Intelligent Exploitation of Cooperative Client-Proxy Caches in a Web Caching Hybrid Architecture[†]

Maha Saleh El Oneis, Mohamed Jamal Zemerly and Hassan Barada
Khalifa University of Science, Technology, and Research
United Arab Emirates

1. Introduction

The technological evolution witnessed by the world today has made the migration of services and information to the World Wide Web (WWW) faster and easier. This, in turn, made the number of Internet users increase exponentially. Such increase in number of users, and the demand of information and services resulted in a variety of problems that affected the user's comfort in using and retrieving information from the Web. Despite all the advanced technologies used today, two main problems are still faced which are *server overloading* and *network congestion*. Network congestion can occur when a network link is carrying too much data that would affect its quality of service. Server overloading happens when the server receives more service requests than it can handle. Many researchers have tackled these issues since the early 90's and some helpful solutions have been implemented. One of the most effective solutions that alleviate server overload and reduce network congestion *is web caching*. Web caching is the process of saving copies of content (obtained from the Web) closer to the end user, in order to reduce bandwidth usage, prevent server overload, as well as reducing the user’s perceived latency. These studies have resulted in the development of a web caching notion which can be implemented at three levels. Level1 (L1) cache is known as the client caching which takes place at the browser level. Level2 (L2) takes place at the proxy level while Level3 (L3) is the cooperation of the proxies in sharing cached objects among the cooperation set (Dykes & Robbins, 2002). Researchers have agreed that caches on the client browser and proxy level can significantly improve performance (Abrahams et al., 1995). In addition, many studies encouraged the broad use of web caching within organizations that provide internet service to users (Korupolu & Dahlin, 1999; Gadde & Robinovich, 1999; Wolman & Voelker, 1999; Lee et al., 2001). Such studies helped in considering the possibility of constructing a large number of web caches from cooperative proxies and client caches (Zhu & Hu, 2003) to introduce a new cooperation level.

† This project is funded by Intel Innovation Center, Dubai, U.A.E.

A range of studies agreed on the benefits of web caching and its major contribution to Internet services. Still the rapid growth of internet traffic and users has made us witness rapid improvements on the broadband services. Nowadays, Internet Service Providers (ISPs) are offering better broadband networking technologies, but still many residential and small-business users are using low-bandwidth connections. Any near promise of the availability of such broadband technologies for users in rural areas is still uncertain because of the associated high cost. However, even with the availability of high bandwidth, there are types of information such as multimedia that always demand more bandwidth. For example when YouTube became very popular, one of the Internet Service Providers (ISP) had huge increase in the amount of information entering the network and an increase in the user's perceived latency. When the problem was observed closely, the ISP discovered that 1Gb/s in the network was consumed by only one website, YouTube.com. In addition to the obvious benefits of web caching, some of the important properties desired in a web caching scheme are fast access, robustness, transparency, scalability, efficiency, adaptivity, stability, load balanced, ability to deal with heterogeneity, and simplicity (Wang, 1999).
Our area of interest in building a new hybrid web caching architecture is to reduce the client latency period in retrieving WWW information in rural areas as well as improving the performance of the broadband technology. This architecture explores and benefits from the free space offered by the client's caches when they are connected to the internet, and reduces the load on the upper tier (proxies & web) servers.
With the exponential growth of the internet, a single web cache is expected to become a hot spot. If the solution is to add more hardware, we can expect a point where no hardware is sufficient enough, and the managment of such number of extra harware is a burden in different aspects.

2. Related Work

Ever since web caching has been found as a solution for network congestion and server overloading, different caching architectures were proposed to ease the process of delivering the requested data through inter-cache cooperation. The next sections discuss some of the most common web caching architectures proposed in recent years. We classify these architectures into hierarchical architecture, distributed architecture, and hybrid architecture.

2.1 Hierarchical Caching Architecture

The idea behind constructing a hierarchical cache is to arrange a group of caches in a tree-like structure and allow them to work together in a parent-child relationship to fulfil the requested objects by the client. If a hierarchical structure is arranged properly, the hit ratio can be increased significantly (Wang, 1999).
In a hierarchical caching architecture, caches are placed at various levels of the network. As shown in Figure 1, there is a client's cache, institutional cache, regional cache, national cache, and at the top is the original server. When a client requests a page, it first checks its browser cache. If the request is not fulfilled, then it is forwarded to the institutional cache. If the request is not satisfied by the institutional cache, then it is passed to the regional cache. If the request is not found at the regional cache, then it is redirected to the national cache. The national cache forwards the request to the original server if it cannot fulfill the request.

When the object is found in a cache or the original server, it travels down the hierarchy and leaves a copy of the object in each caching level in its path to the client.

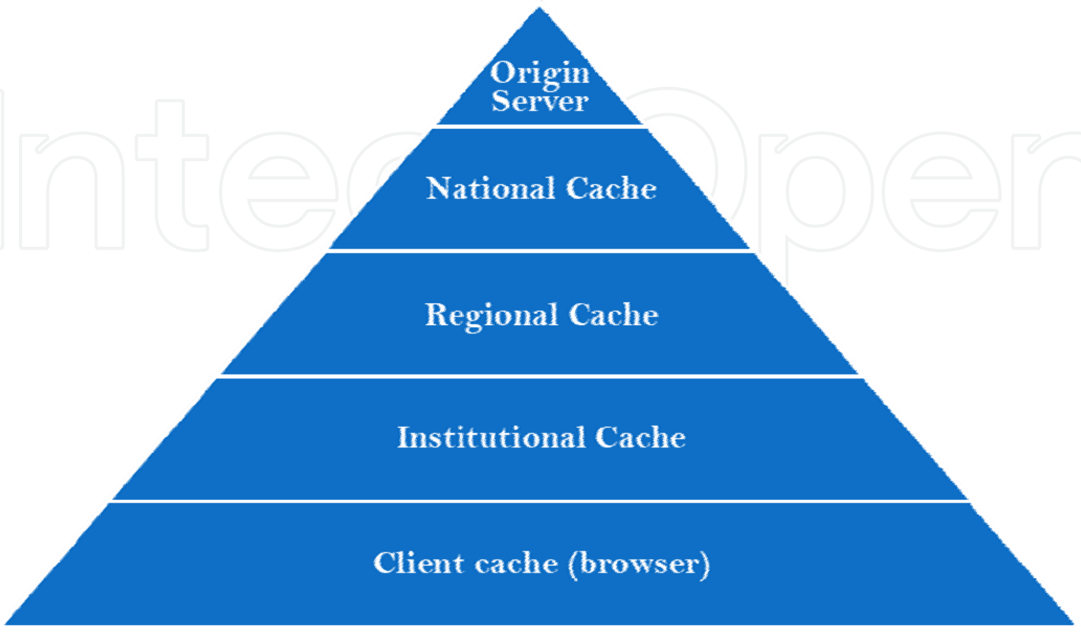


Fig. 1. Hierarchical Caching Architecture

2.2 Distributed Caching Architecture

Researchers have proposed an alternative to the hierarchical caching architecture and eliminated the intermediate tiers except for the institutional tier. All caches in that tier contain meta-data about the content of every other cache. Another approach proposed in this architecture is to employ the hierarchical distribution mechanism for more efficient and scalable distribution of meta-data (Wang, 1999). Figure 2 illustrates this approach.
In a distributed caching architecture that employs the hierarchical distribution mechanism, the layers that contain cached objects are only the client and institutional layers. Other layers contain information about the contents of the caches in the institutional layer.

2.3 Hybrid Caching Architecture

A hybrid scheme is any scheme that combines the benefits of both hierarchical and distributed caching architectures. Caches at the same level can cooperate together as well as with higher-level caches using the concept of distributed caching (Wang, 1999). A rough comparison between hierarchical, distributed, and hybrid caching architectures is shown in Table 1.
A hybrid caching architecture may include cooperation between the architecture's components at some level. Some researchers explored the area of cooperative web caches (proxies). Others studied the possibility of exploiting client caches and allowing them to share their cached data. One study addressed the neglect of a certain class of clients in researches done to improve Peer-to-Peer storage infrastructure for clients with high-bandwidth and low latency connectivity. It also examines a client-side technique to reduce the required bandwidth needed to retrieve files by users with low-bandwidth. Simulations

done by this research group has proved that this approach can reduce the read and write latency of files up to 80% compared to other techniques used by other systems. This technique has been implemented in the OceanStore prototype (Eaton et al., 2004).

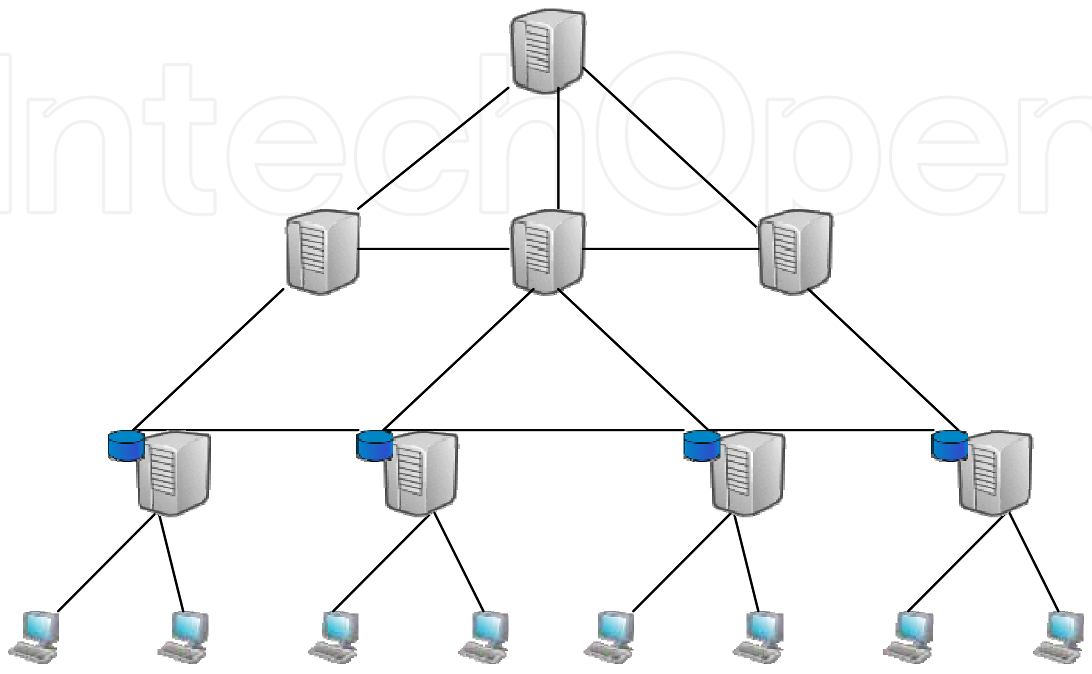


Fig. 2. Distributed Caching Architecture

Another study proposed an internet caching service called CRISP (Caching and Replication for Internet Service Performance). The problems that CRISP tried to solve are the performance, scalability, and organizational problems in large central proxy caches. The idea behind CRISP is to connect cooperative proxies through a mapping service which contains information about cached URLs in each proxy in the cooperative set. A drawback in this structure is the failure of the centralized mapping service. This drawback is solved by forcing the proxies to work individually without letting the user feel the impact of the failure except with the increase in the perceived latency. The study claims that this simple design and implementation of cooperative caching is effective and efficient for distributed web caches serving tens of thousands of users (Gadde & Robinovich, 1997).
Another study was motivated by the studies that have shown that limited bandwidth is the main contributor to the latency perceived by users. And also the fact that the majority of the population was still using modems at that time. An approach to reduce user-perceived latency in limited bandwidth environments is investigated. It explores a technique based on prefetching between caching proxies and client browsers. The idea is that the proxy would predict what the user might request/access next, where it invests the user's idle times while checking the result of the previous request and predicts what the user will request next. The proxy would push its prediction result to the client's browser cache, noting that the proxy will only use the contents of its cache in this prediction. The result of this investigation showed that prefetching between low-bandwidth clients and caching proxies combined with data compression can reduce perceived user latency by over 23% (Fan et al., 1999).

One analysis used a trace-based analysis and analytic modelling to put inter-proxy cooperation into the test and examine its performance in the large-scale WWW environment. It examines the improvement that such cooperation can provide in a 200 small organizations' proxies environments, as well as with two large organizations handling 20,000 and 60,000 clients. However the modeling considered a much larger population containing millions of users. The overall studies and examination done in this paper concluded that cooperative caching has performance benefits only within limited populations (Wolman & Voelker, 1999).
Another examination explored the benefits of the cooperation among proxies under different network configurations and user access patterns. This was achieved by classifying and analysing these cooperation schemes under a unified mathematical framework. These analytical results were validated using a trace-driven simulation. Using the results from the simulation and analysis, the following was concluded:

- Proxy cooperation is beneficial when the average object size is large and the working set does not fit in a single proxy. Such benefit also depends on the cluster configuration.
- The cooperation between proxies, where each proxy serves missed requests from other proxies in its cooperation set, is mostly sufficient when the users' interests are highly diverse.
- The cooperation of the proxies in object replacement decisions would result in more benefits when the user accesses are dense and the requests are focused on a small number of objects.

Overall, the benefit of the cooperation among proxies is dependent on a number of factors including user access, user interest, and network configuration (Lee et al., 2001).
Another study presented a decentralized, peer-to-peer web cache called Squirrel that uses a self-organizing routing substrate called Pastry (Rowstron & Peter Druschel, 2001) as its object location service. The key idea is to enable web browsers on desktop machines to share their local caches, to form an efficient and scalable web cache, without the need for dedicated hardware and the associated administrative cost. An evaluation of a decentralized web caching algorithm for Squirrel is also provided. Studies discovered that it exhibits performance comparable to a centralized web cache in terms of hit ratio, bandwidth usage and latency. It also achieves the benefits of decentralization, such as being scalable, self-organizing and resilient to node failures, while imposing low overhead on the participating nodes. Squirrel tested two different schemes called the home-store and directory schemes on a LAN organization. Performance studies have shown that the home-store scheme depicts less overhead on the serving nodes; this approach works for load balancing among the peer-to-peer nodes (Lyer et al., 2002).
Another research proposes a more effective use of caching to cope with the continuing growth of the internet. This proposal is to exploit client browser caches in cooperative proxy caching by constructing the client caches within each organization as a large peer-to-peer client cache. The goal of this study is to investigate the possibility and benefit of constructing such large peer-to-peer client cache in improving the performance of the internet. In this architecture, clients can share objects cached not only at any proxy in the cooperative set but also at any neighbour's cache connected to the network. After doing some simulations with/without exploiting client caches, results have shown that exploiting

client caches can improve performance significantly. It also introduces a hierarchical greedy dual replacement algorithm which provides cache coordination and utilizes client caches (Zhu & Hu, 2003).
Another study presented the design and implementation of a previously proposed scheme based on a novel Peer-to-Peer cooperative caching scheme. It considers new means of communication between cooperative caches. It also proposes and examines different routing protocols for data search, data cache, and replication of data. The results of the performance studies show the impact of cache coherency on the system's performance. It also shows that the proposed routing protocols significantly improve the performance of the cooperative caching system in terms of cache hit ratio, byte hit ratio, user request latency, as well as the number of exchanged messages between the caches in the cooperative set (Wang & Bhulawala, 2005).
Yet another study presented a trustable peer-to-peer web caching system, in which peers in the network share their web cache contents. To increase the trust-level of the system, they have proposed to use sampling technique to minimize the chance of distributing fake web file copies among the peers. They further introduce the concept of opinion to represent the trustworthiness of individual peer. A prototype has been built and the experimental results demonstrated that it has fast response time with low overhead, and can effectively identify and block malicious peers. This paper proposed a reasonable solution in locating the cached object using a search history similar to a log file that is stored in each peer. Each peer might carry a different log of search history of the peers in the system. When a request is initiated by a client and a cache miss was returned from its local cache, the request is forwarded to the client's nearest neighbour. If a cache miss occurred then this neighbour will look into the search history and find out which was the last peer that initiated a request to the same object and connect to that peer. This is an efficient solution but it would be rather faster if the client looks into the search history log it has before connecting to the neighbour in the first place. At the same time it can check if one of the peers that requested this object is any of its neighbours. Even though many papers have discussed the issue of trust between the peers, and some suggested "building trust" approach between peers, this issue is still largely unresolved and in need of further investigation (Liu et al., 2005).

3. Proposed Architecture

The proposed architecture is a cooperative client-client, client-proxy, proxy-proxy caching system that aims to achieve a broadband-like access to users with limited bandwidth. The proposed architecture is constructed from the caches of the connected clients as the base level, and a cooperative set of proxies on a higher level, as shown in Figure 3. The construction of the large client web cache is based upon some of the novel peer-to-peer (P2P) client web caching systems, where end-hosts in the network share their web cache contents.

3.1 Desired properties in the proposed architecture

The proposed architecture is based upon the idea of a hybrid scheme. It consists of two tiers of cooperative caches: client caches and proxy caches. The properties that we wanted to achieve while designing the architecture are as follows:

- Slight congestion in the parent caches.
- Low latency and data transmission time.
- Evenly distributed network traffic for faster transmission time and low latency achievement.
- Long connection times.
- Low bandwidth usage which is the priority in this architecture along with the low latency property.
- A maximum of two hierarchical levels.
- Low disk space usage therefore low duplication of objects.
- Maintain an easy plan to keep the cached objects fresh.
- Test different object retrieval approaches to achieve a high to a very high hit ratio and grant the user a fast response time.

Features	Hierarchical	Distributed	Hybrid
Parent caches	Congested	slight congestion	slight congestion
Latency	high	low	low
Connection times	short	long	long
Bandwidth required	low	high	low
No. of Hierarchies	<4	1	one - two
Transmission time	high	low	low
Network traffic	Unevenly distributed	Evenly distributed	Evenly distributed
Disk space usage	Significant	low	low
Placement of caches in strategic locations	vital	Not required	up to ISP
Freshness of cached contents	difficult	easy	easy
Hit ratio	High	Very high	high - very high
Response time	moderate	fast	fast
Duplication of objects	high	low	low

Table 1. Comparison between hierarchical and distributed caching architectures

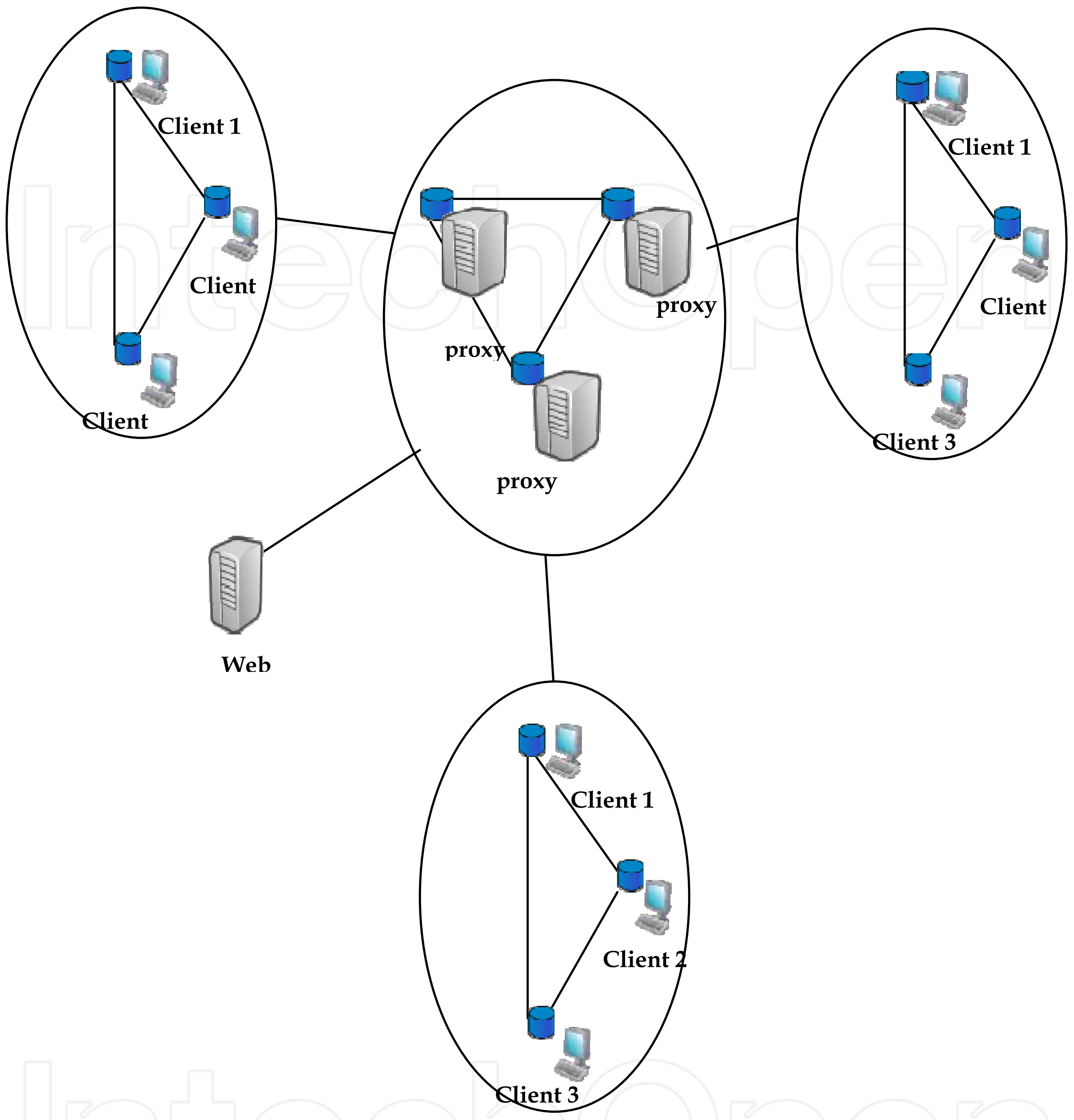


Fig. 3. The new proposed hybrid architecture

3.2 Considerations and design issues

There are many challenges in the proposed approach since we are dealing with an unknown number of clients in an unstable environment. We have chosen to deal with the following issues:

3.2.1 Cache Communication

The main challenge in cooperative cache architecture is how to quickly locate the location of the requested cached object.

Melpani et al. (Malpani et al., 1995) proposed a scheme where a group of caches function as one. When the user requests a page, the request is sent to some random cache. If the page was found in that cache, then it is returned to the user. Otherwise, the request is forwarded to all the caches in the scheme via IP multicast. If the page is cached nowhere, the request is forwarded to the home site of the page.
Harvest cache system (Chankhunthod et. al, 1996) uses a scheme where caches are arranged in a hierarchy and uses the Internet Cache Protocol (ICP) for cache routing (Wessels & Claffy, 1998). When a user requests a page, the request travels up the hierarchy to locate the cached copy without overloading the root caches by allowing the caches to consult their siblings in each level before allowing the request to travel up the hierarchy.
Adaptive Web Caching (Michel et al., 2001) builds different distribution trees for different servers to avoid overloading any root. This scheme is robust and self-configuring. It is more efficient with popular objects. For less popular objects, queries need to visit more caches, and each check requires a query to and responses from a group of machines. It is suggested to limit the number of caches the query visits, to decrease the added delay.
Provey and Harrison (Povey & Harrison, 1997) proposed a distributed caching approach to address the problems faced in the previously proposed hierarchical caching. They constructed a manually configured hierarchy that must be traversed by all requests. Their scheme is promising in the way that it reduces load on top-level caches by only keeping location pointers in the hierarchy (Wang, 1999). A simulation study was done as well, where the results showed that this proposed approach performs well for most network topologies. Results have also shown that in topologies where the number of servers in the upper levels is low, the performance of the hierarchical caching is better than the proposed approach. The conclusion of this paper is that the overall results show that there is no significant performance difference between the old and the proposed approach.
Wang (Wang, 1997) describes an initial plan in Cachemesh system to construct cache routing tables in caches. These tables guide each page or server to a secondary routing path if the local cache does not hold the document. A default route for some documents would help to keep table size reasonable (Wang, 1999).
Legedza and Guttag (Legedza & Guttag, 1998) offered to reduce the time needed to locate unpopular and uncached pages or documents by integrating the routing of queries with the network layer's datagram routing services (Wang, 1999).

3.2.2 Cache Coherency

The most outstanding benefit of web caching is that it offers the user lower access latency. It also defeats the side-effect of providing the user with stale pages (i.e. pages which are out of date with respect to their home site). The importance of keeping the cache's content coherent is to provide the user with fresh and up-to-date pages. Web caching reduces redundant data in the network which eases the process of keeping the pages updated. Some of the proposed mechanisms to keep cache coherency are strong cache consistency and weak cache consistency (Wang, 1999).

- Strong cache consistency
    - Client validation. This approach is also called polling-every-time. The proxy initially considers the cached pages are expired on each access and sends an If-Modified-Since header with each access of the resources.

    - Server invalidation. When a resource is changed at the server, it sends invalidation messages to all clients that have recently accessed and cached the resource. The server has to keep a list of clients who requested and cached the changed resources which becomes unmanageable for the server when the number of the clients is large.
- Weak cache consistency
    - Adaptive TTL. The resource freshness problem is dealt with by adjusting the time-to-live parameter based on observations of its lifetime. If a file has not been modified for a long time, it tends to stay unchanged. Thus, the time-to-live attribute to a document is the current time minus the last modified time of the document.
    - Piggyback invalidation. Whenever a cache has to communicate with the server, it adds along with it a list of resources that are potentially out-of-date and asks for validation.

3.2.3 Cache Contents

Proxy caches have been recognized as an effective and efficient solution to improve the web performance. A proxy serves in different roles: data cache, connection cache, and computation cache. A recent study has shown that caching Web pages at proxy level reduces the user access latency 3% - 5% as compared to the no-proxy scheme (Wang, 1999).
It is very important to set the architecture and prepare it to deal with different types of resources. Most of the web resources are becoming dynamic with the invasion of web services. It is very helpful to use computation caching to retrieve dynamic data. It can be done by caching dynamic data at proxies and migrating a small piece of computation to proxies to generate or maintain the cached data. Also the architecture should be able to retrieve information about the requested resource before adding delay to the request by looking for it in the caches when it is an uncachable resource.

3.2.4 Load balancing

The hot spot problem is one of the issues that triggered the web caching research area. It occurs any time a large number of clients access data or get some services from a single server. If the server is not set to deal with such situations, clients will perceive a lot of delay and errors and the quality of service will be degraded. Several approaches to overcome the hot spots have been proposed. Most use some kind of replication strategy to store copies of hot pages/services throughout the Internet; this spreads the work of serving a hot page/service across several servers (Wang, 1999). Another approach that can be used is to get the server to work in a cooperative set with other servers or caches to fulfil a request without overwhelming the home server with users' requests.

3.3 Flow of information in the architecture

The flow of information in the architecture can have different scenarios and paths. The two scenarios chosen for this architecture are as follows.
Scenario1, each client keeps a search history log of the clients that contacted it. When a client initiates a request it first looks into its local cache. If the requested page is found, then it is fetched from the local cache of the client. Otherwise, it looks into its search history log

and search for the last client who requested this page. If found, it fetches the requested page from the client otherwise it consults the proxy to fetch the requested page. If the proxy finds it in its cache, it forwards the requested page to the client. Otherwise, it consults the proxies in its cooperative set. If none has it, then the request is forwarded to the home server (see Figure 4).

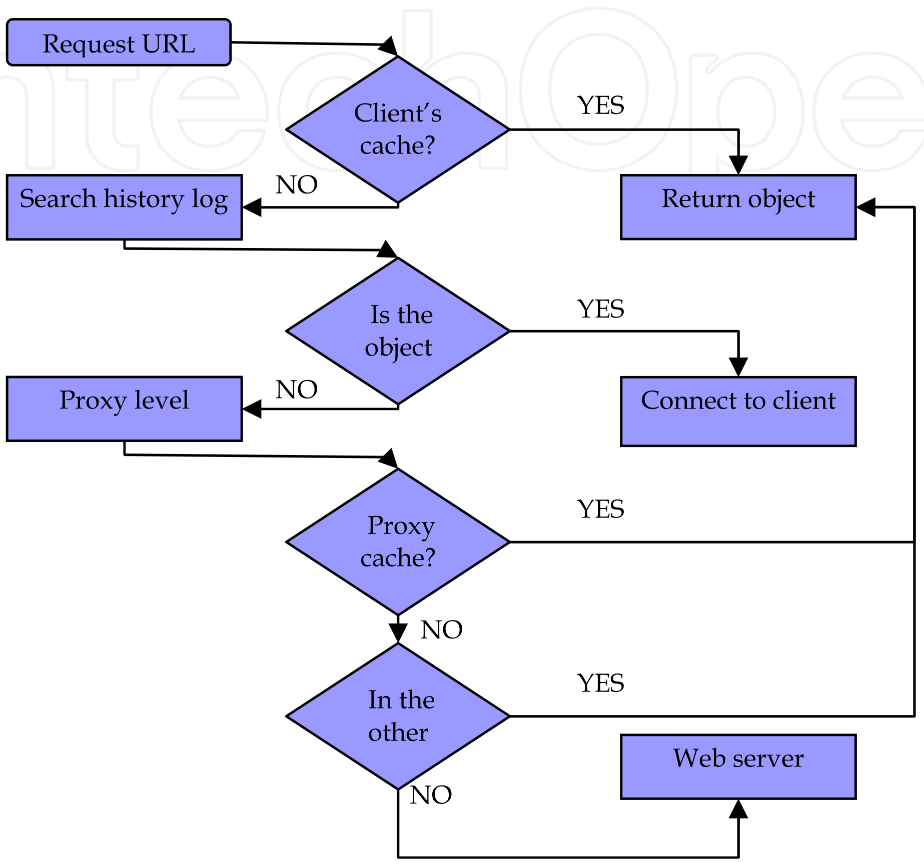


Fig. 4. Scenario 1

Scenario2, each proxy in the cooperative set is responsible of a group of clients that are geographically grouped. Also each proxy acts as the leader of the peer-to-peer connected clients and contains cache and routing information of the client caches. When a client initiates a request, it will first check in its local cache. If it is found then it will be fetched from the local cache of the client. Otherwise, it will send the proxy a page location request. If the page was cached in one of the client's caches, then it would forward the information of the client that has the page in its cache to the requesting client. Otherwise, the proxy will consult the proxies in its cooperative set and check if any of the proxies have the requested page in its cache. If none has the requested page, then it is fetched from the home server (see Figure 5).

Both of the mentioned scenarios are to be tested and analysed using a simulator. The simulation could result in the superiority of one of them or the need for a hybrid of both.
The reason such scenarios are chosen is to explore and benefit from the free space offered by the client's caches when they are connected to the internet. This architecture aims to reduce

the load on the upper tier, the proxy, by initiating direct communication between the clients in P2P-like atmosphere which are geographically close to each other. The communication between the clients better stay as simple as possible as not to produce more delay and load on the client and organize the flow of the network traffic.

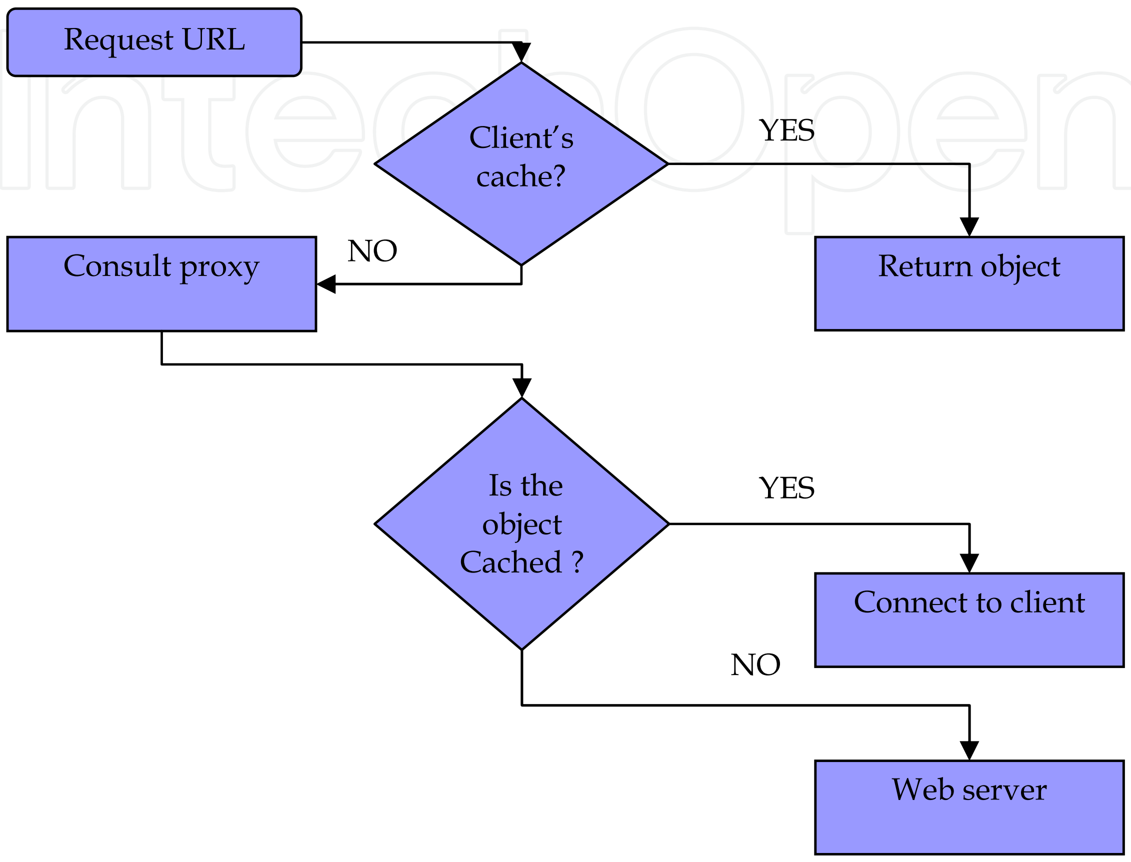


Fig. 5. Scenario 2

4. Conclusion

This chapter presented the basic web caching architectures that has been found in the literature as a solution for network congestion and server overloading problems. A rough comparison of common architectures has been presented to show the pros and cons of each. The chapter also proposed an architecture that is believed to offer a better performance in different aspects which is due to combining the benefits of many architectures and schemes into this new architecture. This architecture will look into some design issues such as the communication between the caches, the path to keep the caches' contents coherent, cache contents, and load balancing at the client and proxy side. Current work is on the simulation of the architecture's flow of information scenarios, using OMNET++, to obtain results and fine tune the architecture.

5. References

Abrahams, M.; Standridge, C.R.; Abdulla G.; Williams S. & E.A. Fox(1995), "Caching Proxies: Limitations and potentials", *in Proceedings of the 4th International World-Wide Web Conference*, pp. 119-133, July 1995.

Chankhunthod, A.; Danzig, P. B.; Neerdaels, C.; Schwartz, M. F. & Worrel, K. J.(1996), "A hierarchical Internet object cache", *in Proceedings of the 1996 Usenix Technical Conference*, pp. 13, 1996.

Dykes, S.G. & Robbins, K.A.(2002), "Limitations and benefits of cooperative proxy caching", *IEEE Journal on Selected Areas in Communications*, Vol. 20, issue 7, September 2002, pp. 1290-1304.

Eaton, P.; Ong, E. & Kubiatowicz, J. (2004) , "Improving Bandwidth efficiency of peer-to-peer storage", in Proceedings Fourth International Conference on Peer-to-Peer Computing, pp. 80-90, ISBN 0-7695-2156-8 , August 2004.

Fan Li; Cao, Pei; Lin, Wei & Jacobson, Q.(1999), "Web prefetching between low-bandwidth clients and proxies: potential and performance", *Performance Evaluation Review*, Vol. 27, issue 1, June 1999, pp. 178-187.

Gadde, S.; Rabinovich, M. & Chase, J. S.(1997), "Reduce, reuse, recycle: An approach to building large internet caches", *in Proceedings of the Workshop on Hot Topics in Operating Systems*, pp. 93–98, ISBN 0-8186-7834-8, May 1997.

Korupolu, M.R. & Dahlin, M. (1999), "Coordinated placement and replacement for large-scale distributed caches", *in Proceedings of the 1999 IEEE Workshop on Internet Applications*, pp. 62–71, August 1999.

Lee, K.W.; Sahu S.; Amiri K. & Venkatramani C.(2001), "Understanding the potential benefits of cooperation among proxies: Taxonomy and analysis", *Technical report, IBM Research Report*, Septmber 2001.

Legedza, U. & Guttag, J.(1998), "Using network-level support to improve cache routing", Computer Networks and ISDN Systems, Vol. 30, Issue 22 - 23, November 1998, pp. 2193-2201, ISSN 0169-7552.

Liu, Jiangchuan; Chu, Xiaowen & Xu, Ke(2005), "On peer-to-peer client web cache sharing", *IEEE international conference on communications*, Vol. 1, May 2005, pp. 306 – 310.

Lyer, S.; Rowstron, A. & Druschel, P.(2002), "Squirrel: a decentralized peer-to-peer web cache", *in Proceedings of the twenty-first annual symposium on Principles of distributed computing*, pp. 213 – 222, ISBN 1-58113-485-1, 2002.

Malpani, R.; Lorch, J. & Berger, D.(1995), "Making World Wide Web caching servers cooperate", *in Proceedings of the 4th International WWW Conference*, Boston, MA, pp. 107 - 117, December 1995.

Michel, S.; Nguyen, K.; Rosenstein, A.; Zhang, L..; Floyd, S. & Jacobson, V.(2001), "Adaptive web caching: towards a new global caching architecture", *IBM Research Report*, RC22173, September 2001.

Povey, D. & Harrison, J.(1997), "A distributed Internet cache", *in Proceedings of the 20th Australian Computer Science Conference*, Sydney, Australia, February 1997.

Rowstron, A. & Druschel, P.(2001), "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems", *in Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001*), pp. 329 – 350, November 2001.

Wang, Z.(1997), "Cache mesh: a distributed cache system for World Wide Web", *in Proceedings of the WCW'97*, Boulder, CO, June 1997.

Wang, J.(1999), "A Survey of Web Caching Schemes for the Internet", *Computer Communication Review*, Vol. 29, issue 5, October 1999, pp. 36-46.

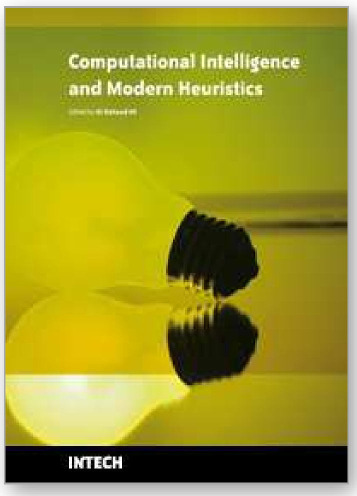
Wang, J.Z. & Bhulawala, V.(2005), "Design and implementation of a P2P cooperative proxy cache system", *in Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, pp. 508-514, September 2005.

Wessels, D. & Claffy, K.(1998), "Internet cache protocol (IPC)", version2, RFC2186, 1998.

Wolman, A.; Voelker, G.; Sharma, N.; Cardwell, N.; Karlin, A. & Levy, H.(1999), "On the scale and performance of cooperative Web proxy caching", *in Proceedings of the 17th ACM Symposium on Operating Systems Principles (SOSP'99)*, pp. 16–31, Kiawah Island Resort, SC, USA, December, 1999.

Zenel, B. & Duchamp, D.(1995), "Intelligent communication filtering for limited bandwidth environments", *in Proceedings Fifth Workshop on Hot Topics in Operating Systems*, pp. 28-34, May 1995.

Zhu, Y. & Hu, Y.(2003), "Exploiting client caches: an approach to building large Web caches", *in Proceedings 2003 International Conference on Parallel Processing*, pp. 419-426, ISBN 0-7695-2017-0, October 2003.



ISBN 978-953-7619-28-2
Hard cover, 348 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010