurrent nature of the browsing process, the local collector and the distribute collector, care must be taken in synchronising access to persistent data. Application state data is synchronised as any other data in the system.

5. Conclusions

We believe that for some web applications it is important to ensure referential integrity at the system level. The reason is that it is not acceptable to find dangling references and to have storage leaks. These situations result from the error-prone memory management done by webmasters.
We presented and implemented a web news service in which referential integrity is ensured at the system level. This is achieved by using a distributed garbage collection algorithm.
So far, our experiments show an overall performance penalty, due to the cost of maintaining referential integrity, less than 10%. We expect to improve this given that it is our first prototype. We also plan to investigate the issue of fault-tolerance, i.e. how the referential integrity can be maintained in spite of communication and site failures.

6. References

[explorer] Reusing IExplorer Technology http://msdn.microsoft.com/workshop/

[Ferreira98] Ferreira, Paulo and Shapiro, Marc, ,1998. Modelling a Distributed Cached Store for Garbage Collection: the algorithm and its correctness proof, ECOOP'98, Brussels, Belgium, July 1998

[hotjava] HotJava Web Browser Component http://java.sun.com/products/hotjava/

[Perdis00] PerDiS: design, implementation, and use of a PERsistent DIstributed Store Paulo Ferreira, Marc Shapiro, et al., 2000. Book chapter in Recent Advances in Distributed Systems, eds. S. Krakowiak and S.K. Shrivastava, Springer Verlag LNCS vol. 1752, Feb. 2000

[2] Both the application itself and the Web Servlets which receive distributed collector messages

[3] Old, outdated data will be removed in background