Distribution of adaptation services for ubiquitous Web access driven by user profiles

Claudia Canali
University of Modena and Reggio Emilia
canali.claudia@unimo.it

Michele Colajanni
University of Modena and Reggio Emilia
colajanni.michele@unimo.it

Riccardo Lancellotti
University of Modena and Reggio Emilia
lancellotti.riccardo@unimo.it

Abstract

The popularity of ubiquitous Web access requires run-time adaptations of the Web contents. A significant trend in these content adaptation services is the growing amount of personalization required by users. Personalized services are and will be a key feature for the success of the ubiquitous Web, but they open two critical issues: performance and profile management. Issues related to the performance of adaptation services are typically addressed by highly distributed architectures with a large number of nodes located closer to user. On the other hand, the management of user profile must take into account the nature of these data that may contain sensitive information, such as geographic position, navigation history and personal preferences that should be kept private.

In this paper, we investigate the impact that a correct profile management has on distributed infrastructures that provide content adaptation services for ubiquitous Web access. In particular, we propose and compare two scalable solutions of adaptation services deployed on the nodes of a two-level topology. We study, through real prototypes, the performance and the constraints that characterize the proposed architectures.

1 Introduction

Ubiquitous access and content personalization are driving the Web towards a new dimension of interesting features and related issues that must be addressed by the underlying infrastructure. The conjunction of *ubiquitous Web access* and *Web personalization* allows the content providers to offer adaptation services that tailor Web resources to the user preferences and to the capabilities of their client devices. On the other hand, providing Web access for a plethora of heterogeneous and mobile client devices requires on-the-fly content adaptation services, because a pre-generation of formats for any combination of devices and network protocols is simply unfeasible. Moreover, providing personalized value-added services implies an accurate management of profiles that may contain sensitive information about user preferences. Preserving the confidentiality of the user profile is and will be a key issue for the success of the ubiquitous Web.

Performance and privacy issues have been addressed separately even by the recent literature (e.g., [2, 4] for performance and [5, 7] for privacy), while we claim that the design of efficient architectures for Web adaptation services should be related to the solutions for a correct management of the user profiles. As the adaptation services must access the user profile information, their location has a mutual influence on both efficiency and security goals. In a centralized system that provides content adaptation and delivery, there is no problem about where to store user profile information. On the other hand, the best choice about the various alternatives is unclear when we consider a distributed infrastructure for providing ubiquitous Web access and personalization services that is intermediate between the clients and the content providers. Most content adaptation services require computationally expensive tasks, hence much interest of the research community has been focused on high performance architectures that are able of guaranteeing efficient and scalable adaptation services. Different highly distributed architectures [6] have recently emerged as a valid solution to efficiently provide content adaptation services for the ubiquitous Web access. To improve the performance of the offered services by diminishing the response time experienced by the users, many intermediate architectures, such as CDNs, propose the use of server nodes located close to the clients (that is, on the *network edge*).

A correct management of the user profiles is a complex task when we consider highly distributed architectures. The need for high performance combined with the necessity of guaranteeing an high security standard for sensitive profile information often results in a trade-off between distributed and centralized approaches. This trade-off is even more critical in a system that updates user profile information in a dynamic way, but this problem is out of the scope of this paper.

The main contribution of this paper is to propose and compare two distributed architectures that aim to combine solutions for performance and privacy. The architectures provide content adaptation services for the ubiquitous Web access by following two different approaches for the user profile management. Both architectures refer to a model where an intermediary provider of adaptation services is interposed between the clients and the content providers. The proposed architectures are based on a two-level topology that distinguishes between a limited number of powerful, well-connected and trusted *internal* nodes and a large number of simpler *edge* nodes located close to the clients [2]. The first architecture, namely *internal oriented*, places every content adaptation service on the internal nodes, while the second architecture (namely *internal-edge oriented*) allows the distribution of content adaptation operations over multiple nodes. Both architectures are designed to provide an adequate level of privacy in the management of user profiles. We evaluate the performance of both architectures through real prototypes. From our analysis, we can conclude that the *internal-edge oriented* architecture is the best choice when a large fraction of the profiles does not contain information that are considered sensitive by the users. On the other hand, if the trend towards highly personalized services and the consequent amount of sensitive information will continue to grow, then the *internal oriented* architecture seems the best alternative for the future.

The remainder of this paper is organized as follows. Section 3 discusses the main issues related to the mapping of adaptation services over the nodes of a two-level topology. Section 4 and 5 describe the *internal oriented* and *internal-edge oriented* architectures, respectively. Section 6 evaluates the performance of the two proposed architectures. Section 7 discusses some related work and Section 8 contains some concluding remarks.

2 Adaptation services

The term *adaptation service* is used to define a plethora of services that range from simple data compression to semantic-aware handling of contents; moreover, basic services may be combined to provide more effective ubiquitous Web access. In this paper, we focus on *personalized* adaptation services that need some personal information for the generation of the resource to be delivered. There are several ways to transfer information about user preferences. For example, it may be explicitly communicated by the user through a fill-in form, or by the client device together with the request (e.g., by means of cookies or HTTP headers). Alternatively, this information may be inferred by the present and/or past user behavior. In other instances, the system infrastructure may get user information from additional techniques, as in the case of location-based services. For complex adaptation services, it is considered unpractical to require the client to communicate all user profile information to the system along with every request. The most common approach is to provide only a small user identifier (e.g., a UserID cookie) and to use this information to retrieve the user profile that has been previously stored in the infrastructure for adaptation and delivery.

In this paper, we consider two main categories of personalized adaptation services, depending on whether or not the services require sensitive information contained in the user profile. Examples of adaptation services that typically do not require any sensitive information include virus scanning, that can use just a flag for enabling/disabling the service, and resource transcoding, usually based on the client device capabilities more than on the user preferences. On the other hand, services such as location-based personalization, adaptation to the user navigation style or adaptation to the user interests usually require profile information that are considered confidential by the large majority of the users. We should also consider that a modern intermediary-based infrastructure for content adaptation must be able to handle complex adaptation services provided by dynamic composition of basic services. The composition of different adaptation services requires the adoption of standard interfaces to represent user profile information and adaptation services. In the implementations of our solutions, we adopt the technologies that emerged as a standard for Web services [14, 8]. For example, we rely on XML to describe the interfaces of adaptation services and the information composing the user profile.

3 Two-level topology

Throughout this paper we consider a distributed infrastructure for adaptation and delivery that is based on a two-level topology where the server nodes are classifies as *edge* and *internal* nodes (Figure 1). The edge level is characterized by a large amount of nodes (shown as boxes drawn with a thin line in the figure), that are located close to the network edge. This means that edge nodes are usually placed in the points of presence of ISPs with the goal of limiting the network delays in the response time of a client request. Using a set of front-end nodes closer to the clients is already a common choice for letting mobile clients to access to Web

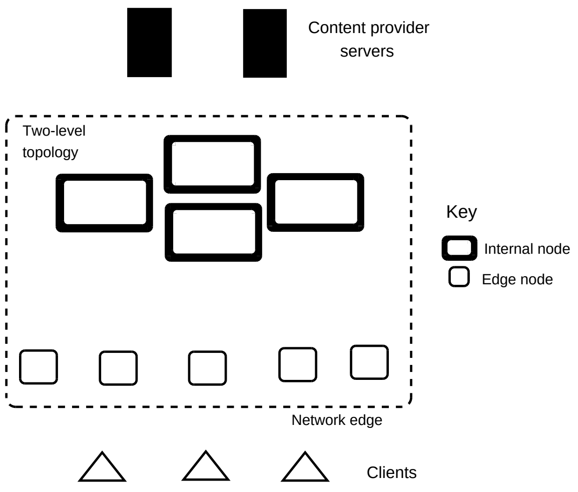

Figure 1. Two-level topology for adaptation and delivery.

based services. It seems then worthy to investigate the convenience of adding novel functionalities to the edge nodes in addition to Internet access and caching.

The internal level of the distributed infrastructure consists of a smaller number of powerful nodes (thick boxes in Figure 1). To limit communication costs especially with the edge nodes, internal server nodes are placed in *well connected* locations, which means in Autonomous Systems with a high peering degree. In previous papers, we have demonstrated that a two-level topology guarantees better performance than flat or hierarchical architectures because it allows us to preserve client access locality and provide good load sharing [2, 6]. Moreover, the reduced number of replicas of internal nodes (in the reality, one or two order of magnitude lower than the number of edge nodes) allows us to simplify many system management issues.

Here, we are interested in how to manage user profiles in a distributed architecture for ubiquitous Web access. We have two main dimensions in the problem of handling user profiles. First, we must consider the heterogeneity of the content adaptation services: some services are based on sensitive user information that impose privacy requirements, while other adaptation services do not involve sensitive information. Second, we must consider the security level of the architecture nodes: it is unfeasible to guarantee a high level of security for every node in a distributed system that may contain up to thousands of servers.

If we refer to the two-level topology for adaptation and delivery (Figure 1), we can easily conclude that the edge nodes are not suitable for storing sensitive profile information for two reasons: first, their high degree of replication may cause data inconsistency problems; second, the user profile privacy cannot be guaranteed when the related files are spread among the nodes of thousands of ISP points of presence around the world. On the other hand, the limited number and the close position of the internal nodes reduce the risk of data inconsistency, simplify server management and improve security.

From the previous considerations, we can conclude that the type of information necessary to the personalization services and the security level of the server nodes offer important limit to the feasible locations of the user profiles among the nodes of the intermediate infrastructure. Consequently, limitations about the locations of the user profiles have direct effects on possible mapping of content adaptation services over the nodes of the infrastructure, simply because personalization services requiring access to sensitive information cannot reside on remote nodes that do not guarantee high security standards.

In the following sections we propose two main alternatives for mapping content adaptation services over a two-level topology: *internal oriented* architecture and *internal-edge oriented* architecture.

4 *Internal-oriented* architecture

The *internal oriented* architecture for ubiquitous Web access is a conservative alternative where the internal nodes carry out all personalization services, while the edge nodes act as simple gateways. The motivation behind this choice comes from the observation that, in a two-level topology, only the internal nodes guarantee an adequate security level for the management of sensitive information that may be contained in the user profiles. As the *internal oriented* architecture does not distribute user profile information to the untrusted edge nodes, they cannot execute any adaptation service.

In this architecture, we assume that each user is managed by one internal node, hence each internal node stores a set of unique versions of user profiles that are necessary for personalization services. The information of each user profile is considered as an atomic set of data that can not be fragmented. The user profiles are mapped on the internal nodes through an hash function $H(x)$, that receives a (unique) user ID and returns an internal node identifier $k = H(\mathrm{ID})$, where $k \in [1, \ldots, n]$, and n is the number of internal nodes of the two-level topology. Our experience with MD5-like hash functions demonstrates that $H(x)$ provides a fair load sharing of the users among the internal nodes.

The edge nodes act as gateways that are responsible for the identification of the internal node that owns the user profile necessary for the content personalization. To this purpose, they implement the same $H(x)$ function. For the simplicity of the provided tasks, the edge nodes can be lightweight systems, with reduced computational and storage capabilities. A consequence of this architectural choice

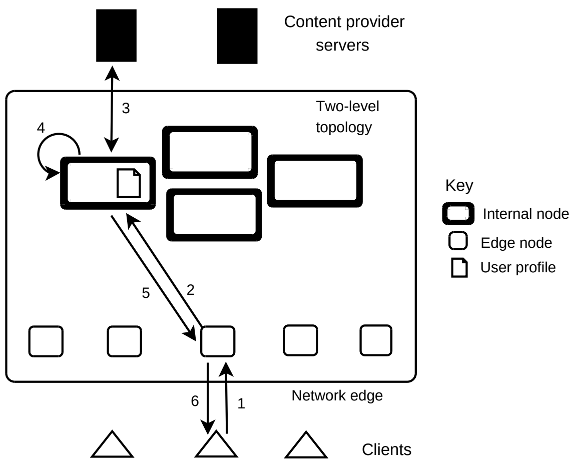

Figure 2. Request service in the *internal oriented* **architecture**

is that we can spread a very large number of edge nodes around the points of presence of the ISPs.

On the other hand, any internal node can provide all personalization services, hence the amount of computational power that may be required to each of them is really high. For performance and availability reasons, we prefer to assume that each logical internal node actually consists of a cluster of physical nodes. A Web cluster is a common architecture for high performance Web-based systems. It is composed by two or more server nodes that are distributed over a local area network, and by a front-end Web switch that masks the node replication. For major details, the interested reader can refer to [3].

Figure 2 illustrates the steps for serving a Web resource through the *internal oriented* architecture. When the edge node receives the client request (step 1), it extracts the user ID and applies the hash function $H(x)$ to identify the internal node that owns the user profile. The request is forwarded to the selected internal node to perform adaptation (step 2). After the internal node has fetched the Web page (HTML container and embedded objects) from the content provider server (step 3), it carries out all necessary personalization services on the basis of the user profile information (step 4). Finally, the adapted resources are sent to the edge node (step 5), and then delivered to the client (step 6).

5 *Internal-edge oriented* architecture

The *internal-edge oriented* architecture relaxes the constraints of the *internal oriented* architecture by distributing adaptation services over both internal and edge nodes. In particular, it preserves the privacy of sensitive information of the user profiles by placing personalization services that require sensitive information only on the trusted internal nodes. Content adaptation services not requiring sensitive information, such as resource transcoding or virus scanning, are carried out on the edge nodes. The goal is to improve user perceived performance through a twofold action: to distribute adaptation services on two nodes and to move some services closer to the clients. This choice for mapping the adaptation services has also another motivation: the “less personalized” content has more probability to be useful for other users, hence edge nodes can exploit caching services and save on costs of onerous adaptation operations, such as image/video transcoding.

The *internal-edge oriented* architecture choices have an important impact on the design of the infrastructure: unlike the *internal oriented* architecture, now powerful servers are required for both internal and edge nodes.

Even more important, the *internal-edge oriented* architecture needs a more sophisticated management of the user profiles. Indeed, it is necessary to introduce some mechanisms for copying and distributing parts of a user profile to the edge node that has been contacted by the client. The profile information is classified in two parts: *sensitive* and *impersonal* information. The classification is done by default, but each user has the possibility of choosing the right set of permission parameters for himself. The user choice influences the adaptation services that can be carried out by the edge nodes (that is, those requiring just impersonal information) and the personalization services that must run on the internal nodes because they require sensitive data.

We now present our solution to the problem of making available to an edge node the decision about which services it has to provide to a user request and also the fragments of the user profile that are necessary for the adaptation services. We should also recall that a user request for a Web resource typically consists of the base HTML file (the so called, container) and multiple embedded objects.

From the observation that each internal node manages a set of complete user profiles and the related differentiation between sensitive and impersonal information, it seems convenient to let the internal nodes to take all decisions about the choices for adaptation, and to communicate to the edge nodes both the decisions and the information they need to carry out their adaptation service. The decision is driven by the classification of the user profile information, and is communicated to the edge nodes through the response for the HTML container. Even the user profile information that is necessary to the edge node is piggybacked on this response for the first file.

We describe the details of this mechanisms through the Figure 3 that shows in separate images the operations related to the base HTML container and to the embedded objects. To avoid complications in this figure, we do not represent the operations due to possible caching of the content

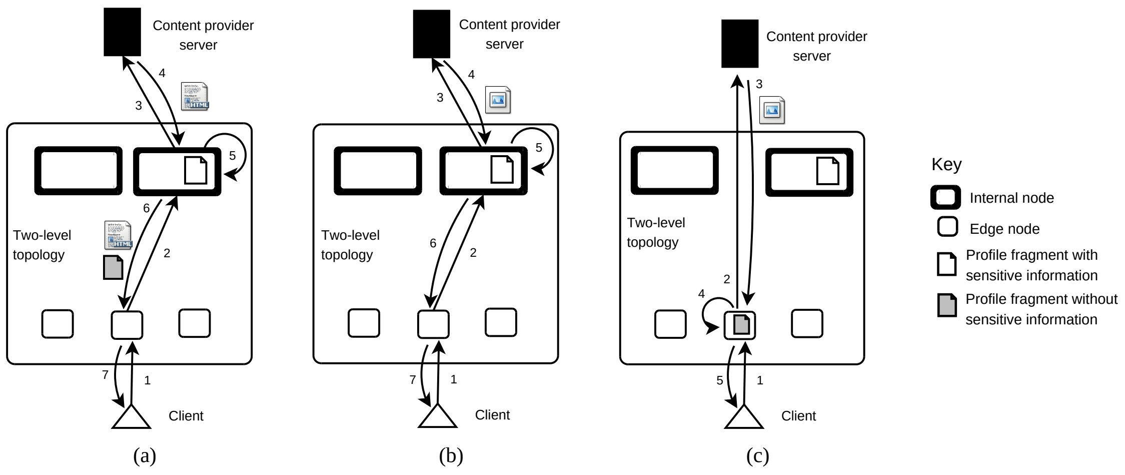
Content provider server
Two-level topology
Client
Key
Internal node
Edge node
Profile fragment with sensitive information
Profile fragment without sensitive information
(a)
(b)
(c)

Figure 3. Request service in the *internal-edge oriented* architecture

at the internal or edge nodes. We will return on caching mechanisms and their effect at the end of this section. The first request issued by a client for accessing a Web resource is the request for the HTML container, which is described in Figure 3(a). The request is sent by the client to the edge node (step 1), that forwards the request to the internal node owning the user profile. The node identification is done through the hash function described in Section 4 (step 2). The selected internal node fetches the HTML page from the content provider server (steps 3 and 4) and, if necessary, carries out content adaptation (step 5). Then, the internal node considers the Web objects embedded within that page and the classification of the user profile information. From this analysis, the internal node can decide which objects can be adapted by the edge node and which must be adapted by itself. When the HTML container is sent back to the edge (step 6) and then to the client (step 7), the user profile information that is required for content adaptation at the edge node is piggybacked in the response. The sequence of the steps for managing requests that need personalization based on sensitive information is described in Figure 3(b). The request service is similar to the case of processing the HTML container, with the only difference that no additional information is piggybacked in the internal-to-edge response (step 6). The case of content adaptation not requiring sensitive information is described in Figure 3(c): the client request is received by the edge node (step 1) which fetches the Web object directly from the content provider server (steps 2 and 3). Since the user profile information has been already copied on the edge node, the content adaptation is carried out immediately on the edge node (step 4) before sending the adapted resource back to the client (step 5).

Introducing a resource caching service on the edge nodes may alter the sequences of steps for servicing a request with respect to the operations shown in Figure 3(c): at the arrival of a client request, a cache lookup is carried out; in the case of cache hit, the fetch operation becomes unnecessary, with a consequent performance gain. This mechanism allows further improvements when already adapted resources are found in the cache. The performance benefits due to caching in the context of ubiquitous Web are well known in literature [2], however caching cannot be applied to every content adaptation service. We adopt caching only for impersonal adaptation services because of privacy problems related to results achieved through sensitive information. Moreover, highly personalized resources lead to an explosion of the working set size, as suggested in [2], with a consequent reduction of the effectiveness of caching. In other words, the cache hit rate is reduced by the large space of personalization parameters.

6 Experimental results

To evaluate the performance of the *internal oriented* and *internal-edge oriented* architectures for ubiquitous Web access we implemented two prototypes based on the popular Apache Web server version 2.0. The profile management software is written in Perl and is processed by the Web server using the *mod_perl* module. This choice allows us to combine the flexibility of an high level programming language with the efficiency of an Apache module that can interact with the Web server internals. The user profiles are stored as XML files that are accessed through the profile management software layer.

To exercise our prototypes, we use a workload model based on traces collected from a real Web site. HTML resources contain embedded objects ranging from a minimum of 2 to a maximum of 18, with a mean value of 10. The

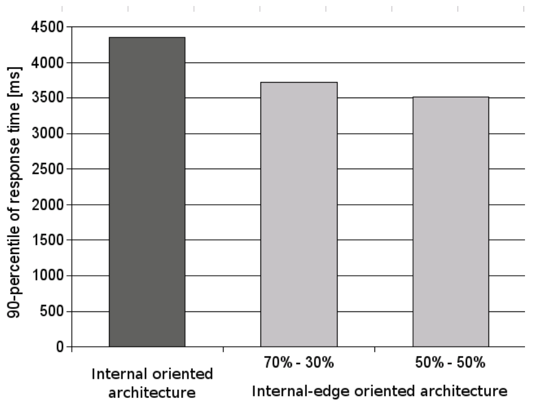

Figure 4. 90-percentile of the page response time

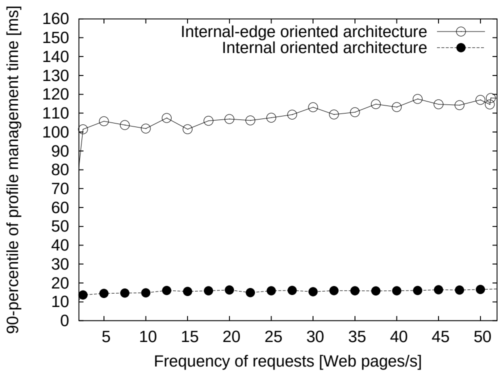

Figure 5. Overhead of profile management

mean size of a single embedded object is 8.5 KB. The service times of the content adaptation services are characterized by heavy-tailed distributions, with 140 and 350 ms for median and 90-percentile, respectively. We assume that the time of an adaptation service does not change if the service is performed on an edge or an internal node.

For our experiments, we set up a system with client nodes running the load generator *httperf* [9]. The two-level infrastructure consists of 4 logical internal nodes (each composed of 2 server nodes) and 8 edge nodes that are equipped with our prototypes. Another node hosts the Web server with the original Web resources. The client and the nodes of the two-level infrastructure are connected through a fast Ethernet network, while the Web server is placed in a remote location. The clients issue requests for 3000 pages (and related embedded objects) that reach the edge nodes at the rate of 5 pages per second. The client requests are evenly distributed among all edge nodes.

We compare the performance of the *internal oriented* architecture with that of the *internal-edge oriented* architecture. For the *internal-edge oriented* architecture we consider two scenarios that are denoted by the amount of content adaptation carried out on the edge nodes. We call the two scenarios 70%-30% and 50%-50%, that indicate that 30% and 50% of the adaptation services are provided by the edge nodes, respectively.

Figure 4 shows the 90-percentile of the page response time for the three cases. The 70%-30% *internal-edge oriented* architecture achieves a performance gain close to 20% with respect to the *internal oriented* architecture. The main motivation for this result is due to the possibility of using two server nodes in parallel in the *internal-edge oriented* architecture. Passing from the 70%-30% to the 50%-50% scenario allows a further, although more limited (close to 5%) performance improvement.

From these results we can conclude that the *internal-edge oriented* architecture is preferable when we have a significant amount of adaptation services which do not require sensitive information. However, the *internal-edge oriented* architecture requires a more sophisticated management of user profiles. We carried out some preliminary experiments, on the profile management overhead in the *internal-edge oriented* and *internal oriented* architectures. We evaluate the time required by user profile access for increasing request rates, without considering the cost of the actual content adaptation service. Figure 5 shows the 90-percentile of profile access time for the two architectures. Both architectures are scalable as shown by the profile management time that remains almost constant for the request rate increasing by more than on order of magnitude. The most interesting result of the overhead evaluation is that the cost of profile distribution on the edge nodes in the *internal-edge oriented* architecture is nearly on order of magnitude higher if compared to the access to a locally available user profile (as in the case of the *internal oriented* architecture). This cost is mainly due to the XML parsing and processing of profile information occurring before the information transfer from the internal to the edge node.

Due to these considerations, it may be interesting to evaluate the opportunity for caching the profile fragments not containing sensitive information on the edge nodes of the two-level topology, instead of piggybacking the information to the edge node for every client request.

7 Related work

The ubiquitous Web access has been recognized in literature as a fundamental challenge [15, 12]. Multiple proposals for content adaptation services are present in literature, both in the field of simple Web content transcoding [4] and in the case of more complex pervasive computing applications [11, 1]. However, such studies focus mainly on the heterogeneity of client devices, without taking into account the issues related to the privacy of sensitive information contained in the user profiles. A noteworthy example of these proposals is a peer-to-peer content adaptation system [13], called Tuxedo, which allows ubiquitous Web access providing both personalization and transcoding services, but the study does not evaluate in deep detail the issues arising from the distribution of sensitive information among untrustworthy nodes.

The importance of providing privacy in the user profile management has been pointed out recently in literature [5, 7]. The need of taking into account privacy guarantees in the design of scalable distributed architectures is confirmed by P3P (Platform for Privacy Preferences) [10], a W3C proposal that suggests a mechanism for Web sites to encode their privacy policies in a standardized format that can be easily retrieved and interpreted by user agents. However, these studies are more tailored to a server-side approach for the generation of personalized Web content rather than to the intermediary-based model for Web content adaptation considered in this paper.

8 Conclusions

The design of efficient architectures providing personalized adaptation services has to address the issues of handling sensitive information that may be contained in the user profiles.

The proposed architectures, namely *internal oriented* and *internal-edge oriented*, follow two opposite approaches to map content adaptation services over the nodes of a two-level topology. The *internal oriented* architecture carries out content adaptation on the internal nodes, while the *internal-edge oriented* architecture moves adaptation services not requiring sensitive information over the edge nodes of the distributed infrastructure. Both architectures provide a high level of privacy in the management of sensitive information contained in the user profiles and are highly scalable.

The message that emerges from our analysis is that the *internal-edge oriented* architecture is preferred when a large fraction of the profiles does not contain sensitive information. On the other hand, the *internal oriented* architecture seems the best alternative for the future if the trend towards personalized services continues to grow.

References

[1] P. Bellavista, A. Corradi, and C. Stefanelli. The ubiquitous provisioning of internet services to portable devices. *IEEE Pervasive computing*, 2002.

[2] C. Canali, V. Cardellini, M. Colajanni, R. Lancellotti, and P. S. Yu. A two-level distributed architecture for web content adaptation and delivery. In *Proc. of The IEEE Symposium on Applications and the Internet (SAINT 2005)*, Trento (IT), Jan./Feb. 2005.

[3] V. Cardellini, E. Casalicchio, M. Colajanni, and P. S. Yu. The state of the art in locally distributed web-server systems. *ACM Comput. Surv.*, 34(2):263–311, 2002.

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

[5] R. K. Chellappa and R. G. Sin. Personalization versus privacy: An empirical examination of the online consumer's dilemma. *Information Technology and Management*, 6(2-3), April 2005.

[6] M. Colajanni and R. Lancellotti. System architectures for web content adaptation services. *IEEE Distributed Systems online*, May 2004.

[7] R. Hull, B. Kumar, D. Lieuwen, P. F. Patel-Schneider, A. Sahuguet, S. Varadarajan, and A. Vyas. Enabling context-aware and privacy-conscious user data sharing. In *Proc. of 2004 IEEE International Conference on Mobile Data Management (MDM'04)*, 2004.

[8] N. Milanovic and M. Malek. Current solutions for Web service composition. *IEEE Internet Computing*, Nov. 2004.

[9] D. Mosberger and T. Jin. httperf - a tool for measuring web server performance. In *Proc. of Workshop on Internet Server Performance*, Wisconsin, jun 1998.

[10] Platform for privacy preferences project, 2005. `http://www.w3.org/P3P`.

[11] H. Rao, Y. Chen, D. Chang, and M. Chen. imobile: a proxy-based platform for mobile services. In *Proc. of the 1st workshop on wireless mobile Internet (WMI2001)*, 2001.

[12] D. Saha and A. Mukherjee. Pervasive computing: A paradigm for the 21st century. *IEEE Computer*, 36(3), Mar. 2003.

[13] W. Shi, K. Shah, Y. Mao, and V. Chaudhary. Tuxedo: a peer-to-peer caching system. In *Proc. of PDPTA03*, Las Vegas, NV, June 2003.

[14] B. Srivastava and J. Koehler. Web service composition-current solutions and open problems. In *Proc. of ICAPS 2003 Workshop on Planning for Web Services*, 2003.

[15] G. C. Vanderheiden. Anywhere, anytime (+ anyone) access to the next-generation www. *Computer Networks and ISDN Systems*, 8(13), Sep. 1997.