IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP4974

DM²P: Dynamic MultiMedia Proxy

Mouna Kacimi
Laboratoire Electronique, Informatique et Image (LE2I), University of Bourgogne, BP 47870 21078 Dijon CEDEX,
kacimi@khali.u-bourgogne.fr

Richard Chbeir
Laboratoire Electronique, Informatique et Image (LE2I), University of Bourgogne, BP 47870 21078 Dijon CEDEX, rchbeir@u-bourgogne.fr

Kokou Yetongnon
Laboratoire Electronique, Informatique et Image (LE2I), University of Bourgogne, BP 47870 21078 Dijon CEDEX, kokou@u-bourgogne.fr

ABSTRACT

Multimedia applications are increasingly popular in the internet. They require large volume of data storage and appropriate user centric retrieval methods. To reduce the network delays experienced by these emerging applications, various caching and proxy techniques are required. We present here a dynamic multimedia proxy method able to take into account the user profiles. The profiles are used to match proxy storage and processing capacities to user demands. The proxies are organized in profile groups composed of a set of proxies cache and treatment proxies cache which control query execution. Another key feature of the approach is a routing profile is used to provide proxies with a global network view and to determine for a query the appropriate proxies that match the user profile. One goal of the approach is to dynamically adapt to evolving network connectivity by redistributing caching functionalities over the network according to servers capacities and users profiles. We present also our prototype DM²P implemented to validate our approach.

INTRODUCTION

Multimedia applications are fast gaining in popularity since a couple of years. As a result, there is a growing demand for various types of data accessible via the internet and for greater network bandwidth. These data require large volume of disk space and new indexing and retrieval methods. To reduce network latency and improve user response delays, various techniques have been proposed such as web caching used to preload or *prefectch* data in order to anticipate user demands. The data can be cached on either clients (web browsers) or intermediate servers (proxies) located near the users.

As the Web has become a significant source of various types of information, many efforts have been directed towards extending the original research to define cooperative caching methods and architectures for web-based applications [1, 3, 4, 5, 6, 9, 10, 11]. Two main categories of web based caching solutions can be distinguished, the hierarchical web caching and the distributed web caching.

In a hierarchical web caching architecture [3], the caches are organized in several levels. The bottom level contains client caches and the intermediate levels are devoted to proxies and their associated caches. When a query is not satisfied by the local cache, it is redirected to the upper level until there is a hit at a cache. If the requested document is not found in any cache, it is submitted directly to its origin server. The returned document is sent down the cache hierarchy to the initial client cache and a copy is left on all intermediate caches to which the initial user requests were submitted. Hierarchical caching has many advantages; it avoids redundant retrieval of documents from data servers, reduces network bandwidth demands and allows the distribution of document accesses over a caches hierarchy. Despite its advantages, hierarchical web caching exhibits several drawbacks. The high level caches, particularly the root cache, are bottlenecks which can significantly degrade the performance of the system. The failure of a cache can affect the system fault tolerance. Several copies of the same document are stored at different cache levels which is very storage expensive and restrictive espacially when treating multimedia data. Moreover, there is a lack of direct links between sibling caches of the same level.

The distributed web caching approach reduces hierarchical links between caches. Several distributed caching approaches have been proposed to address one or more problems associated to hierarchical caching. In [6], the authors propose an extension of the hierarchical caching where documents are stored on leaf caches only. The upper level caches are used to index the contents of the lower level caches. When a query cannot be satisfied by the local cache, it is sent to the parent cache that indicates the location of the required documents. In [4], Li Fan et al. propose a scalable distributed cache approach, called Summary Cache, in which each proxy stores a summary of its cached documents directory on every other proxy. When a requested document is not found in the local cache, the proxy checks the summaries in order to determine relevant proxies to which it sends the request to fetch the required documents. Two major problems restrain the scalability of summary cache approach. The first problem is the frequency of summary updates which can significantly increase inter-proxy traffic and the bandwidth usage. The second problem is related to the storage of the summaries especially when the number of cooperating proxies is important. Aner Armon and Hannoch Levy in [1] investigate Cache Satellite Distribution System which comprises ***P*** proxy caches and a central station. The central station periodically receives from the proxy caches reports containing information about user requests. The central station uses this information to foresee what documents could be required by other proxy caches in the near future. It selects a collection of popular Web documents and broadcasts the selected documents via satellite to all or some of the participating proxies. There are two important advantages of this proposal: (i) it anticipates the user requests; (ii) it allows the collaboration between proxies independently from the geographical distance of a satellite. However, the central station used leads to a weak fault-tolerance.

One of the drawbacks of current caches techniques is that they are too restrictive. They only provide a static architecture not adaptable to the network evolution (new materials, fault tolerance, etc.) and user demands evolution (new users, new profiles, etc.). For instance, the disconnection or the connection of a proxy (even if it was the root one) on the network should be managed dynamically in function of the network traffic and servers capacities. Furthermore, when defining web caching schemes for multimedia applications, a major issue should be addressed which is the optimisation of the storage capacities to alleviate the exchange of large volume of data. Many questions can be raised:

- store the whole or only part of a multimedia document in the cache?
- store only textual data together with metadata based description of media types in the cache?
- integrate descriptions of multimedia and extraction methods in the cache?
- use the principle of media autonomy to group together all images, textual documents, and audio data and load them in separate caches?

This conference paper appears in the book, Innovations Through Information Technology, edited by Mehdi Khosrow-Pour. Copyright © 2004, Idea Group Inc. Copying or distributing in print or electronic forms without written permission of Idea Group Inc. is prohibited.

In this paper, we address these issues and present a dynamic multimedia proxy scheme based on defining user and proxy profiles which are used to match the capacities of the proxies to the user demands. A user profile consists of a set of themes (or subjects); while a proxy profile is composed of one or more themes with in addition a description of the hardware and softwares of the proxy. Users with the same interests (or themes) can easily and quickly retrieve the corresponding documents from one or several proxies having the same profile. Depending on their storage and processing capabilities, two types of proxies are distinguished:

- Cache proxies are primarily used to store multimedia data and various data indexes that are required for data retrieval.
- Treatment proxies caches in addition to storing data are used to manage a group of proxies cache with the same profile.

A key feature of our approach is the *routing profile table*. It is an extension of the traditional network routing table used to provide a global network view to the proxies. When a request is submitted to a (cache or treatment) proxy, the routing profile table is used to determine a treatment proxy cache that is the best match with the user profile. Another important feature of the approach is the ability to dynamically adapt to evolving network connectivity: when a proxy is connected to (or disconnected from) a group, we define different schemes for updating the routing profile table and the contents of the corresponding caches. Furthermore, our approach stores data or/and metadata in function of storage capacities of each proxy (for instance, if storage capacities are minimum, only textual and metadata are cached).

A proptotype DM²P has been implemented to validate our approach. We show here how it works.

The remainder of the paper is organized as follows. Section 2 presents the Dynamic MultiMedia Proxy (DM²P) approach. Section 3 is devoted to the presentation of DM²P functionalities. And finally section 4 concludes the paper and snapshots our future work.

DYNAMIC MULTIMEDIA PROXY

To improve multimedia data retrieval, anticipate end-user queries, and enhance network performances, our approach consists of:

- Organizing machines by profile groups (Figure 1.a) Therefore, the users and the caches having the same interests can easily and quickly exchange related documents.
- Optimizing the storage by selecting only profile-related documents on caches.
- Considering the machines capacities in terms of storage, treatment and communication
- Providing a quick and pertinent answer,
- Maintaining a global vision on each proxy of the network in order to optimize traffic and quickly forward queries to appropriate proxies

Profile

In our approach, two types of profile are used: the ***User Profile*** and the ***Proxy Profile,*** defined as follows:

- **User Profile (UP)**: describes the user interests and preferences in terms of themes (sport, cinema, news, etc.). The UP can be determined using either direct methods (classical formulary where the user indicates his interests) or indirect methods (cookies, spy wares, etc.).
- **Proxy Profile (PP)**: comprises two components: the one or more themes that can match the user profiles and the hardware and software characteristics (disk capacity, IP address, etc.). The associated themes depend upon documents stored in the proxy cache. For example, if the proxy contains medical images, its PP will include {health, medical, etc.}. Associated themes component is defined using various extraction methods that are applied on concerned directory data and corresponding metadata of the proxy cache. Each type of media (image, text, video, etc.) and each application domain necessitate different extraction methods (this issue will not be addressed in this paper). To include a theme T in the PP index, we calculate the number of related documents: if this number is greater than a Minimum Threshold occurrence (MTO), the corresponding theme T is registered, otherwise, it is ignored. The MTO is defined by the end-user or the administrator.

In this manner, we define the profile using the standard CCPP [8] as the union of the ***user profile*** and the ***proxy profile***.

Proxy types

In LAN or WAN, machines have different capacities in terms of treatment, storage, and communication. In addition, some machines may have very specific treatments depending on the activities of the institution. For this reason, we have defined two types of proxies: ***proxy cache*** and ***treatment proxy cache***.

Proxy cache

A Proxy cache is identified as a machine which has low capacities to run parallel or specialized treatments, and to manage communications between the proxies. Its main task is to store multimedia data and other meta-data such as local index, neighbor index, routing profile table, and document description (these concepts will be detailed below). The proxy cache has also several other background tasks like retrieving data from the local cache, sending a query to treatment proxies cache, loading data from another cache, etc. A proxy cache can be connected to several treatment proxies cache, depending on its profile and connection time.

Treatment proxy cache

A treatment proxy cache is identified as a powerful machine with high capacities of storage, treatment and/or communication. In addition to storage capacity, its main task is to manage a set of cache proxies of the same profile and to maintain the index of their content. As shown in Figure 1.a, each group is managed by at least one treatment proxy cache.

Index and Routing profile table

In our approach, each proxy must recognize its cache content and its environment. The processes of recognition and multimedia documents retrieval are based on a set of indexes allowing to group the proxies. There are three types of indexes: local index, neighbor index and routing profile table. To retrieve a multimedia document, the corresponding documents are first searched in the related proxy caches. If there are no hits, this latter proxy forwards the query to the one of its managing treatment proxies which checks the index of associated proxies' cache in order to locate relevant documents.

Local index

A local index is used to process user queries and to fetch documents in the local cache. It is located on proxy cache (and treatment proxy). It is represented as a tree (Figure 1.b) where intermediate nodes contain themes and end-leaves contain document descriptors and one of the following: document abstract, whole document, or document link. The end-leaf content depends upon storage capacities. For instance, a low storage capacity proxy would contain documents descriptors and links only, while a high storage capacities proxy can contain documents descriptors and the full documents.

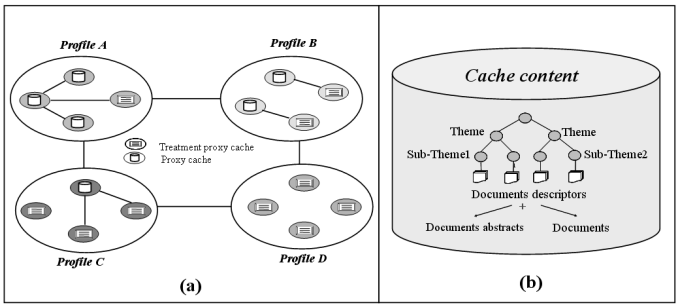
Neighbor index

A neighbor index structure changes according to the proxy type:

- On the proxy cache: A neighbor index allows forwarding queries towards treatment proxies in function of the profile and the connection time. Each line of this index contains a treatment proxy (Name and IP address), its corresponding profiles, and the connection cost defined by its latency.

Copyright © 2004, Idea Group Inc. Copying or distributing in print or electronic forms without written permission of Idea Group Inc. is prohibited.

Figure 1: profile groups and proxy types



Example

Treatment Proxy Name	IP Address	Profile	Cost
T1	10.2.3.10	P1	1
T3	10.2.3.25	P2	2

This proxy cache is connected to the treatment proxy T1, by the profile P1 with a cost of 1 unit of time. It is connected to T3 by the profile P2 with a cost of 2 units of time.

- On the treatment proxy: A neighbor index contains the union of the local and neighbor indexes of all its related proxies' cache. The role of the neighbor index in this level is to allow a quick location of required multimedia documents.

Routing Profile Table (RPT)

The routing profile table is an extension of classical routing table which gives a global view of the network. It contains a list of couples ***(treatment proxy, profile)*** which allows a proxy cache to choose, in function of a profile, the treatment proxy to which queries will be sent. If the query belongs to a foreigner profile (not managed by the group of the dispatcher proxy), it is sent to an appropriate proxy using the routing profile table. The RPT is located on each proxy (cache and treatment) and also used for connection and disconnection purposes.

As we evoked previously, the proxies grouping is done in function of the profile and the cache neighborhood. The cache neighborhood is defined on the basis of the latency: the cache ***A*** is a neighbor of a cache ***B***, if the latency time between ***A*** and ***B*** is lower than the latency time between ***A*** and the original server.

DM²P PROTOTYPE

To validate our approach, we have implemented a prototype called DM²P (Dynamic MultiMedia Proxy). For portability reasons, DM²P uses Java language. We defined a set of XML protocols that warrant standard communications between proxies. MySQL DBMS is used to store profiles.

At DM²P execution, the administrator may choose the proxy type in function of the machine capacities, the application domain, and the network requirements. In essence, we are studying probabilistic approaches in order to automate this step when necessary. After that, users can subscript and define their profiles. In this manner, the proxies are grouped in function of related profiles. The updating process of proxy grouping is done at the connection and disconnection moments. In this paragraph, we present how our prototype addresses the network evolution and proxies capacities (connection and/or disconnection, load balance, etc.)

Proxy connection

When a proxy cache is connected to the proxies' network, the choice of the appropriate treatment proxie(s) is done as follows:

- It downloads the actual routing profile table from any connected proxy (cache or treatment). From the RPT
- It tries to extract all the treatment proxie(s) that manage(s) its profile (already analyzed and defined using manual or automatic extraction methods):
  - If there no treatment proxy which manages the profile of the connected proxy cache, this later will be added to the group having the lowest charge and latency time.
  - Otherwise, the connected proxy cache selects among the concerned treatment proxies those having the lowest latency. When there are several treatment proxies with the same latency time, the one having the lowest charge is chosen.

To compute the latency in DM²P, we simply use a cost module based on the "ping" command. We are currently studying other algorithms proposed in the literature to improve this issue. This command gives the average time of packet transmission between two machines on the network.

Proxy disconnection

When a proxy cache is normally disconnected, all its treatment proxies update their summaries, because the content of the disconnected proxy cache will not be accessible any longer. In the same manner, when a treatment proxy is disconnected, all its links inside its cache proxies will be deleted. After, we apply the process presented before to connect independent cache proxies to other treatment one(s). If the disconnected treatment proxy is the last treatment one on the network, one of the remaining proxy cache changes its type (by comparing its capacities to the others'), and becomes a treatment proxy temporally (Figure 3). The network administrator (when exists) is advised in this situation.

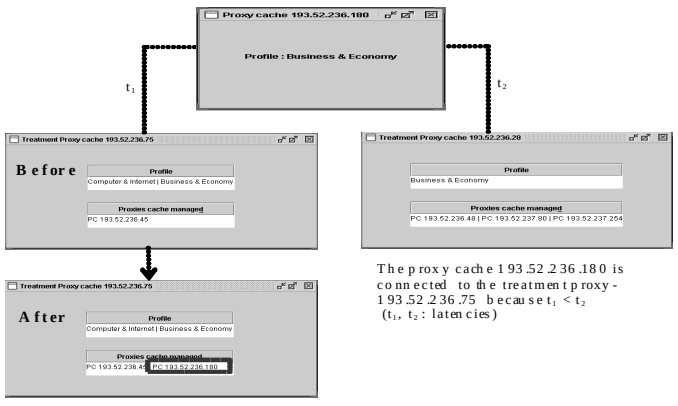
Load balance

In order to maintain high performances, we use proxy cache load balance and cache update. Two types of load balances are introduced: the proxy load balance and the profile load balance. The first type consists of sharing the proxies management between the treatment proxies having common profiles. The idea of the second type is that each treatment proxy manages a set of profiles and shares them with different treatment proxies. To avoid bottleneck and minimize network traffic, this balance is done at the connection and disconnection moments (figure 3).

Profile cache update

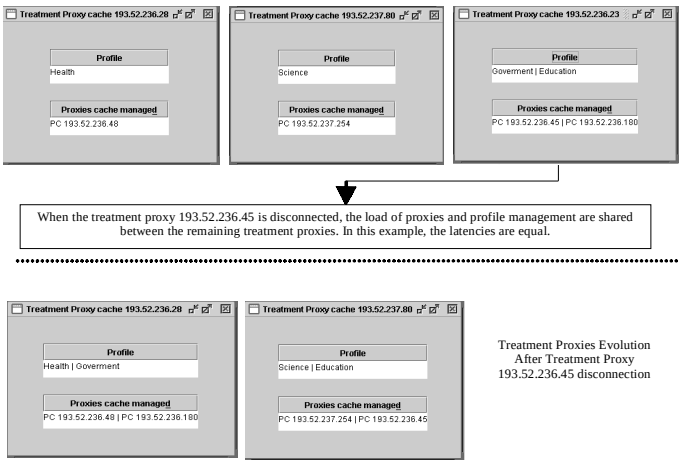
To guarantee data coherence and consistence, a profile cache update is used. At connection, the user may have a profile completely different from the proxy cache profile to where he must be connected. In this case, the proxy will be connected to two groups of profile. The cache profile changes in function of documents loaded by the user. It means that, if the cache is overloaded, the documents of cache profiles will be substituted by documents belonging to the user profile. In this

Figure 2: Example of proxy connection



Copyright © 2004, Idea Group Inc. Copying or distributing in print or electronic forms without written permission of Idea Group Inc. is prohibited.

Figure 3: Example of proxy disconnection and load balance



manner, the cache profile will converge to the user profile and the proxy membership of profile groups can become inconsistent after several changes. For this reason, the profile cache update is carry out in a regular time interval. This latter must be shorter if the proxy environment is very dynamic. In our approach the choice of the update time interval is done either by the administrator or by the end-user.

CONCLUSION

In this paper, we presented a dynamic multimedia proxy approach that:

- uses profiles to match user demands and proxy cache and process ing capabilities
- defines a routing profile table to provide the proxies with a global up-to-date view of network connectivity
- optimizes cache storage and balances storage and processing load by addressing the user profile
- achieves network adaptability through dynamic update of routing tables and proxy migration

We also present a couple of functionnalities of our prototype called DM²P (Dynamic MultiMedia Proxy). The evaluation phase is ongoing and the preliminary results are very satisfactory and are being used to tune the current prototype.

Our future directions are various. First, real life case studies are needed to deploy our approach. Another direction is to integrate fault tolerance techniques in order to automatically resolve abnormal proxy disconnection situations. Other issues will need addressing concern the definition of performance model and analysis tool to take various network parameters into consideration.

REFERENCES

[1] Aner Armon, Hanoch Levy. *Cache Satellite Distribution Systems: Modeling and Analysis.* In IEEE INFOCOM 2003.

[2] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. *Web caching and Zipf-like distributions: Evidence and implications*. In Proceedings of IEEE Infocom, New York, NY, March 21-25, 1999. p.126-134.

[3] A. Chankhunthod, P. Danzig, C. Neerdaels, M. Schwartz, and K.Worrell. Hierarchical internets object cache. *Proceedings of the USENIX Technical Conference*, San Diego, California, January 1996.

[4] Li Fan, Pei Cao, Jussara Almeida, and Andrei Z. Broder, "*Summary cache: a scalable wide-area Web cache sharing protocol*," IEEE/ACM Transactions on Networking, 2000, vol. 8, no. 3, p. 281-293.

[5] S. Paul and Z. Fei. *Distributed caching with centralized control.* Computer Communications journal, 2001, vol 24, n°2, p. 256-268.

[6] D. Povey and J. Harrison, "*A Distributed Internet Cache*", In Proceedings of the 20th Australian Computer Science Conference, Sydney, Australia, February 1997, p. 175-184.

[7] P. Rodriguez, C. Spanner, and E.W. Biersack, *Analysis of Web Caching Architectures: Hierarchical and Distributed Caching*, IEEE/ ACM Trans. on Networking 2001

[8] W3C. Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies. 2001. Available at <http://www.w3.org/TR/2001/WD-CCPP-struct-vocab-20010129/> (last visited 2003/09/29)

[9] Yang-Hua Chu, Sanjay Rao, and Hui Zhang. *A case for end system multicast.* In Proceedings of ACM Sigmetrics, Santa Clara, CA, 2000, p 1-12.

[10] Abdullah Abonamah, Akram Al-Rawi, Mohammad Minhaz, "*A Unifying Web Caching Architecture for the www*", Zayed University, Abu Dhabi, UAE. IEEE ISSPIT, Maroc, Jan 2003, pp.82-94.

[11] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, L. Zhang, "*IDMaps: A Global Internet Host Distance Estimation Service*", IEEE/ ACM Transactions on Networking, 2001,vol 9, n°5, p.525-540.

Copyright © 2004, Idea Group Inc. Copying or distributing in print or electronic forms without written permission of Idea Group Inc. is prohibited.

0 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the publisher's webpage:
www.igi-global.com/proceeding-paper/dm2p-dynamic-multimedia-proxy/32384

Related Content

Existential Aspects of the Development E-Culture
Liudmila Vladimirovna Baeva (2018). *Encyclopedia of Information Science and Technology, Fourth Edition (pp. 4189-4198).*
www.irma-international.org/chapter/existential-aspects-of-the-development-e-culture/184126

Capacity for Engineering Systems Thinking (CEST): Literature Review, Principles for Assessing and the Reliability and Validity of an Assessing Tool
Moti Frank (2009). *International Journal of Information Technologies and Systems Approach (pp. 1-14).*
www.irma-international.org/article/capacity-engineering-systems-thinking-cest/2543

Impact of the Learning-Forgetting Effect on Mixed-Model Production Line Sequencing
Qing Liuand Ru Yi (2021). *International Journal of Information Technologies and Systems Approach (pp. 97-115).*
www.irma-international.org/article/impact-of-the-learning-forgetting-effect-on-mixed-model-production-line-sequencing/272761

Nth Order Binary Encoding with Split-Protocol
Bharat S. Rawal, Songjie Liang, Shiva Gautam, Harsha Kumara Kalutarageand P Vijayakumar (2018). *International Journal of Rough Sets and Data Analysis (pp. 95-118).*
www.irma-international.org/article/nth-order-binary-encoding-with-split-protocol/197382

An Outline of Approaches to Analyzing the Behavior of Causal Maps
V. K. Narayananand Jiali Liao (2005). *Causal Mapping for Research in Information Technology (pp. 388-377).*
www.irma-international.org/chapter/outline-approaches-analyzing-behavior-causal/6526