Performance comparison of distributed architectures for content adaptation and delivery of Web resources

Claudia Canali
University of Parma
claudia@weblab.ing.unimo.it

Valeria Cardellini
University of Roma "Tor Vergata"
cardellini@ing.uniroma2.it

Michele Colajanni
University of Modena
colajanni@unimo.it

Riccardo Lancellotti
University of Modena
lancellotti.riccardo@unimo.it

Abstract

The increasing popularity of heterogeneous Web-enabled devices and wired/wireless connections motivates the diffusion of content adaptation services that enrich the traditional Web. Different solutions have been proposed for the deployment of efficient adaptation and delivery services: in this paper we focus on intermediate infrastructures that consist of multiple server nodes. We investigate when it is really convenient to place this distributed infrastructure closer to the clients or to the origin servers, and which is the real gain that can be get by node cooperation. We evaluate the system performance through three prototypes that are placed in a WAN-emulated environment and are subject to two types of workload.

1. Introduction

The recent proliferation of heterogeneous Web-enabled devices and network connections, and the contemporary growing complexity of Internet-based services have scaled the demand for content adaptation services to an all-time high. The differences among the present devices (desktops, mobile phones, hand held computers, PDAs, and Web-TVs) concern various resources, such as CPU power, storage capacity, ability to accept and manage data types, and network connectivity. Such an emerging scenario determines the need for solutions which provide efficient adaptation and delivery of heterogeneous Web-based resources. These services are typically obtained through complex operations carried out by intermediary agents that are interposed between the traditional Web client and server processes. These agents may run on the client machines (*client-side* solutions) or on intermediary server nodes, that may be placed closer to the clients (*edge server-side* solutions) or to the origin servers (*origin server-side* solutions).

The number of feasible alternatives is huge. In this paper, we do not consider client-side solutions that may be affected by device technological constraints, and prefer to focus on intermediary distributed architectures. The first proposals of edge server-side solutions are represented by multiple stand-alone nodes located close to the client devices [8, 11, 14]. An important evolution is represented by architectures that provide some form of cooperation among the nodes of the intermediate architecture [1, 7]. These cooperative distributed systems, that in this paper are called *cooperative edge server-side* architectures, are motivated by two factors: the computational cost of adaptation services leads to load sharing; the significant increment of circulating Web resources (original and adapted versions) increases disk space needs and leads to cooperative caching.

The contribution of this paper is to investigate the pros and cons of placing the distributed infrastructure, that executes adaptation and delivery services, closer to the clients or closer to the origin servers. We provide the first performance comparison of the three architectures for content adaptation and delivery services that have emerged as leading solutions. Specifically, we consider representative implementations of the origin server-side, the edge server-side, and the cooperative edge server-side architectures to evaluate the sensitivity of the system performance to two main factors: the WAN effects and the workload models. The performance evaluation is carried out through experimental prototypes where the number of nodes of the three distributed architectures is kept constant, and a testbed where wide-area network characteristics are reproduced through WAN emulators. This controlled environment, where the experiments are scientifically reproducible, allows us to carry out a detailed evaluation of the impact of network characteristics such as latency and bandwidth.

Many adaptation services can be carried out in a heterogeneous client environment. In this paper, we focus on image transcoding services because these visual resources still predominate the content of most Web sites. The adaptation of images is carried out by operating on various parameters, such as spatial geometry, color depth, quality factor, and MIME subtype. In all the considered architectures, the adaptation services are provided on-the-fly. While this is the only viable solution for the edge server-side approach, we have chosen it for the origin server-side architecture because it guarantees more flexibility than the off-line solution and it frees the provider from creating in advance and keeping consistent multiple versions of its resources.

In literature there are few experimental performance comparisons of different infrastructures for efficient adaptation and delivery of Web resources, and most of them have been carried out through simulations. After many experiences of experiments of distributed systems in the Internet (e.g., [1, 2]), we are appreciating the benefits of having a controlled testbed environment. Indeed, the performance evaluation of an origin server-side system for content adaptation provided by Chandra *et al.* [3] in an uncontrollable real environment does not allow the authors to give a clear picture of the impact of WAN effects. Similar limits characterize performance evaluations of edge server-side architectures [14]. More detailed analyses on WAN effects have been carried out by Dykes *et al.* [6], that provide a detailed analytical model for evaluating the network sensitivity of a traditional service such as Web caching.

The investigation of the edge server-side architectures has attracted the attention of many researchers. For example, Han *et al.* [8] provide an analytical study of edge server-side transcoding, but their model uses simplified assumptions on network utilization and does not consider the impact of network congestion. Other studies (e.g., [10, 11]) focus on the system architecture and limit their experiments to a LAN scenario. Other performance evaluations of caching in systems for content adaptation have been carried out through simulators [5, 12]. Clearly, the network models in these simulators are simplified and cannot take into account most packet-level dynamics and impact of WAN traffic.

Cooperative edge-side architectures for content adaptation have been proposed more recently (e.g., [1, 13]). The study in [13] focuses on content personalization of peer-to-peer networks and discusses architectural scalability issues, without taking into account network effects. Other studies [1] on cooperative architectures are based on real uncontrollable networks.

The rest of this paper is organized as follows. Section 2 describes the three intermediary architectures for efficient content adaptation and delivery. Section 3 presents the experimental testbed. Section 4 discusses the experimental results on the performance comparison of the considered architectures. Section 5 concludes the paper.

2. System architectures

The basic architecture of a scalable system for efficient content adaptation and delivery services consists of a multi-level infrastructure in which we identify three main levels: the *client*, the *origin server* and an *intermediary* level in between the two. The origin server is the content repository of the original Web resources and does not perform any content adaptation operation. The intermediary nodes provide content *adaptation* to transform, when necessary, the required resources on the basis of the client context information (i.e., the device characteristics and capabilities). For this reason, in the rest of this paper we will refer to the intermediary nodes as *adaptation servers*, which can also cache the adapted version of a resource to exploit data reuse.

The client requests for Web resources are processed by the adaptation servers that can interact with the origin server to retrieve the original resource before the final delivery to the client. Due to the presence of multiple versions of the same resource, a multi-version lookup process is necessary and may cause one of the following three events. An *exact hit* occurs if the cache contains the exact version of the requested resource, which is immediately sent to the client. If the cache contains a more detailed and adaptable version of the requested resource, a *useful hit* occurs and the cached object can be transformed before the delivery to the client. In case of *miss*, the cache does not contain any exact or adaptable version of the requested resource and the node must fetch the original resource from the origin server, if necessary adapts it, and then sends the result to the client.

In the following of this section we describe the three architectures compared in the paper. We can distinguish origin server-side adaptation from edge-server side adaptation, whose main difference for the scope of this paper lies in the location of the adaptation servers with respect to the origin servers and the network edge. We further distinguish two types of edge server-side architectures, depending on whether edge servers can cooperate or not among them.

2.1. Origin server-side architecture

In this architecture the adaptation services are carried out directly on the platform of the content provider for its own resources. Therefore, the adaptation servers are located on the same local network of the origin server. As already noted, we consider an origin-server side approach providing on-the-fly adaptation services because of the benefits of this solution. However, the drawback of on-the-fly adaptation is intuitive: it may require significant computing power for adaptation services in addition to generating and delivering traditional Web-based services. For this reason, the provider platform needs to replicate computing resources. In addition, caching of adapted resources allows to retrieve the requested content from the disk rather than to transform it again. Therefore, we consider a two-tier architecture as shown in Fig. 1, where the front-end tier consists of multiple adaptation servers acting as enhanced reverse proxies, while the back-end tier is a simple content repository of the original resources.

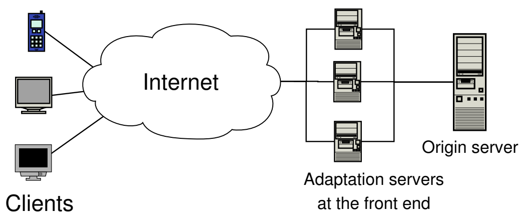

Figure 1. Origin server-side architecture.

2.2. Edge server-side architecture

Fig. 2 illustrates the edge server-based architecture: the wide area network is between the origin server and the adaptation servers, that are located on the network edge close to the clients (the adaptation servers are typically placed within the network of the ISP providing Internet access to the clients). Adaptation services carried out by some edge nodes seem the most viable solution for the following reasons. First, the large majority of new devices needs a gateway to access Web-based services. Second, adaptation services can take advantage of caching of already adapted resources saving not only computational power but also network bandwidth. Finally, the adapted content can be served from a node closer to the client than the origin server.

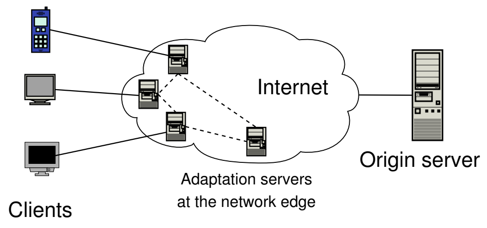

Figure 2. Edge server-side architecture.

An edge server-side architecture can be operated by a third party company on behalf of its customer content providers or by an independent third party. In the first case, the adaptation and delivery services are limited to a specific subset of Web resources and it is possible to exploit a server-directed adaptation process taking into account the semantics of the resource content. On the other hand, an architecture operated by an independent third party, that provides the adaptation and delivery services for all Web resources and has only a reduced interaction with the content provider, cannot provide all possible forms of adaptation. However, we only consider device-driven adaptation services, where the transformation operations are mainly related to the characteristics of the client device and do not require a strict interaction with the content provider.

2.3. Cooperative edge server-side architecture

The cooperative edge server-side architecture is similar to the edge server-side, with the difference that now the adaptation servers can cooperate. The cooperation can regard the distribution of the content adaptation operations from heavily to lightly loaded nodes or the content location to take advantage of cache contents of nearby nodes. As the cooperative content location has a predominant performance impact [1], in this paper we consider only this form of cooperation. Fig. 2 shows the cooperation through dashed lines connecting the adaptation servers.

The cooperative content location in the context of adaptation services may follow multiple schemes, such as query- or summary-based lookup algorithms. Since in a previous study [1] the authors have found that a query-based cooperation achieves the best performance thanks to the effectiveness of its resource location algorithm, in this paper we consider a cooperative architecture that uses a query-based mechanism.

When a node receives a client request, it starts a local lookup for the requested resource in the server cache and may require a cooperative lookup process. The query-based scheme performs the cooperative lookup by sending a query message to each peer. A response from a peer causes a remote hit that must be explicitly fetched, thus preserving the complete transparency to the client. When no suitable resource is found in any peer, the requested resource must be fetched from the origin server.

3. Experimental testbed

In this section, we describe the workload models and the setup of the experimental environment used to compare the performance of three content adaptation and delivery architectures considered in this paper.

3.1. Workload models

We carried out our experiments using two workload models which differ in the nature of the working set. The first model, namely *IRcache*, aims at capturing a realistic Web scenario with a reduced adaptation load. The set of resources is based on proxy traces belonging to the IRCache infrastructure. Some characterizations performed on the images of this workload, such as file size, JPEG quality factor, and colors of GIF images, evidence that they are very close to the characteristics reported in [4].

The second workload model, namely *Photo album*, has a significant amount of large pictures in order to denote a scenario where the adaptation process has a major cost. As the trend of the Web is towards a growing demand for graphical and multimedia resources, this workload can represent a likely Web scenario for the future. For this reason, we provide a more detailed discussion of results obtained with this latter workload, using the IRcache workload only for the initial performance comparison. For both workloads we introduced a popularity resource distribution by defining a set of hot resources (corresponding to 1% of the working set): 10% of the total number of requests refers to this set.

Client requests are issued to the system using a client emulator according to synthetic traces (in a way similar to the experiments described in [1]). The client request rate is the same for both workloads.

3.2. System setup

We set up an experimental test-bench composed of a node acting as a client emulator, 16 nodes equipped with the content adaptation software, and a node with a Web server that acts as the origin server. Fig. 3 shows the experimental testbed, with the three types of nodes that are used throughout the experiments: *client*, *adaptation servers* and *origin server*. The content adaptation function is provided by a prototype based on Squid [1]. Each adaptation server node manages also a caching space, that is configured to hold 25% of the global working set for both workloads.

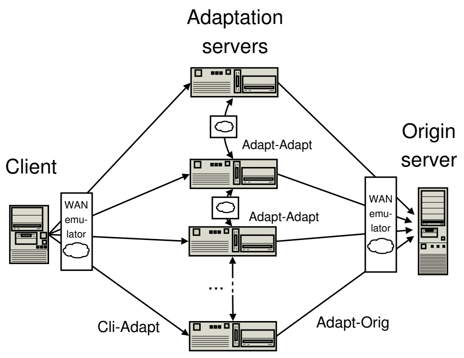

Figure 3. Experimental testbed.

To compare the three architectures we introduce WAN emulation on the links connecting the nodes. As shown in Fig. 3, we have three classes of links, namely *Cli-Adapt* from client to adaptation servers, *Adapt-Adapt* among adaptation servers, and *Adapt-Orig* from adaptation servers to origin server. WAN emulation is provided through special packet schedulers that are part of the 2.6 Linux kernel. We simulate three WAN effects: packet loss, packet delay, and bandwidth limitation. Packet loss and delay are provided by the *netem* packet scheduler, while bandwidth limitation is obtained through the *token bucket filter* traffic shaper. Delay is modeled through a linear combination of Pareto and Gaussian distribution, as suggested in [9].

As the focus of our study is directed towards an architecture comparison at the provider level, we do not take into account the last-mile and we focus on the system part from the client access point to the origin server. Since in the origin server-side architecture the adaptation servers are located on the same network of the origin server, we introduce WAN emulation on Cli-Adapt links, while in the edge server-side architecture we introduce WAN effects only on the Adapt-Orig links of Fig. 3. Finally, in the cooperative edge server-side architecture we introduce WAN emulation both on Adapt-Adapt links and on Adapt-Orig links.

The space of choice of the network emulation parameters is huge, as each link is described by three different parameters (network delay, packet loss probability, and bandwidth). A preliminary study on real networks allowed us to identify a meaningful subset of the possible combinations of values for the network emulation parameters, shown in Table 1, that we used in our experiments.

4. Experimental results

In this section we compare the performance of the three considered architectures, using as main index the system response time. Due to space limitations, we report only a significant subset of the results of our analysis that takes into account other indexes, such as cache hit rate and utilization of hardware and operating system resources of the nodes.

We first compare the architectures under the same network scenario (we use the bold values reported in Table 1 for bandwidth) for the two different workloads, in order to evaluate their impact on caching performance, computational load, and network resource utilization. In the second set of experiments we use only one workload and we evaluate the performance sensitivity to the network scenario.

4.1. Sensitivity to workload

Figs. 4 and 5 show the cumulative distribution of the response time for the origin server- and edge server-side architectures for the IRcache and the Photo album workloads, respectively. A comparison of the two figures shows that the Photo album workload provides higher response times with respect to the IRcache workload. The reason for this performance difference is twofold. First, the Photo

Table 1. Reference values for the parameters of the WAN emulator.

Architecture	WAN emulated links	Bandwidth [Mbit/s]	Delay [ms]	Loss rate
Origin server-side	Cli-Adapt	8-**16**-32	100	1%
Edge server-side	Adapt-Orig	8-**16**-32	100	1%
Cooperative edge server-side	Adapt-Orig	8-**16**-32	100	1%
	Adapt-Adapt	8-**16**-32	25	1%

album workload places higher content adaptation requirements than the IRcache workload, leading to an increase in the 90-percentile of adaptation processing time of eight times. Second, resources in the Photo album workload are on average three times larger than in the IRcache workload and this augments the transmission time.

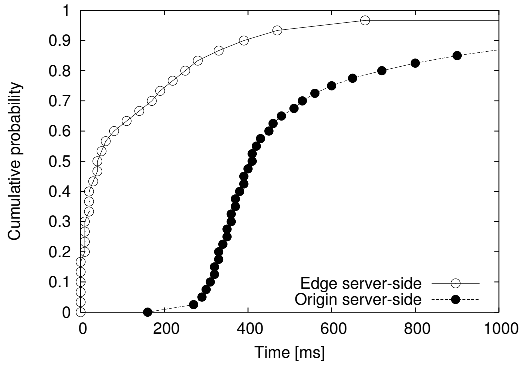

Figure 4. Response time comparison of origin server-side and edge server-side architectures (*IRcache* workload).

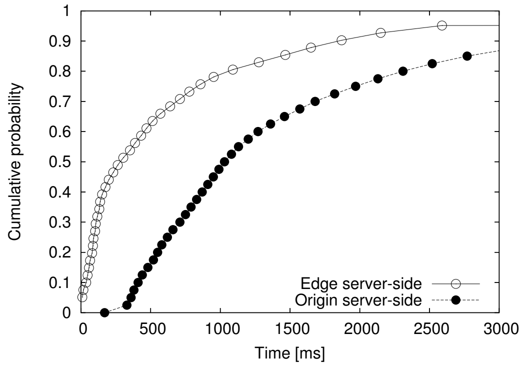

Figure 5. Response time comparison of origin server-side and edge server-side architecture (*Photo album* workload).

The curves shown in Figs. 4 and 5 demonstrate that the edge server-side architecture provides better performance than the origin server-side architecture for both considered workloads. This performance gain is due to the placement of the adaptation servers close to the clients, that allows the edge server-side architecture to serve client requests resulting in cache hits using only local resources, while WAN links are used only in the case of a cache miss. In particular, if we look at the bottom of graphs in Figs. 4 and 5 (i.e., for cumulative probability below 0.3), we see that the requests service time for the origin server-side architecture is one order of magnitude higher than that for the edge server-side architecture. This can be explained by the cache hit rate of the edge server-side architecture, that can serve locally more than 30% of requests.

The comparison between the edge server-side and the origin server-side architecture can be summarized as follows. For the IRcache workload, the median response time is reduced by more than ten times and the 90-percentile is reduced by a factor of four. The Photo album workload confirms these results, showing a reduction of four times of the median response time and a 90-percentile that has been nearly halved by the edge server-side architecture.

Table 2. Response time comparison of edge server-side and cooperative edge server-side architectures.

IRcache workload		
Architecture	**Response time [ms]**	
	median	**90-percentile**
Edge server-side	49	385
Cooperative edge server-side	36	281
Photo album workload		
Architecture	**Response time [ms]**	
	median	**90-percentile**
Edge server-side	180	1848
Cooperative edge server-side	126	1798

We now focus on the performance improvement achievable by means of lookup cooperation among the adaptation servers. Table 2 shows a performance comparison of median and 90-percentile of response time of the edge server-side and cooperative edge server-side architectures for the two workloads. If we compare the median response time we can appreciate the performance gain due to cooperation, that is 33% for the IRcache workload and rises up to 42% for the Photo album workload. This gain is mainly related to the requests resulting in a remote hit on a nearby cache (nearly 20%), that allow to save a download from the origin server and, possibly, a content adaptation operation. As in our WAN emulated scenario Adapt-Adapt links are characterized by lower delays with respect to the Adapt-Orig links, remote hits are usually served faster than misses. On the other hand, both the architectures handle misses in the same way, thus providing similar performance if we look at the 90-percentile of response time.

4.2. Sensitivity to network parameters

We now evaluate the impact of network parameters on the performance of the considered architectures. We carried out the sensitivity analysis for both workloads; however, due to space reasons, we only report the results for the Photo album workload that, due to its larger resource size, places a heavier load on the network. Results for the IRcache workload confirm the main findings, but the observed sensitivity to network parameters is less significant. We studied the effect of both network latency and bandwidth limitation in the WAN emulation, but we report only the results related to the sensitivity analysis to the bandwidth, that has an impact on response time of one order of magnitude higher than the sensitivity to the network latency.

Table 3. Open sockets and response time as a function of the WAN bandwidth.

Origin server-side architecture			
Bandwidth [Mbit/s]	**Average # open socket**	**Response time [ms] median**	**Response time [ms] 90-percentile**
8	190	8430	120100
16	110	1030	3440
32	40	880	2620
Edge server-side architecture			
Bandwidth [Mbit/s]	**Average # open socket**	**Response time [ms] median**	**Response time [ms] 90-percentile**
8	80	470	54680
16	16	180	1848
32	11	170	1630

Table 3 provides a performance comparison of the origin server-side and edge server-side architectures for values of the bandwidth of the WAN links ranging from 8 to 32 Mbit/s. For both architectures we see clearly the detrimental effect of bandwidth reduction on the performance. However, the most interesting observations arise from the comparison as the bandwidth is reduced. The edge server-side architecture always outperforms the origin server-side architecture and the performance gain augmentes as the bandwidth decreases. For example, the performance gain of the edge server-side architecture over the origin server-side approach on the 90-percentile of response time grows from 60% to 86% as the bandwidth is reduced from 32 to 16 Mbit/s. Moreover, when the bandwidth is set to 8 Mbit/s the origin server-side architecture is characterized by a high number of failures (120 sec., which is the 90-percentile of response time, is the timeout of our client emulator). These results clearly demonstrate that the edge server-side architecture provides a more graceful performance degradation even in the case of network congestion.

The median response time shown in Fig. 6 as a function of WAN links bandwidth confirms these findings. We observe a growth in performance gain of the edge-side architecture from 417% to 472% as the bandwidth is reduced from 32 to 16 Mbit/s, and the difference in behavior between the two architectures is even more significant when the bandwidth is further reduced to 8 Mbit/s.

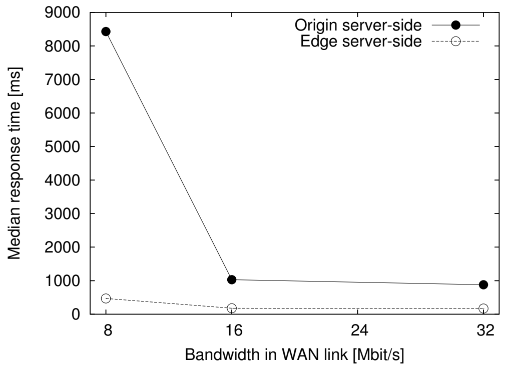

Figure 6. Median response time as a function of the WAN bandwidth.

The performance difference between the two architectures has a strong relationship with the number of sockets open during the experiments. Table 3 shows, in the second column, the average number of sockets open on the adaptation servers throughout our experiments as a function of the bandwidth. The number of open sockets grows as the bandwidth is reduced due to the increase in concurrency in client requests caused by longer service times. The number of open sockets is always less than one third for the edge server-side architecture with respect to the origin server-side approach. This is consistent with the observation that the edge server-side architecture relies on WAN links much less than the origin server-side architecture.

As final sensitivity analysis to network parameters, we now evaluate the performance gain obtained by the cooperative edge server-side architecture over the edge server-side architecture. Table 4 shows the median and 90-percentile of response time as a function of the Adapt-Adapt and Adapt-Orig bandwidths. These results indicate that the introduction of cooperation reduces the sensitivity to the network parameters, due to a more fair usage of network resources. In case of local miss, indeed, downloads can occur from both neighbor caches or from the origin server. Furthermore, when multiple copies of a Web resource are located in different cooperative nodes, the system automatically chooses the fastest responding neighbor, thus contributing to evenly distribute network load across the links.

Table 4. Sensitivity to network parameters for the cooperative edge server-side architecture.

Sensitivity to Adapt-Adapt bandwidth		
Bandwidth [Mbit/s]	**Response time [ms]**	
	median	**90-percentile**
8	150	1960
16	130	1870
32	110	1790
Sensitivity to Adapt-Orig bandwidth		
Bandwidth [Mbit/s]	**Response time [ms]**	
	median	**90-percentile**
8	170	2030
16	130	1870
32	110	1660

The important result is that the response time is much more stable than in other architectures even in front of significant bandwidth variations. The maximum performance difference on 90-percentile of response time is only 22% as the bandwidth of both Adapt-Orig and Adapt-Adapt links ranges from 8 to 32 Mbit/s. This is fairly low if we consider that the 90-percentile of response time ranges over at least one order of magnitude for the other architectures. This result is consistent with previous studies of Dykes *et al.* [6] who, focusing on traditional Web caching, have demonstrated that cooperation provides a major benefit in reducing response time variance due to network-related delays.

5. Conclusions

In this paper we compared the performance of three architectures for Web content adaptation and delivery services in an reproducible and configurable WAN experimental environment. We found that the placement of content adaptation and caching close to the client devices provides an important performance benefit for different workloads and network scenarios. The origin server-side solution is more sensitive to network conditions due to the heavy usage of WAN links, while the edge server-side architecture reduces the sensitivity to WAN effects thanks to client requests resulting in local hits that do not rely on WAN resources.

We demonstrated that cooperation among the nodes of an edge server-side architecture improves performance further. In particular, cooperation in the edge server-side architecture is extremely useful in reducing sensitivity to network parameters down to almost nothing due to a more efficient utilization of WAN links. Indeed, in case of local miss, cooperation allows a distribution of network load among multiple edge servers, without needing to contact the origin server for every request that cannot be satisfied locally. As a consequence, the cooperative edge server-side architecture provides the best performance gain in case of scarce network resources, whereas the other architectures pay high penalties in response time due to network delays.

Acknowledgements

The authors acknowledge the support of MIUR in the framework of the FIRB project “Wide-scalE, Broadband, MIddleware for Network Distributed Services” (WEB-MINDS).

References

[1] C. Canali, V. Cardellini, M. Colajanni, R. Lancellotti, and P. S. Yu. Cooperative architectures and algorithms for discovery and transcoding of multi-version content. In *Proc. of WCW 2003*, Sept. 2003.

[2] V. Cardellini, M. Colajanni, R. Lancellotti, and P. S. Yu. A distributed architecture of edge proxy servers for cooperative transcoding. In *Proc. of IEEE WIAPP 2003*, June 2003.

[3] C. S. Chandra, S. Ellis and A. Vahdat. Application-level differentiated multimedia Web services using quality aware transcoding. *IEEE J. Selected Areas in Comm.*, 18(12):2544–2465, Dec. 2000.

[4] S. Chandra, A. Gehani, C. S. Ellis, and A. Vahdat. Transcoding characteristics of Web images. In *Proc. of Multimedia Computing and Networking Conf.*, Jan. 2001.

[5] C.-Y. Chang and M.-S. Chen. On exploring aggregate effect for efficient cache replacement in transcoding proxies. *IEEE Trans. Parallel and Distributed Systems*, 14(6):611–624, June 2003.

[6] S. Dykes and K. Robbins. A viability analysis of cooperative proxy caching. In *Proc. of IEEE Infocom 2001*, Apr. 2001.

[7] A. Fox, S. D. Gribble, Y. Chawathe, E. A. Brewer, and P. Gauthier. Cluster-based scalable network services. In *Proc. of 16th ACM SOSP*, pages 78–91, Oct. 1997.

[8] R. Han, P. Bhagwat, R. LaMaire, T. Mummert, V. Perret, and J. Rubas. Dynamic adaptation in an image transcoding proxy for mobile Web browsing. *IEEE Personal Comm.*, 5(6):8–17, Dec. 1998.

[9] S. Hemminger. Netem home page. `http://developer.osdl.org/shemminger/netem/`.

[10] S. Ihde, P. P. Maglio, J. Meyer, and R. Barrett. Intermediary-based transcoding framework. *IBM System Journal*, 40(1):179–192, 2001.

[11] A. Maheshwari, A. Sharma, K. Ramamritham, and P. Shenoy. TransSquid: Transcoding and caching proxy for heterogeneous e-commerce environments. In *Proc. of IEEE RIDE 2002*, pages 50–59, Feb. 2002.

[12] B. Shen, S.-J. Lee, and S. Basu. Caching strategies in trascoding-enabled proxy systems for streaming media distribution networks. *IEEE Trans. Multimedia*, 6(2):375–386, Apr. 2004.

[13] W. Shi, K. Shah, Y. Mao, and V. Chaudhary. Tuxedo: A peer-to-peer caching system. In *Proc. of PDPTA 2003*, June 2003.

[14] A. Singh, A. Trivedi, K. Ramamritham, and P. Shenoy. PTC: Proxies that transcode and cache in heterogeneous Web client environments. *World Wide Web*, 7(1):7–28, Jan. 2004.